ISMIR 2012 PORTO

Proceedings of the

13th International Society for Music Information Retrieval Conference

MOSTEIRO S.BENTO DA VITÓRIA
PORTO, PORTUGAL

8-12 OCTOBER 2012

http://ismir2012.ismir.net

Edited by

Fabien Gouyon
Perfecto Herrera
Luis Gustavo Martins
Meinard Müller

ISMIR 2012

Proceedings of the 13th International Society for Music Information Retrieval Conference

October 8th-12th, Porto, Portugal

Edited by
Fabien Gouyon, Perfecto Herrera,
Luis Gustavo Martins, and Meinard Müller

ISMIR 2012 is organized by the International Society for Music Information Retrieval.
Website: http://ismir2012.ismir.net/

Cover design and ISMIR 2012 logo by Raquel Morais (raquelmorais70@gmail.com)

Edited by:

Fabien Gouyon (INESC TEC, Porto)
Perfecto Herrera (UPF, Barcelona)
Luis Gustavo Martins (UCP, Porto)
Meinard Müller (MPI Informatik, Saarbrücken)

Printed by FEUP Edições (http://feupedicoes.fe.up.pt/)
ISBN 978-972-752-144-9

Copies may be ordered from:
FEUP edições
Biblioteca - Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias
4200-465 PORTO
email: feuped@fe.up.pt

Conference Committee

Conference Chairs
Fabien Gouyon (INESC TEC)
Carlos Guedes (INESC TEC)

Scientific Chairs
Perfecto Herrera (UPF, Barcelona)
Luis Gustavo Martins (UCP, Porto)
Meinard Müller (MPI Informatik, Saarbrücken)

Music Chairs
Carlos Guedes (INESC TEC)
Pedro Rebelo (SARC, Belfast)

Music Curators
Francois Pachet (Sony CSL, Paris)
Daniel Teruggi (INA-GRM, Paris)

Tutorial Chair
Emilia Gómez (UPF, Barcelona)

Late-breaking news/demo Chair
Jean-Julien Aucouturier (LEAD, Dijon)

Program Committee:

Juan Pablo Bello (New York University, USA)
Òscar Celma (Gracenote, USA)
Elaine Chew (Queen Mary University of London, UK)
Darrell Conklin (Universidad del País Vasco, Spain)
Sally Jo Cunningham (University of Waikato, New Zealand)
Matthew Davies (INESC, Portugal)
Simon Dixon (Queen Mary University of London, UK)
J. Stephen Downie (University of Illinois, USA)
Douglas Eck (Google, USA)
Dan Ellis (Columbia University, USA)
Rebecca Fiebrink (Princeton University, USA)
Arthur Flexer (Austrian Research Institute for Artificial Intelligence, Austria)
Ichiro Fujinaga (CIRMMT, McGill University, Canada)
Masataka Goto (AIST, Japan)
Emilia Gómez (Universitat Pompeu Fabra, Spain)
Anssi Klapuri (Queen Mary University of London, UK)
Peter Knees (Johannes Kepler University, Austria)
Kjell Lemström (Laurea University of Applied Sciences, Sweden)
Hiroshi G. Okuno (Kyoto University, Japan)
Nicola Orio (University of Padova, Italy)
François Pachet (SONY CSL, France)
Bryan Pardo (Northwestern University, USA)
Geoffroy Peeters (IRCAM, France)
Gaël Richard (Ecole Nationale Superieure des Telecommunications, France)
Robert Rowe (New York University, USA)
Markus Schedl (Johannes Kepler University, Austria)
Joan Serrà (Artificial Intelligence Research Institute, IIIA-CSIC, Spain)
Malcolm Slaney (Yahoo Research Inc., USA)
Godfried Toussaint (New York University Abu Dhabi, Abu Dhabi)
Douglas Turnbull (Ithaca College, USA)
George Tzanetakis (University of Victoria, Canada)
Emmanuel Vincent (INRIA, France)
Anja Volk (Utrecht University, The Netherlands)
Ye Wang (National University of Singapore, Singapore)
Gerhard Widmer (Johannes Kepler University, Austria)
Frans Wiering (Utrecht University, The Netherlands)
Geraint Wiggins (Queen Mary University of London, UK)

Reviewers

Samer Abdallah
Jakob Abeßer
Teppo E. Ahonen
Christina Anagnostopoulou
Régine André-Obrecht
Amélie Anglade
Josep Ll. Arcos
Andreas Arzt
Jean Julien Aucouturier
Federico Avanzini

Ana María Barbancho
Isabel Barbancho
Mathieu Barthet
Stephan Baumann
Dominikus Baur
Mert Bay
Juan Pablo Bello
Emmanouil Benetos
Thierry Bertin-Mahieux
Frédéric Bimbot
Dmitry Bogdanov
N. Boulanger-Lewandowski
Nick Bryan
John Ashley Burgoyne
Don Byrd

Marcelo Caetano
Emilios Cambouropoulos
Sergio Canazza
Mark Cartwright
Òscar Celma
Taylan Cemgil
Ching-Wei Chen
Elaine Chew
Kahyun Choi
Parag Chordia
Ching-Hua Chuan
Nicholas Collins
Darrell Conklin
Arshia Cont
Olmo Cornelis
Emanuele Coviello
Tim Crawford
Ian Cross
Sally Jo Cunningham
Michael Cuthbert

Matthew Davies
Bas de Haas
Johanna Devaney
Emanuele Di Buccio
Giorgio Maria Di Nunzio
Christian Dittmar
Simon Dixon
Marcos Domingues
J. Stephen Downie
Karin Dressler
Zhiyao Duan
Schlomo Dubnov
Jean-Louis Durrieu

Douglas Eck
Tuomas Eerola
Andreas Ehmann
Dan Ellis
Valentin Emiya
Slim Essid
Sebastian Ewert

Mary Farbood
Rebecca Fiebrink
Ben Fields
Thomas Fillon
Derry Fitzgerald
Arthur Flexer
Sebastian Flossmann
Judy Franklin
Ferdinand Fuhrmann
Hiromasa Fujihara
Ichiro Fujinaga

Jörg Garbers
Martin Gasser
Werner Goebl
Emilia Gómez
Paco Gómez
Masataka Goto
Maarten Grachten
Lars Graugaard
Thomas Grill
Peter Grosche
Cynthia Grund

Philippe Hamel
Jinyu Han
Andrew Hankinson
Pierre Hanna
Martín Haro
Christopher Harte
Toni Heittola
Mikael Henaff
Keiji Hirata
Jason Hockman
Matt Hoffman
André Holzapfel
Henkjan Honing
Aline Honingh
Xiao Hu
Anna Huang
Eric Humphrey

José Manuel Iñesta
Charle Inskip
Eric Isaacson
Akinori Ito
Ozgur Izmirli

Kurt Jacobson
Jordi Janer
Dalwon Jang
Tristan Jehan
Kristoffer Jensen
Cyril Joder
Sergi Jordà

Florian Kaiser
Hirokazu Kameoka
Blair Kaneshiro
Haruhiro Katayose
Robert Keller
Johannes Kepper
Mooyoung Kim
Minje Kim
Anssi Klapuri
Peter Knees
Ian Knopke
Gopala Koduri
Alessandro Koerich
Filip Korzeniowski
Florian Krebs
Frank Kurth

Antti Laaksonen
Mathieu Lagrange
Cyril Laurier
Jin Ha Lee
Kyogu Lee
Kjell Lemström
Mark Levy
David Lewis
Zhonghua Li
Cynthia C. S. Liem
Adam Lindsay
João Lobato Oliveira
Lie Lu

Robert Macrae
Michael Mandel
Matija Marolt
Agustín Martorell
Matthias Mauch
Panayotis Mavromatis
Brian McFee
Cory McKay
Matt McVicar
Lesley Mearns
Massimo Melucci
Annamaria Mesaros
Riccardo Miotto
Dirk Moelants
Nicola Montecchio
Dan Morris
Daniel Müllensiefen
Gautham Mysore

Tomoyasu Nakano
Teresa Nakra
Juhan Nam
Bernhard Niedermayer

Nicolas Obin
Mitsunori ogihara
Yasunori Ohishi
Hiroshi G. Okuno
Lartillot Olivier
Nobutaka Ono
Nicola Orio
Laurent Oudre

François Pachet
Kevin Page
Helene Papadopoulos
Bryan Pardo
Jouni Paulus
Steffen Pauws
Geoffroy Peeters
Graham Percival
Antonio Pertusa
Aggelos Pikkrakis
Polina Proutskova
Laurent Pugin

Marcelo Queiroz
Ian Quinn

Stanislaw Raczynski
Zafar Rafii
Yves Raimond
Rafael Ramírez
Preeti Rao
Christopher Raphael
Nicolas Rasamimanana
Andreas Rauber
Suman Ravuri
Christophe Rhodes
Gaël Richard
David Rizo
Andrew Robertson
Matthias Robine
Martín Rocamora
Marcelo Rodríguez López
Robert Rowe
Pierre Roy

Justin Salamon
Craig Sapp
Andy Sarroff
Isaac Schankler
Markus Schedl
Rafael Schirru
Andrew Schloss
Jan Schlueter
Erik Schmidt
Dominik Schnitzer
Xavier Serra
Joan Serrà

William Sethares
Arun Shenoy
George Sioros
Diana Siwiak
Joren Six
Malcolm Slaney
Jordan Smith
Lloyd Smith
Reinhard Sonnleitner
Mohamed Sordo
Adam Stark
Sebastian Stober
Dan Stowell
Bob L. Sturm

David Temperley
Steve Tjoa
Godfried Toussaint
Douglas Turnbull
George Tzanetakis

Sandra Uitdenbogerd
Julián Urbano

Peter van Kranenburg
Remco Veltkamp
Emmanuel Vincent
Tuomas Virtanen
Anja Volk

Thomas Walters
Ye Wang
Jun Wang
Ge Wang
Ron Weiss
Kris West
Ian Whalley
Gerhard Widmer
Alicja Wieczorkowska
Frans Wiering
Geraint Wiggins

Yi-Hsuan Yang
Kazuyoshi Yoshii
Yi Yu

Table of Contents

Poster Session 3

Preface

We would like to welcome all participants of the 13th International Society for Music Information Retrieval Conference (ISMIR 2012) in the São Bento da Vitória Monastery, a classified National Monument at the very heart of old Porto. Porto is Portugal’s second largest city, located in the estuary of the Douro River in Northern Portugal. This year's edition of the conference is organized by the Institute for Systems and Computer Engineering, Technology and Science (INESC TEC), in partnership with the Teatro Nacional de São João.

The present volume contains the complete manuscripts of all the peer-reviewed papers presented at ISMIR 2012. A total of 253 abstracts were entered in the reviewing system before the deadline, out of which 215 complete and well-formatted papers entered the **review process**. Special care was taken to assemble an experienced and interdisciplinary review panel including people from many different institutions worldwide. As in previous years, the reviews were double-blinded (i.e., both the authors and the reviewers were anonymous) with a two-tier model of review committee (i.e. including reviewers and Program Committee members). Reviewers and PC members were able to bid for papers. Each paper was assigned to a PC member, who could then request the assignment of at least one particular reviewer for each paper. Reviewer assignments were done based on topic preferences, bids and PC member assignments. After the review phase, PC members and reviewers entered a (name-disclosed) discussion phase aiming at homogenizing acceptance vs. rejection decisions. This discussion phase, which did not provide any further interaction with authors, replaced the rebuttal phase as used in past ISMIR conferences.

Final acceptance decisions were based on 894 reviews and meta-reviews written by 260 reviewers and PC members. From the 215 reviewed papers, 95 papers were accepted resulting in an acceptance rate of 44%. A set of 13 papers (from the 95 accepted) were "conditionally accepted", where authors were asked to make mandatory changes. These changes where again checked by PC members. After this second round of revisions, all 13 papers were finally accepted. Compared to previous ISMIR conferences, the review process may have been stricter, leading to a lower overall acceptance rate. In addition, slightly reducing the total number of accepted papers, compared to past years, led to a higher competitiveness. The table shown in the next page summarizes the ISMIR publication statistics over the last years.

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. From the 95 papers, 30 papers were chosen for oral presentation based on the topic and broad appeal of the work, whereas 65 were chosen for poster presentation. Oral presentations have a 20 minutes slot (including setup and questions/answers from the audience) whereas poster presentations are done in 2 sessions per day, the same posters being presented in the morning and in the afternoon of a given day. Poster presenters have the opportunity (in 1 minute and 1 slide) to announce orally their posters during a plenary session (so-called "poster-craze") and "tease" the whole audience, inviting to a later, more personal, scientific discussion around their poster.

	Presentations		Total	Total	Total	Unique	Pages /	Authors /	U. Authors/
Location	Oral	Posters	Papers	Pages	Authors	Authors	Paper	Paper	Paper
Plymouth	19	16	35	155	68	63	4.4	1.9	1.8
Indiana	25	16	41	222	100	86	5.4	2.4	2.1
Paris	35	22	57	300	129	117	5.3	2.3	2.1
Baltimore	26	24	50	209	132	111	4.2	2.6	2.2
Barcelona	61	44	105	582	252	214	5.5	2.4	2
London	57	57	114	697	316	233	6.1	2.8	2
Victoria	59	36	95	397	246	198	4.2	2.6	2.1
Vienna	62	65	127	486	361	267	3.8	2.8	2.1
Philadelphia	24	105	105	630	296	253	6	2.8	2.4
Kobe	38	85	123	729	375	292	5.9	3	2.4
Utrecht	24	86	110	656	314	263	6	2.9	2.4
Miami	36	97	133	792	395	322	6	3	2.4
Porto	**36**	**65**	**101**	**606**	**324**	**264**	**6**	**3.2**	**2.6**

The selected submissions are presented over a period of 3.5 days, preceded by a day of tutorials and followed by half a day of late-break/demo session at an auxiliary venue. Two 3-hours **tutorials** are presented in parallel during Monday morning, and two in parallel during the afternoon. In the morning, Josh McDermott, Bryan Pardo, and Zafar Rafii present a tutorial on *Leveraging Repetition to Parse the Auditory Scene*, while Xiao Hu and Yi-Hsuan (Eric) Yang deal with *Music Affect Recognition: The State-of-the-art and Lessons Learned*. In the afternoon, Mark D. Plumbley, Simon Dixon, and Chris Cannam give a tutorial on reusable software and reproducibility in music informatics research. François Pachet, in his afternoon tutorial, explains the basics of jazz, shows why it is interesting, and gives ISMIR attendees insights that can be helpful for designing better (generic and not only jazz-focused) MIR systems.

The ISMIR program also features three distinguished **keynote speakers**, on Tuesday, Wednesday and Thursday after 5 PM. The first keynote is given by José C. Principe, Distinguished Professor of Electrical and Biomedical Engineering at the University of Florida. He will speak about self-organized computational perception in the time-frequency domain. The second keynote is by Maarten de Rijke, full professor of Information Processing and Internet in the Informatics Institute at the University of Amsterdam. He will speak about latest trends in Information Retrieval research and possible links to MIR. The third keynote is by Emmanuel Bigand, full professor of Cognitive Psychology at the University of Burgundy in Dijon. He will speak about cognitive stimulation with music and new technologies.

On Monday at 8 PM, and Tuesday and Wednesday at 10 PM, 1-hour concerts will be held in the main venue. The aim of the ISMIR 2012 music program is two-fold: to encourage the use of MIR techniques in the creation of new music and to explore music that can suggest novel ideas for research in the MIR field. A call for participation was issued which resulted in the submission of 59 music pieces. From these pieces, the music curators Daniel Teruggi and François Pachet selected ten compositions which are presented in the second and third concerts. The first concert features works of the curators and music chairs. An extra "off-ISMIR" concert is held, on Friday night, in the same venue as the late-beak/demo session, featuring compositions using MIR technologies, from young Portuguese composers.

On Thursday morning, the first oral session of the day is dedicated to "Looking back to the past of ISMIR to face the future of MIR", (i.e. the "**MIRrors**" session) in which 6 invited and peer-reviewed papers are presented to reflect on the evolution of the MIR field or a particular MIR topic since 2000. In particular, insights on mid-term future challenges are provided.

On Friday morning, a session on **evaluation initiatives in MIR** features an invited talk by Gareth Jones, Senior Lecturer in the School of Computing at Dublin City University, and a round table with panelists from the annual Music Information Retrieval Evaluation eXchange (MIREX), the Million Song Dataset Challenge and Musiclef/MediaEval. This session provides an overview of diverse MIR evaluation initiatives and discusses current methodologies in MIR evaluations. The session is followed by poster presentations from diverse evaluation initiatives.

The last afternoon of the conference is dedicated to the **Demos/Late-breaking** (D&L) track, which has become a popular and integral part of ISMIR conferences. This year, however, the format of the D&L is changed, following an "unconference" style: there will not be any formal submission nor peer-review. The D&L program is built by participants, both before the event on an open website (http://ismir2012.wikispaces.com/), and even during the conference. Instead of focusing on individuals reporting in a one-to-many way on their personal accomplished research, D&L focuses on capturing those informal yet often insightful and original ideas that typically emerge in collaborative settings at ISMIR (e.g. at coffee breaks, hallways, etc.).

As for **social gathering**, lunches and coffee will be served everyday from Tuesday to Friday. ISMIR 2012 includes in the registration fee a reception on Monday at 6 PM, a banquet on Thursday night, and "reasonable" amounts of Port wine drinking.

We are very proud to present you the proceedings of ISMIR 2012. The conference program was made possible thanks to the hard work of many people including the members of the organization, PC members and reviewers. Special thanks go to this year's sponsors and supporting institutions: Gracenote, Google, Spectral Mind, Quaero, the MIReS European research project, Ableton, Cycling'74, the University of Porto, its Faculty of Engineering, the Catholic University in Porto, and Portugal's Culture Ministry. Last, but not least, ISMIR 2012 program is possible only thanks to the numerous contributions of the MIR community in response to our call for participation. The biggest acknowledgment therefore goes to you, the authors, researchers and participants of this conference.

Fabien Gouyon
Carlos Guedes
ISMIR 2012 Conference Chairs

Perfecto Herrera
Luis Gustavo Martins
Meinard Müller
ISMIR 2012 Scientific Chairs

Keynote Talk 1

José C. Principe
Computational NeuroEngineering Laboratory, University of Florida

Self-Organized Computational Perception in the Time Frequency Domains

Abstract

The auditory cortex in the brain does effortlessly a better job of extraction information from the acoustic world than our current generation of signal processing algorithms. This talk elucidates from some essential ideas in brain models how distributed hierarchical dynamical systems can be utilized to self-organize acoustic perceptions. The proposed architecture is based on Kalman filters with hierarchically coupled state models that stabilize the input dynamics and provide a representation space where sounds produced by different instruments appear distinct, providing therefore features for further recognition. Another characteristic of the methodology is that it is adaptive and self-organizing, i.e. previous exposure to the acoustic input is the only requirement for recognition. Some examples will be provided.

Biography

Jose C. Principe is Distinguished Professor of Electrical and Biomedical Engineering at the University of Florida since 2002. He joined the University of Florida in 1987, after an eight-year appointment as Professor at the University of Aveiro, in Portugal. Dr. Principe holds degrees in electrical engineering from the University of Porto (Bachelor), Portugal, University of Florida (Master and Ph.D.), USA and a Laurea Honoris Causa degree from the Universita Mediterranea in Reggio Calabria, Italy. Dr. Principe interests lie in nonlinear non-Gaussian optimal signal processing and modeling and in biomedical engineering. He created in 1991 the Computational NeuroEngineering Laboratory to synergistically focus the research in biological information processing models. He recently received the Gabor Award from the International Neural Network Society for his contributions. Dr. Principe is a Fellow of the IEEE, Fellow of the AIMBE, past President of the International Neural Network Society, and Past Editor in Chief of the Transactions of Biomedical Engineering, as well as a former member of the Advisory Science Board of the FDA. He holds 5 patents and has submitted seven more. Dr. Principe was supervisory committee chair of 65 Ph.D. and 67 Master students, and he is author of more than 500 refereed publications (3 books, 4 edited books, 14 book chapters, 200 journal papers and 380 conference proceedings).

Keynote Talk 2

Maarten de Rijke
Intelligent Systems Lab, University of Amsterdam

Current Trends in Information Retrieval

Biography

Maarten de Rijke is full professor of Information Processing and Internet in the Informatics Institute at the University of Amsterdam. He holds MSc degrees in Philosophy and Mathematics (both cum laude), and a PhD in Theoretical Computer Science. He worked as a postdoc at CWI, before becoming a Warwick Research Fellow at the University of Warwick, UK. He joined the University of Amsterdam in 1998, and was appointed full professor in 2004. He leads the Information and Language Processing Systems group, one of the leading academic research groups in information retrieval in Europe. During the most recent computer science research assessment exercise, the group achieved maximal scores on all dimensions. De Rijke's current focus is on intelligent web information access, with projects on search and discovery for social media, vertical search engines, machine learning for information retrieval, semantic search and multilingual information. A *Pionier* personal innovational research incentives grant laureate (comparable to an advanced ERC grant), De Rijke has generated over 30MEuro in project funding. With an h-index of 42 he has published over 500 papers, has published or edited over a dozen books, is editor for various journals and book series, and a former coordinator of retrieval evaluation tracks at TREC, CLEF and INEX (Blog, Web, Question answering). He is co-chair for SIGIR 2013, the director of the University of Amsterdam's Intelligent Systems Lab (ISLA), its Information Science bachelor program and its Center for Creation, Content and Technology (CCCT).

Keynote Talk 3

Emmanuel Bigand
LEAD - CNRS, Université de Bourgogne

Cognitive Stimulation with Music and New Technologies

Abstract

During the last 10 years, advances in cognitive neurosciences of music have revealed the potential importance of music for brain and cognitive stimulation. This opens new perspectives for education and health. Music is notably a powerful media that contributes to help stroke patients to recover cognitive and motor functions. Technologies for brain and cognitive stimulation lead to the development of numerous factories ("Happy Neuron", in France for example), but for now, nothing serious has been done with music. I will finish my talk by referring to some examples of new music technologies for cognitive stimulation.

Biography

Emmanuel Bigand is a full professor of Cognitive Psychology at the University of Burgundy in Dijon, France, where he is the director of the LEAD lab (Laboratory for Research on Learning and Development) since 2003. He has received academic degrees in the three disciplines of Applied Mathematics (University of Montpellier, 1984), Musicology (University of Aix-en-Provence, 1987) and Psychology (Ph.D. University of Paris X, 1990), as well as formal training as a professional classical musician (First Contrabass Prize, Conservatoire National de Musique de Versailles). Prof. Bigand's research is concerned with the cognitive aspects of human audition. His research has notably established that, contrary to the traditional views in music education, human's aptitude for music can develop implicitly in the manner of language learning. Since 2007, he has been involved in numerous projects linking musical listening and performance to cognitive stimulation and therapeutic rehabilitation. His recent research has shown that music stimulation can boost linguistic performances in deaf children, and help memorization in Alzheimer patients. Prof. Bigand is the author of more than 70 journal articles, has supervised 11 PhD theses and has been the coordinator for 5 international research programs, including the ongoing ITN EBRAMUS (European Brain and Music) network.

Tutorial 1

Leveraging Repetition to Parse the Auditory Scene

Josh McDermott (New York University, New York, New York, USA)
Bryan Pardo (Northwestern University, Chicago, Illinois, USA)
Zafar Rafii (Northwestern University, Chicago, Illinois, USA)

Abstract

In the last year there has been a convergence of research on the role of repetition in audio parsing by humans and machines. In this tutorial, we will discuss how the concept of repetition can help bootstrap source separation and object recognition in humans and by machine. We will begin by presenting results from research on human sound segregation that indicate that repetition of a sound source provides a neglected but powerful cue for segregation. We will then transition to related work applying repetition to audio source separation. We will first present the REpeating Pattern Extraction Technique (REPET), a novel approach for separating a repeating background from a non-repeating foreground in a musical context. We will then describe extensions of REPET that handle variations in the repeating structure. This will be followed by a discussion of the relationship between these explicitly repetition-based algorithms to other separation algorithms (e.g. robust PCA).

Biographies

Josh McDermott is a research associate in the Center for Neural Science at NYU, studying sound, hearing, and music. He received a B.A. in Brain and Cognitive Sciences from Harvard University, an MPhil in Computational Neuroscience from University College London, a PhD in Brain and Cognitive Sciences from MIT, and postdoctoral training in psychoacoustics at the University of Minnesota. His research addresses sound representation and auditory scene analysis. He is interested in using the gap between human and machine competence to better understand biological hearing and design better algorithms for analyzing sound.

Bryan Pardo is an associate professor in the Northwestern University Department of Electrical Engineering and Computer Science. He received a M. Mus. in Jazz Studies and a Ph.D. in Computer Science, both from the University of Michigan. He is an associate editor for IEEE Transactions on Audio Speech and Language Processing. He has developed speech analysis software for the Speech and Hearing department of the Ohio State University, statistical software for SPSS and worked as a researcher for General Dynamics.

Zafar Rafii is a Ph.D. candidate in Electrical Engineering & Computer Science at Northwestern University. He received a Master of Science in Electrical Engineering from ENSEA in France and from IIT in Chicago. In France, he worked as a research engineer at Audionamix. His current research interests are centered audio analysis and include signal processing, machine learning and cognitive science.

Tutorial 2

Music Affect Recognition: The State-of-the-art and Lessons Learned

Xiao Hu (The University of Hong Kong, Hong Kong)
Yi-Hsuan (Eric) Yang (Academia Sinica, Taiwan)

Abstract

The affective aspect (popularly known as emotion or mood) of music information has gained fast growing attention in Music Information Retrieval (MIR) community. The recent years witnessed an explosive growth of studies on music affect recognition. This tutorial provides ISMIR participants an opportunity to learn a range of topics closely involved in affective indexing of music and to discuss how findings and methods can (or cannot) be borrowed from and applied to other multimedia information types such as speech (audio), images (visual) and movies (audio-visual). Topics in this tutorial include: the most influential psychological models of human emotion; musical, personal, and situational factors of music listening that influence the perception and description of music affect; building emotion taxonomies from online music metadata and social media; best practices of constructing ground truth datasets; approaches to and tools for automatic affect classification and regression; benchmarking and evaluation; a sample of deployed prototyping systems; issues and challenges on affect analysis; and the common ground of affect in music, image and movies. All the tools and systems covered in this tutorial are open source or freeware, and the datasets are available in transformed formats (due to the copyright of the audio and lyrics). The format of the tutorial will include lectures, group discussions, demonstration of sample systems and technical results with illustrative musical examples, and spontaneous interactions between the presenters and the audience.

This tutorial is for general MIR researchers and students who are interested in the affective aspects of music information. It would also benefit designers and developers of music retrieval systems and services. No background in music or music processing is required.

Biographies

Xiao Hu is an Assistant Professor at the Library and Information Science Program at the University of Denver. She obtained her PhD in Library and Information Science in 2010 and Master of Computer Science in 2008 from the University of Illinois at Urbana-Champaign. Dr. Hu has been studying music mood taxonomy and classification since 2006 and has won awards at international conferences.

Yi-Hsuan Yang is an Assistant Research Fellow at Academia Sinica, Taiwan, where he leads the Music and Audio Computing Lab. Dr. Yang obtained his Ph.D. degree in 2010 from National Taiwan University. His research interests include music information retrieval, multimedia systems, machine learning, and affective computing. He was awarded the 2011 IEEE SPS Young Author Best Paper Award. In 2011, Dr. Yang co-authored a book entitled Music Emotion Recognition published by CRC Press.

Tutorial 3

Reusable Software and Reproducibility in Music Informatics Research

Chris Cannam (Centre for Digital Music, Queen Mary University of London, UK)
Simon Dixon (Centre for Digital Music, Queen Mary University of London, UK)
Mark D. Plumbley (Centre for Digital Music, Queen Mary University of London, UK)

Abstract

The need to develop and reuse software to process data is almost universal in music informatics research. Many methods, including most of those published at ISMIR, are developed in tandem with software implementations and some of them are too complex or too fundamentally software-based to be reproduced readily from a published paper alone. For this reason, it is helpful for sustainable research to have software and data published along with papers. In practice, however, non-publication of code and data is still the norm. In this tutorial we will discuss barriers to publication of software and data and present a hands-on session in which attendees will explore tools and methods to help overcome these barriers. The tutorial will rapidly cover the use of version control software, code-hosting facilities, aspects of testing and provenance, and software licensing for publication. Examples will be drawn from the music and audio fields, and help will be provided by researchers-developers from the Centre for Digital Music (C4DM), Luís Figueira and Steve Welburn. This tutorial will be of immediate interest to researchers within the music informatics community, and will be highly relevant to research supervisors and research group leaders with an interest in policy and guidance.

Biographies

Chris Cannam is principal developer for the Sound Software project. His work with the C4DM includes the widely used Sonic Visualiser audio analysis software.

Simon Dixon leads the Music Informatics area of C4DM and the Sound Data Management Training project. Among other responsibilities, he is President-elect of ISMIR.

Mark D. Plumbley is Director of C4DM. He leads the Sound Software project and coordinates the EPSRC Digital Music Research Network and the ICA Research Network.

Acknowledgements

Supported by the EPSRC through the SoundSoftware project (http://soundsoftware.ac.uk/) and by JISC through the Sound Data Management Training and Sustainable Management of Digital Music Research Data projects (http://rdm.c4dm.eecs.qmul.ac.uk).

Tutorial 4

Why is Jazz Interesting?

François Pachet (Sony CSL, Paris)

Abstract

Jazz is a lively music genre which has long been a favorite genre of music for computer music research. This tutorial aims at explaining the basics of jazz, show why it is interesting and give ISMIR attendees insights that can be helpful for designing better jazz-focused MIR systems. I will show that jazz is a game that is based, on first approximation, on well-defined rules. I will describe these rules in a non-technical way, understandable by non-jazz specialists. I will also cover some more advanced topics such as the use of side-slips as a reasoned mechanism to play "outside the rules". The tutorial uses many video and audio examples throughout. It has been presented twice already with great success.

Biography

François Pachet received his Ph.D. and Habilitation degrees from Paris 6 University (UPMC). He is a Civil Engineer (Ecole des Ponts and Chaussées) and was Assistant Professor in Artificial Intelligence and Computer Science, at Paris 6 University, until 1997.

He then set up the music research team at SONY Computer Science Laboratory Paris, where he developed the vision that metadata can greatly enhance the musical experience in all its dimensions, from listening to performance. His team conducts research in interactive music listening and performance and musical metadata and developed several innovative technologies (constraint-based spatialization, intelligent music scheduling using metadata) and award winning systems (MusicSpace, PathBuilder, The Continuator for Interactive Music Improvisation, etc.). He is the author of over 80 scientific publications in the fields of musical metadata and interactive instruments.

His current research focuses on creativity and content generation, as he was recently awarded an ERC Advanced Grant to develop the concepts and technologies of "flow machines": a new generation of content generation tools that help users find and develop their own "style".

MIRrors

ISMIR 2012 features a special session based on the idea of "looking back to the past of ISMIR to face the future of MIR" or “the past of Music Information Research reflects on its future”. As the session name indicates, it offers different types of reflections on MIR, while learning from past erRORS to shape the future of our discipline. The session has been crafted in the context of the European project MIRES (http://www.mires.cc/), targeted to elaborate a roadmap for Music Information Research.

A special call for challenging and thought-provocative or controversial positional papers, was made. We were looking forward for papers that, for example: i) reflected on why a particular topic has failed, is systematically not improving, and why it may be doomed to continue this way; ii) reflected on negative results that have, or have not, had the proper impact on MIR research; iii) traced and explained the evolution of a given idea through different editions of ISMIR; iv) provided a review of the impact of a particular idea on the MIR community; v) provided a review of the impact of a particular MIR idea/topic on other conferences or neighboring fields of science; vi) proposed replication studies, in particular showing discrepancies between commonly accepted ideas in the MIR community and reality.

Submissions followed a different schedule than the regular papers but the same rigorous and blind peer-reviewing process that characterizes the regular track of ISMIR. Six out of a total of eight submissions were finally selected for the oral session. They cover a broad spectrum of essential topics, though there it seems to be some pressure to deal with problems raised when considering the human factor involved in MIR systems (either as users or as listeners). The accepted papers were made available two months before the conference in order to facilitate the right mindset and the preparation of questions and interaction with the authors.

It is expected that extended and improved versions of the submitted papers plus some additional new submissions become a special issue of the International Journal of Intelligent Information Systems, to be published by August 2013.

Panel session on Evaluation Initiatives in MIR

Chair
Geoffroy Peeters (IRCAM, Paris)

Panelists
Brian McFee (University of California at San Diego)
Nicola Orio (University of Padova)
Julián Urbano (University Carlos III of Madrid)
J. Stephen Downie (University of Illinois at Urbana-Champaign)

Invited speaker
Gareth Jones (Dublin City University)

The aim of this panel is to discuss the methodologies currently used in MIR evaluations and compare them to the evaluation practices in other research fields. For this, we invited key-actors of current MIR benchmarking initiatives (MIREX, Million-Song-Dataset Challenge and MusicClef/MediaEval), of MIR meta-evaluations and one of the key actors of IR evaluation.
Among the potential topics to be discussed are:

- **Definition of the tasks** to be evaluated. What methodology should be used to define the task (bottom-up vs. top-down)? For which purpose should a task be evaluated: low-level tasks (functionality-oriented such as beat, chords) vs. full-system tasks (use-case-oriented such as music recommendation systems). Specific tasks that are part of large-scale international evaluations define de facto the specific topics that new contributors to the MIR field will work on. The methodology followed to define tasks is therefore of utmost importance.
- **Evaluation**. How should a specific task be evaluated? Which data, which measures, what is the reliability of the results obtained?
- **Data**. How to get more data? How to deal with data availability (not only music collections, but also raw system outputs, judgments, annotations)? Should we go to low-cost evaluation methodology (see TREC Million Query Track 2007, 2008 and 2009)? Currently most MIR systems are concerned with audio-only or symbolic-only scenario. Multi-modal systems (such as aggregating information from the audio-content, from lyrics content or web mining) should allow deciding also on the impact on final user application of each technology.
- What is the best methodology to drive improvements? What kind of evaluation framework (open vs close evaluation)? What could be improved in previous evaluation initiatives? How can we make results reproducible? How can we make MIR evaluation sustainable along time?

Agenda

Introduction
Short overview of diverse MIR evaluation initiatives
. MIREX and MIREX-Next-Generation
. MillionSong Dataset Challenge
. MusiClef/MediaEval
Invited Talk by Gareth Jones
Round table on current methodologies in MIR evaluations
MIR technologies performance in 2012 - Results
MIR technologies performance in 2012 - Posters

INFLUENCE IN EARLY ELECTRONIC DANCE MUSIC: AN AUDIO CONTENT ANALYSIS INVESTIGATION

Nick Collins
University of Sussex
N.Collins@sussex.ac.uk

ABSTRACT

Audio content analysis can assist investigation of musical influence, given a corpus of date-annotated works. We study a number of techniques which illuminate musicological questions on genre and creative influence. By applying machine learning tests and statistical analysis to a database of early EDM tracks, we examine how distinct putatively different musical genres really are, the retrospectively labelled Detroit techno and Chicago house being the core case study. Further, by building predictive models based on works from earlier years, both by a priori assumed genre groups and by individual tracks, we examine questions of influence, and whether Detroit techno really is a sort of electronic future funk, and Chicago house an electronic extension of disco. We discuss the implications and prospects for modeling musical influence.

1. INTRODUCTION

Genre is a contentious area at the best of times [1], but an especial minefield in electronic dance music, where producers, journalists and consumers are always eager to promote new micro-genres [12]. As Brewster and Broughton have written of one highly strained genre term 'if you name a genre of music after a club which was open for ten whole years and which was known for its eclecticism, you're going to run into problems of definition pretty quickly. The word 'garage' is by far the most mangled term in the whole history of music' [4, p. 307]. [1]

Electronic dance music's origins range across African-American music and European synth pop, against a backdrop of increasingly affordable synthesis and sampling technology [6, 11, 15, 17, 18]. The important role through the 1980s of the US cities Chicago and Detroit as crucibles of new club music is unassailable, though they were not the only centres of activity (New Yorks' frenetic hip hop developments into electro, or the post-Moroder italo-disco movement in Europe are also worth mentioning, as indeed are general trends to danceable and synthesizer-laden pop throughout the 1980s mainstream). The cities are enshrined in the genre names Chicago house and Detroit techno as two foundational pillars of later electronic dance music: they form the core of the study in this paper, though we do not assume without investigation that they are really as distinct as their names imply.

Even before audio content analysis investigation, there are good reasons for a musicologist to be wary of treating house and techno too individually in their 1980s growth. Chicago is around a five hour drive from Detroit, and Detroit artists often went to Chicago to sell their records in the larger market there; Derrick May sold Frankie Knuckles his TR-909 drum machine! The term 'techno' has many precedents, including track titles from Buggles, Yellow Magic Orchestra and Kraftwerk, although most famously used in *Techno City*, a 1985 Cybotron track co-produced by Juan Atkins (the elder of Techno's 'Belleville Three'). The genre term was finally applied as a differentiating stamp in 1988 for the *Techno! The New Dance Sound of Detroit* compilation curated by Neil Rushton, at Juan Atkins' insistence on techno over Derrick May's 'Detroit house.' Nevertheless, the compilation itself includes a 'megamix' at its close called *Detroit is Jacking* (jacking being a standard Chicago dance term) and another track called *Share this house* by Members of the House!

Detroit artists have themselves attempted to characterise musical differences with Chicago. In the liner notes to the *Techno!* compilation Derrick May writes 'House still has its heart in 70s disco; we dont have any of that respect for the past, its strictly future music. We have a much greater aptitude for experiment' (sic) [9], and most famously, that 'It's like George Clinton and Kraftwerk are stuck in an elevator with only a sequencer to keep them company' [9]. The hypothesis of Chicago house as an extension of disco, and Detroit techno as a combination of electronic funk and synth pop, will be examined herein.

Two previous studies of musical influence [5, 8] published in ISMIR, on synth pop and sampling, have indicated the benefits of data-annotated corpora in new musicological investigations. Through musical similarity measures, this paper examines the use of automatic audio content analysis to establish the strength of links between historic tracks and putative genre groupings. Where Bryan and Wang [5] worked on an existing database of annotations over sample-based music concerning 'WhoSampled who' (http://www.whosampled.com/), Collins [8] examined audio similarity between date-annotated synth pop as

[1] 'hardcore' is another example of a heavily overloaded genre term.

Genre	Dates	Num Tracks	Duration (mins)	Notes
Chicago House	1986-1989	31	197.7	Sourced in particular from *Chicago Trax* and *The Original Chicago House Classics* as well as compilations including *Warp10+1:Influences*
Detroit Techno	1986-1989	31	186.5	Including Derrick May, early Model 500, and the *Techno!* compilation. No second wave, nor Cybotron
1980s Pop	1985-1989	31	127.7	Including Michael Jackson, Madonna, Prince
Funk	1965-1978	31	118.6	Selected tracks from Parliament's *Mothership Connection*, *The Godfather, James Brown, The Very Best Of...* and *Funk Soul Classics*
Disco	1973-1980	31	112.5	Selected tracks from *Anthems Disco* and *Disco Fever*
Synth Pop	1977-1981	31	145.6	Including Kraftwerk, Human League, Gary Numan, Ultravox, Depeche Mode
Electro and Hip Hop	1980-1984	31	180.6	Some early rap, with an emphasis on the transition into electro. Includes Grandmaster Flash and the Furious Five *The Message* (1982) and excerpts from *The Tommy Boy Story*
Punk/Post-Punk	1977-1979	31	89.7	UK artists including Sex Pistols, UK Subs, Wire, The Cure, Gang of Four

Table 1. Overview of music corpus

part of the process of identifying influence. The latter might be justified as the more general case and is followed here: it is of particular import when scaling up to larger databases of audio where annotations are impractical for musicologists. Network techniques introduced in [5] are still valuable in providing applicable metrics for later analysis once similarity scores are established. However, this paper will look at direct first generation influence rather than longer-term networks spanning chains of multiple nodes.

The paper proceeds through section 2 detailing the set of 248 source audio files split over eight genre groups, and section 3 which discusses the technicalities of the predictive models used. Section 4 explores the separability of genre groups suggested, using machine learning algorithms, and the Anderson Darling statistical test to look for any rejection of the null hypothesis that they are drawn from the same distribution. In section 5 we apply the predictive models to examine questions of the strength of influence of precursor work on Detroit techno, Chicago house, and a late 80s pop control group. As well as working with models trained on whole groups of tracks, we also run tests for some famous individual tracks, such as Donna Summer's *I Feel Love* (1977). Finally, in section 6 we explore the implications of the experimental findings, and broach larger questions for studies of musical influence using MIR techniques.

2. SOURCES

Table 1 is an overview of the materials used in this study. Eight genre groups are presented, with 31 tracks per group. Five of these are precursor genres, movements in popular music from the mid 1960s to the early 1980s. The three top groupings are Chicago house, Detroit techno, and a control group of mid- to late-1980s pop including Madonna and Michael Jackson, coincident with an explosion in popularity of electronic dance music in the UK. The earlier genre groups include four important to the origins of electronic dance music: funk, disco, synth pop and hip-hop (particularly in its electro form). Some UK punk and post-punk records are included as a further control. [2]

Although the total duration of the genre groups differ, the critical thing is the equal number of examples in each, since tests are based on equal length excerpts from individual tracks, or otherwise involve a normalization for duration, such as the average log loss of a predictive model. 1980s examples of electronic dance music tend to involve longer tracks, where many 1960s and 1970s singles are much closer to the three minute (or less) pop song (short songs were also revived with punk's throwaway numbers); creating groups of an equal number of tracks all balanced in duration and the number of years associated is an unsolvable dynamic programming challenge.

There are many overlaps between these ostensibly separate groups, such as the shift to disco via Philly soul, the use of synthesizers by new wave acts as well as more explicitly by synth pop groups, or the appropriation of funk and disco backings in early rap records. [3] The a priori use of genre groups is justified on the grounds of the musicians' statements themselves, such as Derrick May cited above, who treat 'funk' and 'disco' as known areas of musical endeavour. The groups have been constructed from well known examples of the genres in question, and one confound in particular avoided in construction; synthesized electronic instrumentation in disco is not represented in the disco group, but a few examples from the Moroder camp are included under synth pop. Part of our analysis shall be to consider the well-definition of the groups, in terms of their internal consistency; as well as considering genre based influence, we shall also take a look at influential individual tracks later in the paper, to avoid any claims of resting too heavily on genre constructions.

No categorisation can be perfect, and there are some missing early 1980s genre groups, such as European electronic body music and industrial (e.g. Liaisons Dangereuses, Front 242) and italo-house (e.g. Klein and MBO, Alexander Robotnick), and mid 1980s New York production developments (freestyle, Mantronix etc.). Manageability of the overall study, and the greater overlap with the formation dates in Chicago and Detroit, made these categories out of the scope of the current investigation; however, again,

[2] Joy Division were specifically excluded, since their New Order manifestation intersects with electro circa 1983.

[3] The term 'house' itself floats around in 1983, for example on *Rock the House* by Pressure Drop, a 1983 release on Tommy Boy.

we return to a few individual tracks rather than whole genre groups below. Although there are some earlier prototype Chicago house tracks, such as Jesse Saunders' *On and On* (1984), we have avoided these for some separation in date from the precursor genres in this study; most commercially available international Trax releases, for example, tend to be available from 1986 at the earliest. [4]

Complete track lists can be made available on researcher request; all music was purchased.

3. PREDICTIVE MODELING

Bag of features assumptions [7] are avoided in favour of using time series modeling to construct predictive models; in particular, Prediction by Partial-Match (PPM) variable order Markov models [2, 13]. The strength of prediction of one piece or group of pieces by another is measured by average log loss in information theoretic terms [2], as further detailed below.

Various musical attributes of the pieces under consideration are modelled, such as timbral, rhythmic and harmonic change components. The final model combines three core elements:

1. A model τ of timbre based on 11 features, with feature vectors accumulated by beats, vector quantised by a k-Means classifier into symbols, and used to train a PPM model
2. A model ι of inter-onset intervals after onset detection on polyphonic audio, using a classifier for IOI sizes into symbols, and subsequent PPM
3. A model η of harmonic change, based on extracting beat-wise 12TET chroma, forming the sum of differences between beats, a classification into symbols, and PPM

For τ, a more general set of timbral descriptors was selected than MFCCs, to try to reflect the character of analyzed audio, without high dimensionality (which would impact on the k-Means step). The timbral features were perceptual loudness, sensory dissonance (using a Sethares model [16]), two transient detection measures using the wavelet method of [10], spectral centroid, spectral percentile at 0.8% and 0.95% energy, zero crossing rate, spectral crest measure, spectral slope, and a raw onset detection function (the preprocessed signal for an onset detector). The onset detector for the raw detection function, and for the isolation of onsets for IOI detection in ι, is based on work by Stowell [19], and is applied to polyphonic audio tracks; the rectified complex deviation (RCD) onset detection function used here has proven reasonable for such applications. All features were subject to normalization with respect to corpus derived minimum and maximum values, and were gathered in beats by averaging feature vectors. Chroma for η were also accumulated in beats, the difference between beatwise chroma vectors taken, and summed over the difference vector. This created a one-dimensional measure of harmonic change, where the summation process avoided issues with different absolute pitch centres in the music.

Feature vectors, IOIs and delta chroma sums were subject to vector quantisation into 20 tokens before PPM modeling. In order to symbolize the multidimensional timbral feature vectors in τ, vectors extracted from the training corpus were clustered with the unsupervised k-Means algorithm, with k=20. As one dimensional quantities, the IOIs in ι and harmonic change sums in η were classified into twenty bands by histogram equalisation [3, p.188]. A histogram for categorisation was constructed by sorting the values into order, splitting them by twenty equal size bands, and taking histogram bin positions by the maximum in each band. Twenty bands was a good compromise for a reasonable alphabet size for the PPM, without invoking too high a dimensionality. PPM models were then trained on the sequences in the 20 token alphabet, using consecutive subsequences of five values at a time. The particular model variant used here is what Pearce and Wiggins [13] denote the PPM-AX variant.

Scoring for a given PPM model γ on novel data set X was then calculated by

$$\text{averagelogloss}_{\gamma}(X) = \frac{-\sum_{x \in X} \log_2 P(x|\gamma)}{|X|} \qquad (1)$$

where the x are all the sequence contexts of the data to be tested [2] Minimal scores correspond to high probability sequences, that is, highly expected with respect to the model. The log is critical to avoid floating point underflow on multiplication of small probabilities. This average log loss measure is robust to different durations of sequences considered; in any case, we use equal length excerpts from pieces.

In applying this to a corpus, a predictive model is trained on a subset of pieces. The model can then be used to examine one or more target pieces, via equation 1. In this work, the models are applied to equal size groups of pieces, summing the model predictions within the group to get a total score for the predictability of that group with respect to the probabilistic model. [5] The final scores are actually the sum of those from the three models τ for timbre, ι for rhythm and η for harmonic change; these three components are individually normalized before the final sum. Whilst a given model's predictions are internally comparable, care must be taken in comparing the absolute value of scores between different models; the normalization reflects that only relative degree and order is comparable.

All implementations used the open source SuperCollider Music Information Retrieval library by the author, [6] which includes an example with the three component model presented here. Specific client source code for the work is available on researcher request.

[4] One example of a famous and influential track which was recorded earlier but released much later is Phuture's *Acid Tracks*, recorded late 1985, released 1987.

[5] For a common group size, we can divide by the group size without compromising comparability, rather than taking an average over a different number of contributing members.

[6] http://www.sussex.ac.uk/Users/nc81/code.html

	Probability of rejecting null hypothesis							
Model	Chicago	Detroit	Pop	Funk	Disco	Synth Pop	Electro	Punk
Chicago	0.4297	0.0033	0.3879	0.2131	0.0776	0.0832	0.3986	0.2532
Detroit	0.0033	0.0023	**0.0013**	**0.0005**	**0.0001**	**0.0004**	0.0074	**0.0003**
Pop	0.3879	**0.0013**	0.4402	0.1113	0.2218	0.0283	0.2739	0.2109
Funk	0.2131	**0.0005**	0.1113	0.3554	0.0040	0.1224	0.1629	0.2456
Disco	0.0776	**0.0001**	0.2218	0.0040	0.3057	**0.0003**	0.0625	0.0741
Synth Pop	0.0832	**0.0004**	0.0283	0.1224	**0.0003**	0.0982	0.0671	0.0474
Electro	0.3986	0.0074	0.2739	0.1629	0.0625	0.0671	0.3366	0.1992
Punk	0.2532	**0.0003**	0.2109	0.2456	0.0741	0.0474	0.1992	0.4389

Table 2. Application of Anderson-Darling tests within and between genre groups. Each cell entry is the probability of rejecting the null hypothesis that the tracks being tested together are a homogenous entity. Significant table entries are in bold, with respect to a Bonferroni significance level for multiple comparisons.

4. MACHINE LEARNING TESTS AND STATISTICAL TESTS OF SEPARABILITY FOR REPRESENTATIVE FEATURE VECTORS

An initial examination was carried out on the genre groups themselves, to see how "separable" the genre groups were from one another with respect to the ability of machine learning to differentiate them, and in terms of statistical tests for their internal and paired consistency. All tests were repeated twice, first for the timbral feature vector detailed in section 3 for model τ, and secondly for a vector of 11 MFCCs. Single vectors summarised single tracks; 45 seconds of feature vectors were extracted from a point 25% of the way into a given sound file, and averaged (normalization factors had already been calculated across the entire corpus of 248 tracks in an earlier sweep). ARFF files were exported for tests in Weka, and arrays of data into MATLAB for statistical tests.

For machine learning, we tested the discriminatory power of supervised classifiers to learn the training sets (given the genre labels 0-7), and of unsupervised clustering algorithms to match these labels (the 'classes to clusters' evaluation setting in Weka). Over 8 genres, the best scores came from the 11 different features rather than the MFCCs,[7] but were still of low classification success. The best results were for a k-Means clusterer (correctly classified 69 of 248 instances, 27.8%) and naive Bayes (68 of 248, 27.4%); a range of other algorithms were investigated including neural nets and J48 decision trees. The best MFCC results were for k-Means (correct 47, 18.95%), and naive Bayes (correct 53, 21.4%). Examination of 2-dimensional subsets of features revealed a lot of overlap between genres. This result motivated using a more sophisticated time series modeling approach rather than average feature vectors, and using the mixed feature vector for timbre rather than MFCCs. With just the house and techno groups, and the heterogenous feature vector, classification accuracy was around 50% (at chance given two groups), with best performance from k-Means (correctly classified 37 of 62, 60%) and naive Bayes (34 of 62, 54.8%). For the 11 MFCCs, nothing better than 33 out of 62 (53.2%) accuracy was observed, with most algorithms performing worse than chance.

Statistical tests were also applied to the model τ feature vector data, to look for overlap between genre groups, and internal consistency. An Anderson-Darling test was utilized [20], which tests the null hypothesis that all feature data arose from the same distribution (without assuming normality of that distribution); a significant p-value indicates that the data is from multiple distributions. Table 2 presents the symmetric matrix of values for all pairwise (62 tracks at a time) and within-group (31 tracks) tests. The Detroit techno group is seen as more heterogenous, and the null hypothesis would be rejected if the threshold was set at 0.05% p-value. Because there are 28 pairs and 8 individual genres = 36 tests, under the Bonferroni correction the p-value of $0.05/36 = 0.0013889$ has the statistical power to cover everything up to 0.05 and may give a better sense of whether the Detroit techno result is aberrant; we may keep the null hypothesis of Detroit techno as consistent at this level. To the extent that the probabilities point to degree of homogeneity, the Detroit corpus is more heterogenous. The make-up of the Detroit corpus unbalances things when paired with other groups, whilst there seems to be a strong overlap of Chicago house and late 80s pop (given UK No.1s by Chicago house producers, this may not be so unexpected) as well as with electro and funk. Nonetheless, the average feature vector approach is quite coarse, and the predictive models are now deployed for a finer-grained examination.

5. INVESTIGATION OF INFLUENCE THROUGH PREDICTIVE MODELS

In this section, results are reported for the predictive scores given to particular genre groups, and to individual tracks, from models constructed from first genre groups, and then some interesting precursor tracks. Section 3 describes the model construction and algorithm for prediction scores. Scores are normalized for a particular run of a particular model to fall from 0 to 1, where 0 would be totally predicted by the model at probability 1, and 1 is the least expected observed situation. Models can be constructed in two directions; we favour constructing a model from an earlier historical genre or track, to predict later tracks. There is an argument that construction in the opposite direction would also indicate the degree of derivation of the later work from earlier, and we report such constructions symmetrically for the genre groups. We do not form the fully symmetric score from a matrix plus its

[7] Vectors of 40 MFCCs were also tested without any improvement in classification scores.

	Prediction Score							
Model	Chicago	Detroit	Pop	Funk	Disco	Synth Pop	Electro	Punk
Chicago	**0.04013**	0.81935	0.83104	0.81382	0.82633	**0.7646**	0.87111	0.80286
Detroit	0.7951	**0.04889**	0.8146	0.77913	0.79449	0.76644	0.861	**0.75883**
Pop	0.8961	0.86525	**0.04548**	0.8945	**0.83403**	0.85331	0.90055	0.89854
Funk	0.85148	0.85413	0.90196	**0.02236**	0.91059	**0.84123**	0.89624	0.91292
Disco	0.83158	0.76417	0.82118	0.8417	**0.03575**	**0.76346**	0.89092	0.813
Synth Pop	0.8846	0.88163	0.88508	0.87967	**0.8056**	**0.07829**	0.88813	0.86481
Electro	0.85575	0.85762	0.89833	0.87927	0.91642	**0.82893**	**0.02486**	0.87419
Punk	0.76547	**0.70549**	0.89158	0.91145	0.86657	0.81012	0.86613	**0.03303**

Table 3. Prediction scores of genres from models constructed for each genre. Starred italics on the diagonal correspond to the prediction of the data used to construct a model by that model; one bold entry in each row indicates the closest other genre to the model genre.

transpose, since model construction itself is not guaranteed to produce scores in exactly the same ranges, and the post calculation normalization reported here, whilst helpful for seeing links, is not uniform in comparison (even unnormalized, differences in model construction would question comparability; results are relative to a given model).

Table 3 presents the asymmetric matrix of results over the predictive models. The diagonal is italicized; all models find their own source database most highly predicted, as we'd expect for any sensible probabilistic modeling. The description of Chicago house as disco with a drum machine, and Detroit techno as future electro funk, is only partially borne out in these figures. One aberrant factor is the close link of Detroit techno with late 1970s punk and post punk guitar tracks; Chicago house tracks are also seen as closer to punk than Detroit techno on this view. Of the three factors in the scores, the onset detection driven IOI model is the point of similarity here; a related density of events has an impact, as does timbre to an extent, perhaps through some degree of sonic exploration in house and techno instrumental tracks. Examining relative values within rows, the links to synth pop are clear; Detroit is closer to funk than disco, but Chicago also that way round. From the funk model, Chicago is very marginally ahead of Detroit, if within the third decimal place. Electro is closest to synth pop, which is musicologically sensible; the synth pop model finds Chicago, Detroit and late 80s pop of a muchness in terms of potential influence.

The relative degree of influence of a seminal piece can be investigated by creating a predictive model from it. Table 4 compares 22 interesting tracks from the 1970s to the early 1980s; these precursor tracks were selected from mentions in sources on EDM history such as [11,18]. Complete individual tracks are used to form predictive models, which are then deployed to predict the Chicago house, Detroit techno and late 1980s pop corpuses.

Given these mainly synthesizer-flavored precedents, Detroit makes the most whole-hearted embrace of the technologized future, and shows the greater link to James Brown to boot (though not Parliament, which links more to pop, perhaps through the inclusion of Prince in that corpus in particular). The sanity checks show some consistency, with two versions of *Planet Rock* both leading to similar results, and two runs on the same Kano track also coming out with a similar ordering. The tests were repeated over the whole set of songs, using a version of the predictive model with 10 states rather than 20 per vector quantiser, without any great divergence from the results presented here, excepting *Mothership Connection*, *Numbers* and *Clear* being assigned to techno, *Magic Fly* to Chicago, and *Problèmes D'Amour* to pop. The greater vocal content in Kraftwerk's *The Model* may be an explanation of its stronger link to certain elements of the Chicago house corpus, or the link of the female vocal of *I Feel Love* through to pop.

Individual tracks across the corpus of 93 can be examined, to find the most predicted and the most divergent from a model. For instance, for Kano's *It's A War*, the three closest were Derrick May's *Spaced Out* and *Nude Photo*, and Blake Baxter's *Ride Em Boy*, all three from the Detroit corpus (as we might hope for this track's reception history, though there are also aural links to Prince), and the furthest away, in pop, Madonna's *Live to Tell* and the Bangles' *Eternal Flame* and *Hazy Shade of Winter*.

6. DISCUSSION

Musical influence is a complex mechanism; the assumption that strength of prediction is related to degree of influence seems reasonable, but may hide other factors, such as indirect influence, common equipment and teaching tools (such as music technology magazines), social currents, and even independent co-creation of the same idea.

The audio content analysis used here cannot be claimed to be on a par with the musicologist's ear. On the other hand, computer tools can point to useful currents of inquiry, and provide an alternative stimulus to musical historical and analytical investigation. Furthermore, it is really worth exploring the musicological applications at an early stage, to clarify the potential impact of such tools, and feedback their effectiveness.

The genre groups used in this study make categorical assumptions which can hide musical continuity. Whilst their construction was to answer some questions of influence and overlap, the most interesting results relate more to the scope of individual tracks. Future work may drop genre assumptions entirely, creating a predictive model from every individual track, to assess every other; given pairwise similarity measures and chronological distance, multidimensional scaling may give insight into structure. It may also be productive to consider rates of change per year, by

Model	Chicago	Detroit	Pop
Giorgio Moroder *From Here To Eternity* (1977)	0.6551	**0.5998**	0.6562
Donna Summer *I Feel Love* (1977)	0.6461	0.6824	**0.6346**
Kraftwerk *The Model* (1978)	**0.6659**	0.6929	0.7564
Kraftwerk *Numbers* (1981)	0.6146	0.5761	**0.5723**
Kraftwerk *Trans-Europe Express* (1977)	0.5419	**0.4784**	0.5648
Cerrone *Supernature* (1977)	0.9124	**0.8361**	0.8573
Dee D. Jackson *Automatic Lover* (1978)	0.6994	**0.6878**	0.7077
Space *Magic Fly* (1977)	0.6855	0.6993	**0.6753**
Sylvester *You Make Me Feel (Mighty Real)* (1978)	0.8448	**0.7551**	0.8348
Lipps Inc. *Funkytown* (1979)	0.7209	**0.6336**	0.6734
Kano *It's A War* (1980)	0.5009	**0.4494**	0.5877
Kano *It's A War* (1980)	0.5974	**0.5235**	0.6465
Soft Cell *Tainted Love* (1981)	0.8372	**0.8085**	0.8496
Depeche Mode *Get The Balance Right* (1983)	0.7934	**0.727**	0.7852
Afrika Bambaataa et al. *Planet Rock (12" Vocal Version)* (1982)	0.8073	**0.7501**	0.7697
Afrika Bambaataa et al. *Planet Rock* (1982)	0.7959	**0.7562**	0.7701
James Brown *Funky Drummer Pts. 1 and 2* (1970)	0.6383	**0.5262**	0.6318
Parliament *Mothership Connection (Star Child)* (1975)	0.8166	0.798	**0.7881**
Alexander Robotnick *Problèmes D'Amour* (1983)	0.6395	**0.6128**	0.6496
Cybotron *Enter* (1983)	0.707	**0.6833**	0.6852
Cybotron *Clear* (1983)	**0.5503**	0.5552	0.5601
Cybotron *Cosmic Cars* (1983)	0.65	**0.6387**	0.657

Table 4. Prediction of genres using models constructed from individual tracks. The closest genre from each model is indicated in bold.

constructing a model on one year (or on other windows of time) and testing how predictable the next is.

A human study of similarity on this corpus would be a useful follow-up, though one confound is the factor of recognition; expert musicologists of EDM would recognise many of the Chicago and Detroit tracks immediately, and the corpus used here involves many famous works. Its historical importance, however, makes it a very interesting corpus to work on, of great musicological relevance.

In future work we may consider extending out to a larger-scale investigation of the history of electronic music. Alternative time series models, such as Hidden Markov Models, could be employed, avoiding vector quantisation simplifications, and possibly using symmetrised distance measures such as the cross-likelihood discussed in [21]. More sophisticated statistical models of causality may also help to stretch the machinery for modeling influence [14].

7. CONCLUSIONS

This paper presented a study of applying MIR techniques to probe the borderline of Chicago house and Detroit techno. More generally, we related later 1980s works to 1960s to early 1980s precursors through a number of methods. We saw that the house and techno genres overlap, and are not necessarily tightly defined. Nonetheless, there was some corroboration of Derrick May's characterisation of Detroit techno as futuristic in its sound world, though less support for its separate funkiness; the disco and synth pop heritage is a strong link to the two styles. The study presents a template for future work over the same corpus, as refined sound analysis models become available, and for more general future audio-content driven examination of musical influence.

8. REFERENCES

[1] J. J. Aucouturier and F. Pachet. Representing musical genre: A state of the art. *Journal of New Music Research*, 32(1):83–93, 2003.

[2] R. Begleiter, R. El-Yaniv, and G. Yona. On prediction using variable order Markov models. *Journal of Artificial Intelligence Research*, 22:385–421, 2004.

[3] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. OReilly Media, Sebastopol, CA, 2008.

[4] Bill Brewster and Frank Broughton. *Last Night a DJ Saved My Life*. Headline Book Publishing, London, 2006.

[5] Nicholas J. Bryan and Ge Wang. Musical influence network analysis and rank of sample-based music. In *Proceedings of the International Symposium on Music Information Retrieval*, Miami, October 2011.

[6] Mark J. Butler. *Unlocking the Groove*. Indiana University Press, Indiana, 2006.

[7] Michael Casey, Remco Veltkamp, Masataka Goto, Marc Leman, Christophe Rhodes, and Malcolm Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668–696, April 2008.

[8] Nick Collins. Computational analysis of musical influence: A musicological case study using MIR tools. In *Proceedings of the International Symposium on Music Information Retrieval*, Utrecht, August 2010.

[9] Stuart Cosgrove. Techno! the new dance sound of Detroit (liner notes), 1988.

[10] Laurent Daudet. Transients modelling by pruned wavelet trees. In *Proceedings of the International Computer Music Conference (ICMC)*, Havana, Cuba, 2001.

[11] Peter Kirn, editor. *Keyboard Presents the Evolution of Electronic Dance Music*. Backbeat Books, Montclair, NJ, 2011.

[12] Kembrew McLeod. Genres, subgenres, sub-subgenres and more: Musical and social differentiation within electronic/dance music communities. *Journal of Popular Music Studies*, 13:59–75, 2001.

[13] Marcus Pearce and Geraint Wiggins. Improved methods for statistical modelling of monophonic music. *Journal of New Music Research*, 33(4):367–385, 2004.

[14] Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3:96–146, 2009.

[15] Simon Reynolds. *Generation Ecstasy: Into the World of Techno and Rave Culture*. Routledge, New York, 1999.

[16] William A. Sethares. *Tuning Timbre Spectrum Scale (2nd Edition)*. Springer Verlag, Berlin, Germany, 2005.

[17] Peter Shapiro. *Turn the Beat Around: The Secret History of Disco*. Faber and Faber Limited, London, 2005.

[18] Dan Sicko. *Techno Rebels: The Renegades of Electronic Funk (2nd Ed.)*. Wayne State University Press, Detroit, MI, 2010.

[19] D. Stowell and Plumbley. M. D. Adaptive whitening for improved real-time audio onset detection. In *Proceedings of the International Computer Music Conference (ICMC)*, Copenhagen, 2007.

[20] A. Trujillo-Ortiz, R. Hernandez-Walls, K. Barba-Rojo, L. Cupul-Magana, and R.C. Zavala-Garcia. Andarksamtest:anderson-darling k-sample procedure to test the hypothesis that the populations of the drawn groups are identical., 2007.

[21] Tuomas Virtanen and Marko Helén. Probabilistic model based similarity measures for audio query-by-example. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New York, 2007.

ASSOCIATION MINING OF FOLK MUSIC GENRES AND TOPONYMS

Kerstin Neubarth[1,2] **Izaro Goienetxea**[3] **Colin G. Johnson**[2] **Darrell Conklin**[3,4]

[1]Canterbury Christ Church University, Canterbury, United Kingdom
[2]School of Computing, University of Kent, Canterbury, United Kingdom
[3]Department of Computer Science and Artificial Intelligence,
University of the Basque Country UPV/EHU, San Sebastián, Spain
[4]IKERBASQUE, Basque Foundation for Science, Bilbao, Spain

ABSTRACT

This paper demonstrates how association rule mining can be applied to discover relations between two ontologies of folk music: a genre and a region ontology. Genre–region associations have been widely studied in folk music research but have been neglected in music information retrieval. We present a method of association rule mining with constraints consisting of rule templates and rule evaluation measures to identify different, musicologically motivated, categories of genre–region associations. The method is applied to a corpus of 1902 Basque folk tunes, and several interesting rules and rule sets are discovered.

1. INTRODUCTION

In recent years music information retrieval (MIR) research has increasingly turned towards folk and ethnic music and its contexts [6, 19], and perspectives for collaboration between MIR and ethnomusicology have been outlined [22, 23]. While musicologists consider interactions between folk music genres, their geographical distribution and musical characteristics [12, 18], MIR research has mainly focused on geographically organised folk music corpora [10, 11, 21]. A recent study on Cretan folk music extracts distinctive melodic interval patterns for genre and for area classes, but does not link genres and areas, although the idea of conjunctive genre–area classes is mentioned [5]. In this paper we analyse relations between genres and regions in a collection of Basque folk music, through association rule mining.

Association rule mining has been developed extensively in the wider field of knowledge discovery and data mining, but has seen only limited attention in MIR (e.g. [4]). Work on cultural heritage, not restricted to but also covering music, has used association rule mining to populate a heritage ontology with new relations between concepts: annotations of heritage objects were mined and discovered associations proposed to a domain expert, who categorised them as subclass or associative relations [13]. Our research goes beyond these studies by applying association rule mining to suggest different, labelled, categories of associations.

2. TASK DESCRIPTION

The folk music collection Cancionero Vasco was originally compiled by the musicologist Padre Donostia for a musical heritage competition in 1912 [18, article 'Basque Music']. It has been digitised and curated by Fundación Euskomedia in collaboration with musicologists at Fundación Eresbil, under the auspices of the Basque Studies Society.[1] The digitised collection contains 1902 Basque folk songs and dances. The examples are annotated with genre information and the location (toponym) where they were collected. The annotation vocabulary is defined in two ontologies: a geographical ontology of provinces, municipalities and towns or villages, and an ontology of hierarchically organised genres. The aim of applying association rule mining to this collection is to discover patterns of genres and regions co-occurring in the annotations, which suggest that certain genres are particularly associated with certain regions and vice versa.

A common challenge in applications of association rule mining is to manage potentially large numbers of discovered association rules and identify those rules which are interesting to a user [14]. Genre–region relations feature prominently in folk music research, and the musicological interest goes beyond simple mappings: In a qualitative analysis we examined surveys of traditional music in 25 European countries [18], extracted statements linking genres and regions, grouped them according to similar meaning and for each group suggest a shared interpretation to facilitate translation into association rules (Table 1). As these association categories are based on recurring observations in musicological reference articles, they are considered relevant to users of folk music collections. The data mining task then is not only to discover associations between the two ontologies of genres and regions, but to distinguish associations of the categories listed in Table 1.

3. DATA AND METHODS

This section describes the data representation and the data mining method for discovering genre–region associations in the Cancionero Vasco. In order to identify association

[1] http://www.euskomedia.org/cancionero

Category	Example observation	Interpretation
Present	"survivals of calendar ritual and wedding music are found in the Opole area"	genre present in region
Absent	"the [two-part] form does not exist at all in Kosovo and Metohija"	genre not present in region
Local	"localized dances include the corridinho (Algarve)"	genre present in exactly one region
Mainly	"krakowiak dances [...] are found mainly in Małopolska"	genre over-represented in region with respect to other regions
Dominant	"the polka is the most popular dance in these regions"	genre over-represented in region with respect to other genres
Typical	"the dance-song seguidillas is typical of New Castile"	genre over-represented in region with respect to other regions and genres
Hardly	"the klarino style is hardly found at all on the islands"	genre under-represented in region with respect to other regions
Rare	"Genres [...] such as the epic are quite rare in central European repertories, whereas genres [...] such as the ballad are quite common."	genre under-represented in region with respect to other genres

Table 1. Categories of genre–region associations, with example quotations from the New Grove [18].

rules of the different categories summarised in Table 1, these categories are translated into association constraints consisting of rule templates and rule evaluation measures.

3.1 Ontologies

The annotations mined in this study are standardised and structured in two ontologies: a genre and a geographical ontology, formalised in description logic (DL) [3, 9].

The ontology of folk music genres consists of two parts: a set of statements $G \sqsubseteq G'$ defines the genres and their subsumption relations, e.g.

$$\mathsf{work\ songs} \sqsubseteq \mathsf{life\text{-}cycle\ songs}$$

$$\mathsf{life\text{-}cycle\ songs} \sqsubseteq \mathsf{genre}$$

state that work songs are subsumed by, i.e. more specific than, life-cycle songs and that life-cycle songs are a genre. The second part of the ontology is a set of assertions $G(e)$ where G is a genre concept and e a folk tune example: this part formalises the genre annotations of the examples in the Cancionero Vasco, using 31 genres. Examples are asserted with their most specific known genre annotation, which is not necessarily the lowest level of the subsumption hierarchy. The genre counts used in the association rule mining (Section 3.2) can be derived from the ontology by querying for the examples instantiating a genre; here the inference capabilities of DL allow to count examples not only for the directly asserted genre but also for more general, subsuming genres. Out of the 1902 examples in the corpus 341 examples are without a genre annotation.

The geographical ontology covers Euskal Herria, the Basque speaking areas in North-East Spain and South-West France. The ontology is organised into the three levels of provinces (7 toponyms), municipalities (681 toponyms) and towns or villages (2280 toponyms). Locations are formalised as instances in DL, assigned to levels, e.g.

$$\mathsf{province}(\mathsf{Lapurdi}) \quad \text{and} \quad \mathsf{municipality}(\mathsf{Azkaine}).$$

In DL the geographical relationships are defined in terms of spatial containment roles, e.g.

$$\mathsf{contains}(\mathsf{Lapurdi}, \mathsf{Azkaine})$$

asserts that Lapurdi (a province) contains Azkaine (a municipality). Each folk tune example e is asserted with the region R in which it was collected: $\mathsf{collected}(R, e)$. As with the genre ontology, assertions can be made at any level of the hierarchy and higher-level counts are inferred based on the transitivity of the containment relation. Out of the 1902 folk tunes in the Cancionero Vasco 272 tunes are without a toponym annotation.

The original annotation terms for genres are Spanish or Basque. For the presentation in this paper we give English translations for genres. As toponyms we use the Basque, rather than Spanish or French, names.

3.2 Association Rule Mining

Association rules are rules of the form $a \rightarrow b$ with an antecedent item set a and a consequent item set b ($a \cap b = \emptyset$) [20]. A rule $a \rightarrow b$ with confidence c states that $c\%$ of the data records containing items a also contain items b. Rule templates [14] define the form of a rule and specify which items can occur in the antecedent and consequent. In this study we mine for rules with one item in the antecedent and one item in the consequent. Here an item can be a genre (denoted G in the rule templates), a region (denoted R) or the complement of a genre or region (denoted $\overline{G}$ and $\overline{R}$ respectively).

	a	$\overline{a}$	
b	n_{ab}	$n_b - n_{ab}$	n_b
$\overline{b}$	$n_{a\overline{b}} = n_a - n_{ab}$	$n_{\overline{b}} - n_{a\overline{b}}$	$n_{\overline{b}} = n - n_b$
	n_a	$n_{\overline{a}} = n - n_a$	n

Figure 1. Contingency table for a rule $a \rightarrow b$.

The rule templates determine how the genre and region of a candidate association are mapped onto a contingency table from which a rule evaluation measure can be calculated [15]. Figure 1 presents a 2 × 2 contingency table for an association rule $a \rightarrow b$, where a is the antecedent item, b the consequent item, n_a the number of folk tunes annotated with a, n_b the number of folk tunes annotated with b, n_{ab} the number of folk tunes annotated with both a and b, and n the total number of folk tunes in the corpus. The notation $\overline{a}$ denotes the complement of item a. For example, with the rule template $G \rightarrow R$, n_a refers to the genre count and n_b refers to the region count; n_{ab} is the number of tunes instantiating both the genre and the region.

The evaluation measures most commonly used in association rule mining are support (frequency of the co-occurrence) and confidence (conditional probability of the co-occurrence given the antecedent). These two measures, however, are not sufficient to distinguish all association categories of Table 1, e.g. *Typical* against *Mainly* and *Dominant*. We thus considered further existing measures and their properties (e.g. [8, 16]). Measures for each category were selected in two steps. First, we defined the requirements for each category, given its interpretation (Table 1), based on established measure properties:

- Asymmetric vs symmetric measures: For all categories except *Present* and *Typical* the measure should be asymmetric, i.e. distinguish between $a \rightarrow b$ and $b \rightarrow a$.

- Increasing function with number of examples: The measures for *Mainly*, *Dominant*, *Hardly* and *Rare* should increase with n_{ab} for fixed n_a, while the measure for *Typical* should increase with n_{ab} for both n_a and n_b fixed.

- Decreasing function with number of counter-examples: The measures for *Mainly*, *Dominant*, *Hardly* and *Rare* should decrease with increasing $n_{a\overline{b}}$, and thus n_a, for fixed n_{ab}, while the measure for *Typical* should decrease with both increasing $n_{a\overline{b}}$ and $n_{\overline{a}b}$, and thus both n_a and n_b, for fixed n_{ab}.

- Sensitivity vs. insensitivity to sample size: As the measures are used to capture the relationship between a and b, they should be insensitive to changes in $n_{\overline{ab}}$ and thus to changes in n for fixed n_{ab}, n_a and n_b.

Second, we determined measures that match the requirements. Where more than one measure meets the category criteria, a measure is preferred in which variations of the measure value can easily be related to values in the contingency table [16]. Table 2 lists the resulting constraints for each association category.

Category	Template	Measure
Present	$G - R$	support
Absent	$G \rightarrow \overline{R}$	confidence ($c = 1$)
Local	$G \rightarrow R$	confidence ($c = 1$)
Mainly	$G \rightarrow R$	confidence
Dominant	$R \rightarrow G$	confidence
Typical	$G - R$	Jaccard
Hardly	$G \rightarrow \overline{R}$	Sebag-Schoenauer
Rare	$R \rightarrow \overline{G}$	Sebag-Schoenauer

Table 2. Constraints for the association categories.

Measure	Definition
support	$s = n_{ab}$
confidence	$c = n_{ab}/n_a$
Jaccard	$J = n_{ab}/(n_a + n_b - n_{ab})$
Sebag-Schoenauer	$S = n_{ab}/n_{a\overline{b}}$

Table 3. Definitions of the rule evaluation measures.

The definitions of the measures are given in Table 3. To ensure invariance with changes in n, absolute rather than relative support is applied for *Present*. For the categories describing under-representation (*Hardly* and *Rare*), the measure of Sebag-Schoenauer was found to discriminate better than confidence: confidence accepts most infrequent pairs, while Sebag-Schoenauer accepts pairs that are less frequent than comparison pairs.

During the mining, each candidate pair of a genre and a region is evaluated against the category constraints, i.e. the genre, region and pair counts are mapped onto n_a, n_b and n_{ab} according to the template, and the measure value is calculated. Pairs are tested for all categories, and associations can be assigned to more than one category (e.g. *Present* and *Mainly*).

4. RESULTS

Table 4 lists selected highly ranked rules for all categories. The p-values in Table 4 are calculated according to Fisher's exact test with left tail for *Absent*, *Hardly* and *Rare* and right tail for all other categories [7]. The p-value measures the probability of finding at most (left tail) or at least (right tail) the number of co-occurrences given by the pair count, under the conditions of the genre and region counts. To account for multiple comparisons, i.e. testing multiple hypotheses on the same data, results are marked for both significance level α and the Bonferroni-corrected significance level β; it should be noted, though, that the Bonferroni correction is highly conservative. Rules above the significance level are not necessarily rejected as uninteresting (see also [16]), rather the p-values provide additional information for interpreting discovered rules, with respect to the distribution of genres and regions in the total corpus.

Genre (count)	Region (count)	Pair count	Template	Measure	p-value
Present					
life-cycle songs (477)	Nafarroa (897)	259	$G - R$	$s = 259$	0.00089^{++}
Artaxuriketak (38)	Nafarroa (897)	30	$G - R$	$s = 30$	4.3e-05**
Absent					
dances (495)	Hazparne (23)	0	$G \rightarrow \overline{R}$	$c = 1$	0.00093^{++}
sacred songs (301)	Araba (27)	0	$G \rightarrow \overline{R}$	$c = 1$	0.00922^{*}
Local					
smugglers' songs (1)	Lapurdi (383)	1	$G \rightarrow R$	$c = 1$	0.02321^{+}
smugglers' songs (1)	Azkaine (61)	1	$G \rightarrow R$	$c = 1$	0.03207^{+}
Mainly					
Artaxuriketak (38)	Nafarroa (897)	30	$G \rightarrow R$	$c = 0.79$	4.3e-05**
moral songs (11)	Nafarroa (897)	8	$G \rightarrow R$	$c = 0.73$	0.04470^{+}
Dominant					
dances (495)	Araba (27)	24	$R \rightarrow G$	$c = 0.89$	7.8e-12**
dances (495)	Eugi (15)	13	$R \rightarrow G$	$c = 0.87$	1.4e-06**
Typical					
Carlism songs (1)	Biriatu (3)	1	$G - R$	$J = 0.33$	0.00158^{*}
Hardly					
dances (495)	Atharratze (23)	1	$G \rightarrow \overline{R}$	$S = 494$	0.00857
life-cycle songs (477)	Bizkaia (21)	2	$G \rightarrow \overline{R}$	$S = 237.5$	0.07232
Rare					
Artaxuriketak (38)	Lapurdi (383)	1	$R \rightarrow \overline{G}$	$S = 382$	0.01112^{+}
moral songs (11)	Nafarroa (897)	8	$R \rightarrow \overline{G}$	$S = 111.13$	0.98935

* significance level $\alpha = 0.01$, ** with Bonferroni correction $\beta = 0.0002$
$^{+}$ significance level $\alpha = 0.05$, $^{++}$ with Bonferroni correction $\beta = 0.001$

Table 4. Examples of discovered association rules.

In all cases of *Local*, genres are represented by only one example in the corpus and thus are necessarily identified as local. The occurrence of the smugglers' song in Lapurdi could be linked to the site of Lapurdi, stretching from the coast inland across the Pyrenees between Spain and France: in fact, the example in the Cancionero Vasco was collected more specifically in the municipality of Azkaine, in the Larrun area known to have been used by smugglers; in nearby Sara the "smugglers' race", celebrated in August, has become part of 20th-century folklore [17].

5. DISCUSSION

The interest of this study lies in discovering genre–region associations in the Cancionero Vasco which are potentially interesting to users who browse or analyse the collection. The association categories defined in this paper provide additional semantics for rules as compared to traditional association rule mining, and can help organise and understand the folk music corpus. Multiple labels can further specify an association (Example 1) and even capture different aspects of the same genre–region pair (Example 2):

Example 1: Not only are the corn-harvesting songs Artaxuriketak present in Nafarroa, the Artaxuriketak examples in the Cancionero Vasco are mainly from Nafarroa.

Example 2: Within the Cancionero Vasco moral songs are rare in Nafarroa, i.e. under-represented with respect to other genres in Nafarroa (8 out of 897 instances), but are over-represented in Nafarroa with respect to their occurrence in other regions, i.e. they occur mainly in Nafarroa (8 out of 11 instances).

The p-value is a symmetric measure, i.e. it does not distinguish between e.g. rule $G \rightarrow R$ and rule $R \rightarrow G$. Using different rule templates with confidence or Sebag-Schoenauer as evaluation measure, on the other hand, al-

Region (count)	Pair count	Category	Measure	p-value
Araba (27)	0	*Absent*	$c = 1$	0.57771
Bizkaia (21)	0	*Absent*	$c = 1$	0.65307
Gipuzkoa (175)	4	*Present*	$s = 4$	0.46864
Lapurdi (383)	1	*Present*	$s = 1$	0.9964
		Rare	$S = 382$	0.01112
Nafarroa-Beherea (47)	1	*Present*	$s = 1$	0.61722
Nafarroa (897)	30	*Present*	$s = 30$	4.3e-05
		Mainly	$c = 0.79$	4.3e-05
Zuberoa (80)	2	*Present*	$s = 2$	0.48504

Table 5. Group of rules for the genre Artaxuriketak (genre count 38).

lows one to identify the asymmetric cases of over-representation (*Mainly* vs. *Dominant*) or under-representation (*Hardly* vs. *Rare*), which correspond to different observations in folk music surveys.

Example 3: Artaxuriketak occur mainly in Nafarroa as compared to other regions (79% of the Artaxuriketak instances in the Cancionero Vasco are from Nafarroa). On the other hand, dances are dominant in Araba with respect to other genres in the same region (89% of examples from Araba are dances).

The association categories can facilitate the analysis of groups of discovered rules. Folk music collections are often organised according to genres or regions [12], and folk music surveys may review genres against an underlying geographical classification [18]. For example, a summary of folk-dance in Finland states: "Although most polska melodies have been collected in Pohjanmaa, the dance was known throughout the country except in the far north and Karelia." [18]

Example 4: The geographical distribution of Artaxuriketak at the level of provinces can be described in a similar way (Table 5): While most Artaxuriketak in the corpus were collected in Nafarroa, the genre is also known in the other provinces except – within the Cancionero Vasco – in Araba and Bizkaia. Given the large number of instances for Nafarroa (897 instances, 47% of the corpus) it is not surprising that most Artaxuriketak were collected in Nafarroa, but the high proportion (nearly 80% of Artaxuriketak) is statistically significant. It is interesting to note that traditionally farmers in Nafarroa have cultivated corn (maize) while the chief crops in Araba have been cereals as well as fruit, wine and olives [2]. For Lapurdi, surveys in the 1980s reported that nearly half of the area was dedicated to wine (45%), followed by other crops (40%), woods (10%) and urban or uncultivated areas (5%) [1].

Other folk music surveys follow a mainly geographical organisation. For example, the New Grove article on traditional music in Croatia is structured according to regions; the following statement is taken from the section on Western Croatia: "This region is characterised by the tanac and balun dances. [...] Other dances, like the polka and the valcer, are also performed." [18]

Example 5: Folk music in the province of Araba, according to the Cancionero Vasco, is dominated by dances (89% of the Araba examples are dances, p-value $= 7.8e$-12). Related rules, not shown in Table 4, indicate that of the other genres only life-cycle songs were also collected in Araba (pair count 3, *Hardly*, $S = 158$, p-value $= 0.06411$), and more particularly within life-cycle songs the subgroup of work songs (*Present*, $s = 3$, p-value $= 0.02580$).

6. CONCLUSIONS

In this paper we have shown how association rule mining can be used to discover relevant relations between genres and geographical locations of folk tunes, more specifically how association rule mining with rule templates and evaluation measures can be used to identify different, musicologically motivated, categories of genre–region associations. The method was applied to a collection of Basque folk music, and example associations discussed in the context of folk music research.

This research represents an original contribution to MIR both in terms of retrieval task (discovering associations between folk music genres and their geographical distribution) and method (systematically combining rule templates and evaluation measures to distinguish different association categories). As a case study of interdisciplinary collaboration between MIR and musicology it demonstrates how musicology can inform the task definition, method design and discussion of results; the examples illustrate how MIR can both support musicological observations and stimulate further analysis.

Our work can be extended in several ways. The mining results can support more traditional information retrieval of music, like browsing and searching of music collections guided by association categories or by groups of association rules. Classification using association rules could be explored to suggest genre and toponym annotations for unlabelled tunes. The method could also be adapted to search for associations between annotations and music content classes. In addition the approach could be applied to data in other MIR areas such as user tagging, folksonomies and music recommendation.

7. ACKNOWLEDGEMENTS

We thank Fundación Euskomedia and Fundación Eresbil for providing the Cancionero Vasco for study. This research was partly supported by a grant *Análisis Computacional de la Música Folclórica Vasca* (2011-2012) from the Diputación Foral de Gipuzkoa, Spain.

8. REFERENCES

[1] Enciclopedia Auñamendi. Fondo Bernardo Estorés Lasa. http://www.euskomedia.org/aunamendi.

[2] Encyclopaedia Britannica Online. academic edition, 2012. http://original.search.eb.com/.

[3] F. Baader, D. Calvanese, D. McGuiness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge, 2003.

[4] D. Conklin. Melodic analysis with segment classes. *Machine Learning*, 65(2-3):349–360, 2006.

[5] D. Conklin and C. Anagnostopoulou. Comparative pattern analysis of Cretan folk songs. *Journal of New Music Research*, 40(2):119–125, 2011.

[6] 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.

[7] 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.

[8] L. Geng and H. J. Hamilton. Interestingness measures for data mining: a survey. *ACM Computing Surveys*, 38(3):1–32, 2006.

[9] I. Goienetxea, J. Arrieta, I. Bagüés, A. Cuesta, P. Leiñena, and D. Conklin. Ontologies for representation of folk song metadata. Technical Report EHU-KZAA-TR-2012-01, Department of Computer Science and Artificial Intelligence, University of the Basque Country UPV/EHU, 2012. http://hdl.handle.net/10810/8053.

[10] R. Hillewaere, B. Manderick, and D. Conklin. Global feature versus event models for folk song classification. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 729–733, Kobe, Japan, 2009.

[11] Z. Juhász. A systematic comparison of different European folk music traditions using self-organizing maps. *Journal of New Music Research*, 35(2):95–112, 2006.

[12] I. J. Katz. The traditional folk music of Spain: explorations and perspectives. *Yearbook of the International Folk Music Council*, 6:64–85, 1974.

[13] T. Kauppinen, H. Kuittinen, K. Seppäla, J. Tuominen, and E. Hyvönen. Extending an ontology by analyzing annotation co-occurences in a semantic cultural heritage portal. In *Proceedings of the ASWC 2008 Workshop on Collective Intelligence, 3rd Asian Semantic Web Conference (ASCW 2008)*, pages 1–6, Bangkok, Thailand, 2008.

[14] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. In *Proceedings of the 3rd International Conference on Information and Knowledge Management*, pages 401–407, Gaithersburg, Maryland, 1994.

[15] N. Lavrač, P. Flach, and B. Zupan. Rule evaluation measures: a unifying view. In *Proceedings of the 9th International Workshop on Inductive Logic Programming (ILP-99)*, pages 174–185, Bled, Slovenia, 1999.

[16] P. Lenca, P. Meyer, B. Vaillant, and S. Lallich. On selecting interestingness measures for association rules: user oriented description and multiple criteria decision aid. *European Journal for Operational Research*, 184(2):610–626, 2008.

[17] J. A. Perales Díaz. Fronteras y contrabando en el Pirineo Occidental. *Zainak. Cuadernos de Antropología-Etnografía*, 17:127–136, 1998.

[18] S. Sadie, editor. *New Grove Dictionary of Music and Musicians*. Macmillan, London, 2001.

[19] X. Serra. A multicultural approach in music information research. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, pages 151–156, Miami, Florida, USA, 2011.

[20] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD-97)*, pages 67–73, Newport Beach, CA, USA, 1997.

[21] J. Taminau, R. Hillewaere, S. Meganck, D. Conklin, A. Nowé, and B. Manderick. Descriptive subgroup mining of folk music. In *2nd International Workshop on Machine Learning and Music (MML 2009)*, Bled, Slovenia, 2009.

[22] G. Tzanetakis, A. Kapur, W. Schloss, and M. Wright. Computational ethnomusicology. *Journal of Interdisciplinary Music Studies*, 1(2):1–24, 2007.

[23] 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.

CHARACTERIZATION OF EMBELLISHMENTS IN NEY PERFORMANCES OF MAKAM MUSIC IN TURKEY

Tan Hakan Özaslan
Artificial Intelligence
Research Institute - CSIC
Bellaterra, Spain.
tan@iiia.csic.es

Xavier Serra
Music Technology Group
Universitat Pompeu Fabra
Barcelona, Spain
xavier.serra@upf.edu

Josep Lluis Arcos
Artificial Intelligence
Research Institute - CSIC
Bellaterra, Spain
arcos@iiia.csic.es

ABSTRACT

The embellishments of makam music in Turkey are an inherent characteristic of the music rather than a separate expressive resource, thus their understanding is essential to characterize this music. We do a computational study, in which we analyze audio recordings of 8 widely acknowledged Turkish ney players covering the period from the year 1920 to 2000. From the extracted fundamental frequency, we manually segment and identify 327 separate embellishments of the types *vibrato* and *kaydırma*. We analyze them and characterize the behavior of two features that help us differentiate performance styles, namely vibrato rate change and pitch bump. Also we compare these embellishments with the ones used in Western classical music. With our approach, we have an explicit and formalized way to understand ney embellishments, which is a step towards the automatic characterization of makam music in Turkey.

1. INTRODUCTION

Each music repertorie has specific characteristics that require specific analysis approaches [11]. This is clearly the case of the makam music in Turkey and there has been very few computational studies that focus on it.

Makam music in Turkey is mainly an oral tradition and thus the audio recordings become a fundamental source of information for its study [1]. For this research approach we need well annotated large data sets, and we need to extract the appropiate audio features from which to then perform musically meaningful computational studies.

Among all the characteristics of makam music, the embellishments are the most relevant ones, they are more than simple expressive resources, in fact they are essentials of the music [12]. Embellishments are not taught even named by teachers. Their places are not marked in the score, the choice and character of the embellishment is the freedom of the performer. Ney is one of the oldest and most characteristic instrument of makam music in Turkey. The way the embellishments have to be performed in the ney is not taught and their places are not marked in the score. The choice and character of the embellishment is very much in the freedom of the performer and thus they are valuable for understanding the difference between performers and between performances.

In this paper we focus on two types of embellishments used in ney performances, *vibrato* and *kaydırma*. This choice is the result of interviews and discussions with well known ney players [1] . To characterize these embellishments we combine the use of well known features used in previous studies with our proposed audio features.

The paper is organized as follows; In Section 2 we provide a brief introduction to the makam music in Turkey and to the ney and its performance practice. Section 3 describes the data set used and in Section 4 we describe the features that we have developed. We summarize and discuss our findings in Section 5.

2. BACKGROUND

2.1 Makam Music in Turkey

Makam is a complex musical concept that cannot be defined in a simple straightforward way. Moreover there are differences between theory and practice of makams. In theoretical terms, a given makam is described by the set of the tones or notes that compose it, but in practice it is often much more complex. The upper and lower extensions of a tone in the makam may be regarded as tones belonging to the same makam.

An important practical aspect of a Makam is the *seyir*, which can be translated as the way that the performer uses the notes or, briefly, his/her *navigation*. The important notes for the seyir are; *baslangic*(start), and *karar* (decision). A possible translation would be the initial and finish tone.

Another complex phenomenon is intervals of makams. There is a long lasting and never ending discussion about makams in Turkey [12], but we can say that there are more than 17 intervals in an octave [12]. Among the different interval systems, it is common practice to use the Holde-

[1] We have interviewed with Osman Erkahveci, Orçun Güneşer, Ali Tan and and Oğuz Mülayim.

Figure 1: Examples of Different Neys.

rian comma (Hc) as the smallest intervallic unit in makam music in Turkey [2].

2.2 Ney

The ney is an end-blown flute which is mainly used for makam music. It is one of the oldest and most characteristic blown instrument of makam music in Turkey. From the beginning of the 20th century a score representation which was developed by extending the Western music is used. However because of the extensive use of embellishments, the written scores are far from the music that is actually performed. Ney is taught and transmitted orally.

Ney has a real importance and solid place in Turkish classical and religious music. Ney is made of reed, and it is a rim-blown, oblique flute. The Turkish ney has six finger-holes in front and a thumb-hole in back. Although it is highly dependant on the talent and experience of the performer, a ney can produce any pitch over a two-and-a-half octave range or more. Nearly all Turkish neys have a mouthpiece made of water buffalo horn, or sometimes ivory, ebony, plastic, or a similar durable material. Also there are different sizes of neys', ranging from the Davud ney (95 cm long), to the highest, Bolahenk Nısfiye ney (52.5 cm long).

2.2.1 Ney and Its Performance Practice

Ney tradition is transmitted via master-pupil relationship in Turkey. Written scores only represent the border lines of the pieces. The embellishments, which are distinctly important, are never marked or even explicitely taught. The main way to learn how to apply these embellishments is to learn from masters via listening, which makes it very hard. On the other hand, without embellishments makam music is considered dry, monotonous and not deemed as acceptable [12]. This is specially so in ney music. Although we have not found any documents describing the techniques used, our study shows the existance of clear patterns in the performance of these embellishments, (Section 5).

From our interviews with experts we realized that the naming of ney embellishments is a problem in Makam music. The Ney players we interviewed agreed on naming frequency and amplitude modulation as *Vibrato*. However they all had difficulty naming the embellishment that is widely used for connecting two consecutive notes and that in some Makam literature is called *Kaydırma* [2] .

[2] The literal translation can be *sliding*, however the purpose of this behavior is to give the feeling of non-edge connections all through the piece rather than sliding between notes. Possible the most similar embellishment in western music is the *portamento*.

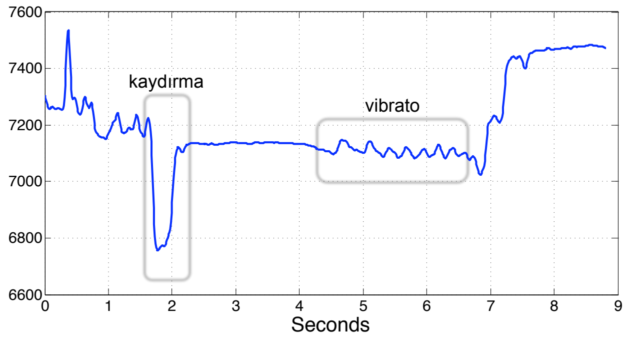


Figure 2: Example note sequence.

Performer	Time (Minutes)	#Embellishments
Hayri Tümer	7.75	16
Ulvi Ergüner	7.33	16
Aka Gündüz	7.12	45
Niyazi Sayın	7.66	53
Sadrettin Özçimi	11.20	66
Salih Bilgin	8.50	44
Ömer Bildik	2.33	21
Burcu Sönmez	5.50	66

Table 1: Performers with their test data.

From our inital quantitative studies we found that vibrato and kaydırma are the most used embellishments. We will further explain the behaviour of both embellishments but lets just say that the vibrato is a much more precise musical entity than kaydırma (Section 5) and that the kaydırma includes several subsets of embellishments.

3. DATA COLLECTION

For our analysis we annotated 8 different performers from different eras. Our set contains recordings starting from 1930's to now. Our concern was to analyse and characterise the ney embellishments rather than differentiating among them. Since our study is one of the first about embellishments in makam music, our first concern is to understand which embellishments were and are used by well known ney performers and then characterize these embellishments by using fundamental frequency analysis.

Our data set includes 58 minutes of audio and a total number of 327 hand annotated embellishments, summarized in Table 1.

Except Hayri Tümer and Ulvi Ergüner, there are around 6-8 embellishment for each minute of audio. For Hayri Tümer and Ulvi Ergüner, embellishment per minute is between 2 to 4. One of the reason for this difference is that these two players are the oldest ones, from 1930's, and that they shared a different style. However, this hypothesis should be given with further studies with a much more larger data set.

3.1 Performers

We are covering some of the most acknowledged ney players. According to our oral discussions with professional

Turkish ney players, Niyazi Sayın and Aka Gündüz Kutbay are considered as one of the most influential ney players of today. However because of the sudden death of Aka Gündüz Kutbay at the age of 45, most of the recent players are influenced by Niyazi Sayın. Through the oral discussions with Ali Tan [3] , he stated that even in Turkish Conservatories teachers follow the way of Niyazi Sayın. Moreover, most of the ney players (both amateur and professional), who even did not have a chance to study with Niyazi Sayın, they consider themselves as students of his by listening and studying his recordings . In our test set Salih Bilgin and Sadrettin Özçimi are one of the most famous students of Niyazi Sayın. Burcu Sönmez is a student of both Salih Bilgin and Niyazi Sayın. Ömer Bildik is a student of Sadrettin Özçimi. All these ney players have the influence of Niyazi Sayın.

In our analysis set, in order to avoid lineage bias we also include some old ney players recordings like Hayri Tümer, Aka Gündüz Kutbay and Ulvi Ergüner, who are also well-known and highly respected ney players with distinct styles.

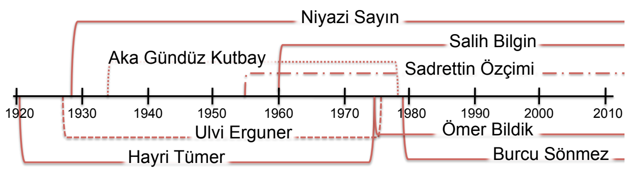


Figure 3: Lifespan of ney performers we used for our analysis. We carefully chose these performers in order to cover most of the recorded era of ney.

4. ANALYSIS

Our method is a combination of state of the art signal processing techniques and empirical observations. From the audio files, we obtain first the fundamental pitch of each recording. After that, we analyze the annotated sections one by one. Each embellishment is analyzed according to the behavior of its fundamental frequency (Section 4.1).

4.1 Fundamental Frequency

To obtain a fundamental frequency estimation of each solo ney recording, Makam Toolbox was used [1]. Makam Toolbox uses an implementation of Yin [3] with hop size of 10ms for fundamental frequency estimation. On the top of the f0 implementation Makam Toolbox makes a post-processing for octave correction.

All embellishment samples are measured in the 1/3 Holderian Comma (HC) resolution. We choose this resolution because it is considered as the highest precision we could find in theoretical pitch scale studies [1]. All measurements were taken manually. For the statistical significance, all pieces vary in tempo and also chosen from different Makams. HC is widely considered as the smallest interval therefore we used HC in our tables.

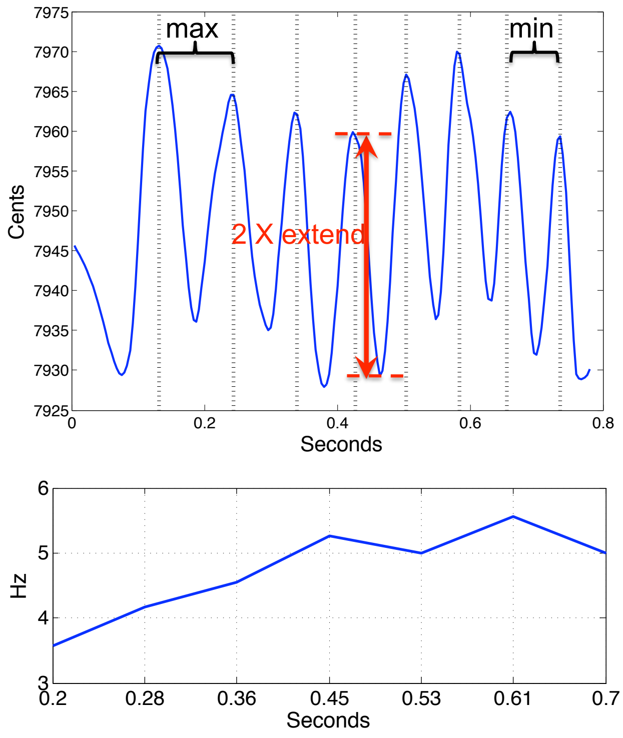


Figure 4: Top figure, is an example of the change in fundamental frequency in a vibrato. Bottom figure provides the change in rate of Vibrato that is shown in top figure.

4.2 Vibrato

According to Tura [12], the extend of the vibrato in Makam music in Turkey is around 1 HC. Both from oral discussions and written resources [5,12], we could not obtain any particular information about the structure of the vibrato in ney.

For the vibrato analysis, 170 different vibratos from 8 different ney players were analyzed. Each embellishment was analyzed one by one manually. In Table 2, the vibrato rate is given in Hz and the extend value is given in HC resolution. We also propose another feature for the deep analysis of the vibrato in Ney, *Vibrato Rate Change*.

4.2.1 Extend

Extend value is calculated according to the change in the mean f0 during vibrato. On the top graph of Figure 4, extend value is shown with red arrow. For the analysis, we followed the definition of Tura [12]. In his book he defined vibrato in makam music as the upper and lower change in f0 in Holderian Comma. Therefore, the extend value that is shown in top graph of Figure 4 is actually 2 times the values that are shown in Table 2. For each vibrato of each performer, we calculated its extend.

4.2.2 Rate

For each performer and for each vibrato, we measured the maximum and the minimum rate values, top graph of Fig-

[3] Ali Tan is a full-time research asistant in Istanbul Technical University Turkish Conservatory in the Ney performance department. http://www.neyzenalitan.com/.

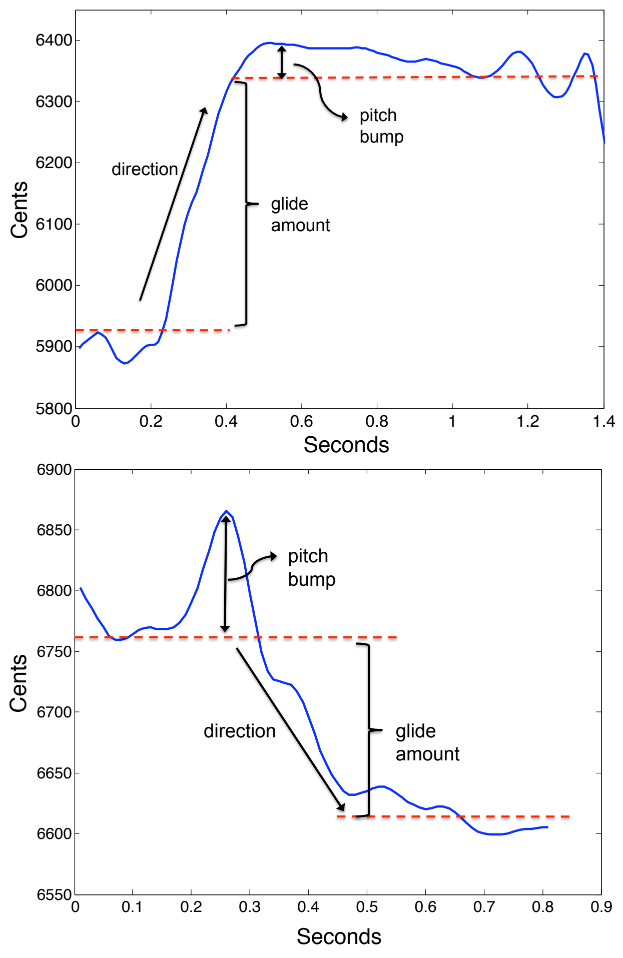


Figure 5: Kaydırma pitch graphs.

ure 4. These values are reported in Table 2.

4.2.3 *Rate Change*

For the initial tests we applied the *AR Prediction* method [10] and the *Automatic Detection* technique [8]. Both implementations have its pros and cons. AR prediction technique was successful for detecting vibratos regions with having more than 4-5 periods. Automatic Detection technique could detect vibratos with having less than 4-5 periods but since it is specialized for string instruments, it also gave false positives if the player performers non-periodic f0 deviations.

In both techniques features of the vibrato are the rate and extend. Although we are not aiming to propose an automatic detection algorithm, after these initial tests with the existing algorithms, we realized that for the characterization of vibrato in ney, rate and extend features are not enough.

As shown in the top graph of Figure 4, the distance between peeks decrease. Thus, as shown in the bottom graph of Figure 4, the vibrato rate is increasing. According to our analyses, we discovered that this increase in rate is a characteristic behavior that most of the ney players apply. Vibrato Rate change in Table 2 is the ratio of maximum rate versus the minimum rate for a single vibrato. Minimum value represents the minimum rate change the performer applied among all his/her vibratos and maximum is vice versa.

4.3 Kaydırma

As we stated in Section 2.2.1, the main purpose of kaydırma is to give the feeling of a smooth transition between notes. However, after analyzing 157 different kaydırmas from 8 different performers, we discovered that its characteristic is much more complicated than a simple transition. Different from the possible equivalent study on string instrument, [7], kaydırma has a distinct difference; its glide amount is not constant. Moreover as shown in Table 3, in the glide column in both ascending and descending kaydırmas, the Standard Deviation σ, values are so high that, we can conclude that the glide amount is not predictable.

Thus, we add a new feature named *pitch bump*. It models the pitch deviation just before the ascending or descending transition. These features are shown in Figure 5.

4.3.1 *Direction*

This feature represents the movement direction of the glide. Performers use both ascending and descending glides.

4.3.2 *Glide Amount*

Glide amount is the difference between the target note and the base note in Holderian Commas. It is observed that there is not a fixed amount of glide, Table 3. However, we observed that most of the time during the ascending kaydırma, the difference between the base note and the transition note is much more higher than the descending kaydırma.

4.3.3 *Pitch Bump*

As shown in Figure 5, for both ascending and descending kaydırma, there is a lower or higher bump in the f0 just before the transition note. According our observations, this is a characteristic movement for kaydırma. We also confirmed this behaviour with our oral discussions with ney performers.

5. DISCUSSION

In this study we analyzed two distinct embellishments of ney, vibrato and kaydırma. Both embellishments have similar counterparts in Western classical music.

Vibrato has different characteristics for different instruments. For string instruments of Western classical music, vibrato extend ranges from 0.2 to 0.35 semitones which corresponds to 0.9 to 1.5 HC, and for the singing voice of Western Classical music it ranges 0.6 to 2 semitones, 2.7 to 9 HC [13].

As reported in Table 2, vibrato extend has a strong regularity among performers. If we check the mean values for the performers, we may observe that except Hayri Tümer, all of them either have a regularity of 1 or so close to 1. These results match with the analyzes of Tura for the vibrato of makam music in Turkey [12]. Hayri Tümer is the oldest ney player among the performers we analyzed. We

	VIBRATO					
		Extend (HC)		Rate (Hz)	Rate Change	
Performer	#	min / max	μ / σ	min / max	min / max	μ / σ
Hayri Tümer	9	**0.5** / 0.7	0.57 / 0.1	1.8 / 7.6	1.3 / 2.7	1.69 / 0.58
Ulvi Ergüner	10	0.8 / 1.2	0.9 / 0.19	3.21 / **9.01**	1.23 / 2	1.61 / 0.32
Aka Gündüz	31	0.7 / 1.5	1.1 / 0.2	2.8 / 6.2	1.18 / 1.96	1.49 / 0.26
Niyazi Sayın	29	0.5 / 1	0.9 / 0.15	2.6 / 6.6	1.12 / 1.94	1.41 / 0.25
Sadrettin Özçimi	31	0.8 / 1.3	0.9 / 0.21	2.7 / 5.5	**1.05** / 1.94	1.38 / 0.26
Salih Bilgin	25	0.8 / 1.2	1 / 0.1	3.2 / 6.66	1.15 / 1.73	1.34 / 0.21
Ömer Bildik	14	0.5 / **1.7**	0.98 / 0.14	2.7 / 4.5	1.43 / 1.8	1.57 / 0.19
Burcu Sönmez	21	0.8 / 1.4	1.1 / 0.23	**1.7** / 6.2	1.19 / **2.8**	1.67 / 0.39
All Performers	170	0.5 / 1.7	0.93 / 0.18	1.7 / 9.01	1.05 / 2.8	1.44 / 0.25

Table 2: Vibrato analysis table. The Extend column, min is the minimum value among all vibratos and max is the vice versa. We also computed the mean μ, and standard-deviation σ, values for all vibratos of each performer.

	KAYDIRMA								
		Ascending				Descending			
		Glide Amount		Pitch Bump		Glide Amount		Pitch Bump	
Performer	#	min / max	μ / σ	min / max	μ / σ	min / max	μ / σ	min / max	μ / σ
Hayri Tümer	7	12 / 23	16.5 / 5.06	1 / 7	2.75 / 3	7 / 10	9 / 1.73	1 / 3	1.67 / 1.15
Ulvi Ergüner	6		-	-	-	3 / **17**	9.6 / 5.02	1 / **6**	3.66 / 2.51
Aka Gündüz	14	7 / 35	18.6 / 13.7	1 / 2	1.1 / 0.3	1 / 7	4 / 4.24	1 / 2	1.5 / 0.7
Niyazi Sayın	24	**3** / 53	19.5 / 13.8	1 / 4	1.8 / 1.13	1 / 10	4.27 / 3.23	1 / 2	1.34 / 0.53
Sadrettin Özçimi	35	5 / 59	28.7 / 20.4	1 / 3	2.17 / 0.75	1 / 12	8.25 / 2.98	1 / 4	2.46 / 1.05
Salih Bilgin	19	7 / 23	20 / 15.3	1 / 3	1.8 / 1.15	2 / 15	7.45 / 2.45	1 / 3	2.2 / 0.8
Ömer Bildik	7	10 / 18	14.7 / 12.1	1 / 8	3.7 / 1.14	3 / 15	6.2 / 3.1	1 / 3	2 / 1.1
Burcu Sönmez	45	5 /**114**	28 / 25.78	1 / 5	2 / 1.49	3 / 9	4.58 / 2.14	1 / 5	2.75 / 1.89
All Performers	157	3 / 114	21.87 / 17.3	1 / 8	2.13 / 1.75	1 / 17	5.67 / 3.43	1 / 6	2.51 / 1.43

Table 3: Kaydırma analysis table. μ represents the mean value and σ represents the standard deviation of all the kaydırma excerpts.

do not have the exact dates for the recordings but we are assuming that they were recorded between 1930 and 1950. He is the only player that has the vibrato extend around 0.5 Holderian Coma. This distinction also can be heard in his recordings. He has smoother and lighter vibrato compared to all other players.

In Western music vibrato rate changes from 4-12Hz [4]. In our analyses, vibrato rate changes from 1.8 to 9Hz. If we avoid the extremes we can say that in ney vibrato rate changes from 2 to 7Hz. As reported in Table 2, in the Rate column, performers have different choices.

On the contrary, the rate change feature has a strong regularity. When we analyze the mean values of rate change, they are between 1.34 to 1.69 and the standard deviation values are less than 0.5. This means that the vibrato rate changes between 30% to 60% through the end of the vibrato. In western music, rate change is common for violin and soprano singers [9]. However, their rate change is much less, around 15%.

In his thesis [6], Mallikarjuna described the features of portamento. He considered portamento as a smooth transition between two notes. However different from the portamento, kaydırma has different characteristics. As shown in Table 3, in the Glide column, minimum and maximum values vary a lot. Although mean values are close, the high standard deviation values show that there is no regularity for the glide amount. It can be 1 Holderian Coma or 114 Holderian Coma (around 2 octaves).

The important finding we obtain for *glide amount* is that ascending kaydırma has typically much bigger glide amount then descending kaydırma. Mean values for ascending kaydırma are around 16 to 28 Holderian Coma whereas for descending kaydırma they are around 4 to 9 Holderian Comas.

The unique feature for ney kaydırma is, *pitch bump*. It is common for both ascending and descending kaydırma. Moreover, different from the glide amount, pitch bump has the same characteristics for both ascending and descending kaydırmas. For all of the players except Hayri Tümer, the amount of pitch bump is, for ascending 1-8 and for descending 1-5 Holderian Comas. If we check the mean values, for both ascending and descending it is around 1-3 Holderian Coma. Therefore we can conclude that there is strong regularity for pitch bump for both ascending and descending kaydırma. Low standard deviation values support our observation.

6. CONCLUSION

In this paper we characterize the most commonly used embellishments in ney. Our analyzes show that there are differences between ney embellishments and their Western equivalents. One of the important step was to characterize these differences. We proposed two new features for this characterization, vibrato rate change and kaydırma pitch bump. We analyzed 327 embellishments of 8 different performers from different eras of makam music in Turkey.

We believe that with our study, understanding of ney embellishments are much more clear for both musicological and computational studies.

7. ACKNOWLEDGMENTS

We would like to thank Barış Bozkurt for the insightful discussions. Also we would like to thank Osman Erkahveci, Ali Tan, Oğuz Mülayim and Orçun Güneşer for sharing their valuable knowledge with us.

8. REFERENCES

[1] Barış Bozkurt. An automatic pitch analysis method for turkish maqam music. *Journal of New Music Research*, 37:1–13, 2008.

[2] Barış Bozkurt, Ozan Yarman, Kemal Karaosmanoğlu M., and Akkoç. Weighing diverse theoretical models on turkish maqam music against pitch measurements: A comparison of peaks automatically derived from frequency histograms with proposed scale tones. *Journal of New Music Research*, 38, March 2009.

[3] A. de Cheveigné and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[4] P Desain and H Honing. Modeling continuous aspects of music performance: Vibrato and portamento. In *Proc. Int. Conf. on Music Perception and Cognition*, 1996.

[5] Suleyman Erguner. *Ney Method*. Erguner Muzik, 2007.

[6] T. Mallikarjuna. Towards expressive melodic accompaniment using parametric modeling of continuous musical elements in a multi-attribute prediction suffix trie framework. Master's thesis, Georgia Institute of Technology, USA, 2010.

[7] T. Özaslan and J.Ll. Arcos. Legato and glissando identification in classical guitar. In *7th Sound and Music Computing Conference (SMC)*, pages 457–463, July 2010.

[8] T. Özaslan and J.Ll. Arcos. Automatic vibrato detection in classical guitar. In *Technical Report IIIA*, 2011.

[9] E Prame. Vibrato extent and intonation in professional western lyric singing. *Ac. Soc. of America*, 102:616–621, 1997.

[10] S. Rossignol, P. Depalle, J. Soumagne, X. Rodet, and J.-L. Collette. Vibrato: Detection, estimation, extraction, modification. In *In Proc. Digital Audio Effects Workshop (DAFx)*, 1999.

[11] Xavier Serra. A multicultural approach in music information research. In *Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, Miami, Florida (USA), 24/10/11 2011.

[12] Yalçın Tura. *Türk Musikisinin Meseleleri.* Pan Yayıncılık, 1988.

[13] Vincent Verfaille, Catherine Guastavino, and Philippe Depalle. Perceptual evaluation of vibrato models. In *Conference on Interdisciplinary Musicology (CIM05)*, pages 1–19, 2005.

CROSS-CULTURAL MUSIC MOOD CLASSIFICATION: A COMPARISON ON ENGLISH AND CHINESE SONGS

Yi-Hsuan Yang
Academia Sinica
yang@citi.sinica.edu.tw

Xiao Hu
University of Denver
xiao.hu@du.edu

ABSTRACT

Most existing studies on music mood classification have been focusing on Western music while little research has investigated whether mood categories, audio features, and classification models developed from Western music are applicable to non-Western music. This paper attempts to answer this question through a comparative study on English and Chinese songs. Specifically, a set of Chinese pop songs were annotated using an existing mood taxonomy developed for English songs. Six sets of audio features commonly used on Western music (e.g., timbre, rhythm) were extracted from both Chinese and English songs, and mood classification performances based on these feature sets were compared. In addition, experiments were conducted to test the generalizability of classification models across English and Chinese songs. Results of this study shed light on cross-cultural applicability of research results on music mood classification.

1. INTRODUCTION

There have been a number of studies on music mood classification in the Music Information Retrieval (MIR) community in recent years [7][17]. However, most of existing studies have focused on Western music, in particular English songs. The two mood-related tasks in the Music Information Retrieval Evaluation eXchange (MIREX): Audio Mood Classification (AMC) and Audio Tag Classification (mood tag set) have been using datasets consisting of Western music [3]. Although such research activities have shown promising performances on classifying Western music by mood, there is little research on whether and how the mood categories and techniques applied to Western music can be equally well applied to non-Western music. This study aims to bridge the gap using Chinese contemporary pop songs as a case of non-Western music. In particular, three research questions are answered in this study:

1) How well mood categories developed from English songs can be applied to Chinese songs and what are the differences of mood distributions among Chinese and English songs?
2) Are audio features commonly used in mood classification of Western songs applicable to Chinese songs?
3) Can prediction models built using English songs be reliably applied to Chinese songs?

Answers to these questions will further our understanding on cross-cultural generalizability of music mood categories, audio features and classification models.

2. RELATED WORK

2.1 Mood Categories in Western and Chinese Music

In building datasets for evaluating mood classification algorithms, MIR researchers have used a variety of mood categories. Quite a number of studies have used four categories derived from the four quadrants of Russell's *valence-arousal* dimensional model [12]: *contentment, depression, exuberance, and anxious/frantic* (or similar ones). It is noteworthy that these four categories have been used for both Western music [10][14] and non-Western (e.g., Chinese) music [4][14].

In the industry, online music repositories may organize music by mood. For example, allmusic.com, a large and influential online repository for Western popular music, provides 182 mood labels that are applied to songs and albums by professional music editors. In fact, the mood categories used in the AMC task in MIREX (cf. Table 1) were based on the most popular mood labels on allmusic.com [3]. However, no research has been done to investigate whether these mood labels are suitable for non-Western music. In this study, we take on this challenge using Chinese pop music as a case (see Section 3).

2.2 Mood Classification of Western Music

Quite a number of studies have been conducted on automated mood classification on Western Music, as reviewed in [7][17]. Most of these studies extracted acoustic features from music audio files. Some studies combined acoustic features with features extracted from other information sources such as lyrics and social tags [2][9]. As one of the first studies comparing mood classification techniques on Western and non-Western music, this paper focuses on acoustic features and leave it to future work to compare approaches using combined information modals.

The classification models often used include neural network, k-nearest neighbor (k-NN), maximum likelihood, decision tree, and support vector machines (SVM). Of these, SVM seems the most popular due to its reliable classification performance. In this study, we use SVM as the classification model for all the experiments as our focus is on the generalizability of acoustic features.

2.3 Mood Classification of Chinese Music

There have been few studies on mood classification of non-Western music and they are predominately on Chinese music. Hu *et al.* [4] and Xia *et al.* [15] classified Chinese pop songs using lyric features. Yang *et al*. combined lyrics and audio features to classify Chinese pop songs [17]. It is noteworthy that all these previous studies used four or fewer mood categories (e.g., two categories were used in [15], "lighthearted" and "heavy-hearted."). Moreover, none of these studies compared Chinese music to Western music on either mood categories or mood classification techniques and performance.

2.4 Comparison between Mood Classification on Western and Chinese Music

To the best of our knowledge, [14] is the only existing study that evaluated the same mood classification techniques on both Western and Chinese music. Wu *et al.* [14] extracted three acoustic feature sets of pitch, rhythm and timbre from 20 pieces of Chinese traditional instrumental music and 20 pieces of Western classical music. They used a Bayesian probabilistic structure to classify the moods in these pieces into four mood categories based on the Russell model and the results indicated better classification performance on Western pieces. At the same time, the authors found "different identification about the moods between the Oriental and Western culture" (p.152).

Our study differs from this prior work in the following aspects. First, the music we focus on is popular music from Western and Chinese cultures. Second, while [14] used four mood categories, we compare more than 30 mood labels in terms of their distributions in Chinese and English songs. Third, the dataset in [14] was evidently too small (20 pieces in each culture) to draw reliably conclusions, whereas our study uses a dataset of 500 Chinese pieces and even more English pieces. Fourth, in addition to the rhythm and timbre features examined in [14], our study also examines dynamics, MFCCs, psychoacoustic and tonal features. Fifth, besides comparing classification performances on music in each culture, our study also investigates the generalizability of classification models built with music in one culture being applied to music in another culture (cf. Section 4.2).

3. MOOD CATEGORIES IN CHINESE SONGS

This section describes how we determine the mood categories and obtain the mood annotations of Chinese songs from the experts we recruited and those of English songs from allmusic.com.

3.1 Allmusic.com Mood Labels and Translation

To answer our first research question, we chose mood labels contained in the five mood clusters used in the AMC task in MIREX [3]. This is due to the following reasons. First, the AMC mood clusters were derived from the most popular mood labels on allmusic.com, one of the most popular sites for Western music, and thus they are representative to the mood of Western songs. Second, allmusic.com provides "top songs" for each of its mood labels, which empowers us to obtain English songs with expert-annotated mood labels. Third, the 29 mood labels in AMC mood clusters (see Table 1) are of finer granularity than the commonly used four categories based on Russel's model [12]. Previous studies based on the four categories were not able to discern the difference of mood distributions among Western and Chinese songs [14]. Fourth, as pointed out by music psychology research [6], classical models like Russell's may not reflect the social context of everyday music listening. Since allmusic.com serves a large quantity of listeners in real life, its mood labels are closer to the reality of music listening at present time.

C1	Rowdy, rousing, confident, boisterous, passionate
C2	Amiable/good natured, sweet, fun, rollicking, cheerful
C3	Literate, wistful, bittersweet, autumnal, brooding, poignant
C4	Witty, humorous, whimsical, wry, campy, quirky, silly
C5	Volatile, fiery, visceral, aggressive, tense/anxious, intense

Table 1. Mood categories used in MIREX AMC task [3].

It is noteworthy that, in discussions with music experts (see below), seven additional labels were added in because the experts believed they might be important to represent moods in Chinese pop songs: "calm/peaceful," "dreamy", "encouraging," "nostalgic," "relaxed," "soothing," and "tender," making 36 mood categories in total. Except for "encouraging" and "tender", the other five labels had appeared on allmusic.com at various time points (as allmuic.com changes its mood labels from time to time).

The mood categories were translated into Chinese for consistent understanding among the annotators who are native Chinese speakers. The translation was first done by one of the authors who is a native Chinese speaker and fluent in English. The translation and the original English terms were then examined by three expert annotators. The four of them discussed several difficult cases (e.g., "autumnal," "visceral") before reaching agreements on the translations.

3.2 Selection of Chinese Songs

The Chinese pop songs were collected from an in-house collection which contains albums released in Taiwan, Hong Kong and Mainland China during the years from 1987 to 2010. To maximize the diversity of songs, we selected one song from each album to be included in this study. Songs with non-Chinese (mostly English, some in Japanese) titles were eliminated because they were not in Chinese despite being sung by Chinese artists. This process resulted in 500 Chinese songs.

An excerpt of 30 seconds was extracted from each song, for the purposes of limiting the burden of human annotators and mitigating the cases where mood changes during the entire course of a song [7]. Using 30 second excerpts rather than entire songs is a common practice in MIR, but it remains in debate which 30 second segment should be chosen [17]. For the purpose of mood classification, we decided to choose the segment with strongest emotion from each song. Specifically, we used a sliding window of 30 seconds to exhaustively extract all 30 second segments from each song, and then used a regression model to predict the *valence* and *arousal* values of each segment. The segment with highest ($|valence|^2 + |arousal|^2$) value was chosen to represent each song. Please note that the regression model was built on an external dataset of Western pop music [16], as there were no Chinese songs with proper *valence* and *arousal* annotations that could be used to train the regression model.

3.3 Annotation of Chinese Songs

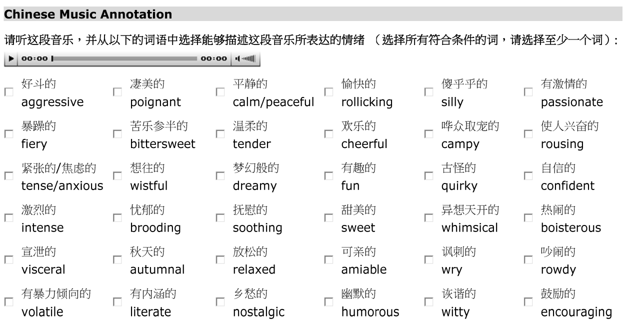
To ensure the quality of annotation and to reduce the variance of annotations across annotators, this study adopts the approach of expert annotation.

Three experts were recruited from a university in the United States. All of them were female, Chinese (Mandarin) native speakers, raised in Mainland China, majored in music, and fans of Chinese pop music. Table 2 shows their demographic and background information.

ID	Age	Specialty	Year in college	Years in the US	Freq. of listening to Chinese pop music
1	23	Theory	Senior	4	Several times a week
2	25	Violin	Graduate	0.5	Daily
3	25	Vocal	Graduate	3	Several times a week

Table 2. Information about the experts.

The annotation was conducted through a web-based survey system. Figure 1 shows the interface of annotating a piece. One or more mood labels could be applied to each music piece. This is more in accordance to the reality where a song can express multiple moods [2].


Chinese Music Annotation

请听这段音乐，并从以下的词语中选择能够描述这段音乐所表达的情绪 （选择所有符合条件的词，请选择至少一个词）:

00:00 00:00

☐ 好斗的 aggressive	☐ 凄美的 poignant	☐ 平静的 calm/peaceful	☐ 愉快的 rollicking	☐ 傻乎乎的 silly	☐ 有激情的 passionate
☐ 暴躁的 fiery	☐ 苦乐参半的 bittersweet	☐ 温柔的 tender	☐ 欢乐的 cheerful	☐ 哗众取宠的 campy	☐ 使人兴奋的 rousing
☐ 紧张的/焦虑的 tense/anxious	☐ 想往的 wistful	☐ 梦幻般的 dreamy	☐ 有趣的 fun	☐ 古怪的 quirky	☐ 自信的 confident
☐ 激烈的 intense	☐ 忧郁的 brooding	☐ 抚慰的 soothing	☐ 甜美的 sweet	☐ 异想天开的 whimsical	☐ 热闹的 boisterous
☐ 宣泄的 visceral	☐ 秋天的 autumnal	☐ 放松的 relaxed	☐ 可亲的 amiable	☐ 讽刺的 wry	☐ 吵闹的 rowdy
☐ 有暴力倾向的 volatile	☐ 有内涵的 literate	☐ 乡愁的 nostalgic	☐ 幽默的 humorous	☐ 诙谐的 witty	☐ 鼓励的 encouraging

Figure 1. Screenshot of the annotation interface.

Before the annotation started, all three experts met together with one of the authors for a training session. Annotation requirements were made clear during the session, including "focusing on the mood expressed by the pieces instead of mood induced in yourselves;" "making use of the lyrics but without looking them up anywhere;" "ignoring the order by which the mood labels are arranged." Also in the training session, the experts listened to eight 30 seconds long pieces and discussed which moods each piece expressed. Half of the eight pieces were English songs with very different moods selected from a previous study [16] and the other half were randomly selected from the Chinese pieces to be annotated. Through the discussion, the experts reached better agreement on musical meanings of the mood labels.

Each piece was assigned to one expert and the three experts annotated 150, 200 and 150 pieces respectively. Using one human judge's opinion as gold standard for evaluation has been verified to be effective in the domain of text information retrieval [13]. Admittedly, this has yet to be verified in MIR, which will be our future work. The experts reported that each song took each of them about 1 minute to annotate and were paid for their work on an hourly basis.

3.4 Mood Categories in Chinese and English songs

A total of 2,453 mood labels were applied to the 500 pieces, with one piece getting 1 to 13 labels. On average, each piece had 5 labels (standard derivation: 2.08). Each of the 36 mood labels were applied to 6 to 202 pieces, with an average of 68.14 pieces each label (standard derivation: 40.85). The most popular mood labels among the 500 Chinese songs were "tender" (202), "wistful" (164), and "passionate" (128). The least popular mood labels were "volatile" (6), "wry" (6), and "nostalgic" (8). The distribution of Chinese songs across mood categories is shown in the left panel of Figure 2.

As comparison, we counted the number of "Top Songs" provided by allmusic.com for each of the mood labels, as shown in the right panel of Figure 2. Note that the numbers of Chinese songs in Figure 2 are limited to the songs available to us. While allmusic.com has a much larger song pool, they only provided up to 100 top songs for each mood label. Despite this, it is still clear from Figure 2 that there are more Chinese songs than English songs labeled with "relaxed," "wistful," "passionate," "brooding," or "rousing." On the other hand, there are much fewer Chinese songs labeled with "wry," "volatile," "humorous," "aggressive," and "fiery." Such difference might reflect the differences between the Chinese and Western cultures: the Chinese culture tends to restrain the expression of feelings and Chinese people are more introverted compared to Western people [11]. Another possible reason for the absent of radical moods (e.g., "aggressive," "fiery") in Chinese songs might be that Chinese popular music has a shorter history comparing to Western one and the whole mood spectrum is yet to be developed.

To further illustrate the applicability of the mood categories to Chinese songs, we also examined and compared the relative distance among mood categories based on the two datasets. The distance between a pair of mood categories was calculated based on the common songs shared by them. We then projected the mood labels based on their distances to a 2-D space using multidimensional scaling (MDS) for each of the two datasets, as shown in Figure 3. It can be found that the relative positions of mood labels for both sets share some similarity: “aggressive,” “fiery,” “intense” and “volatile” are clustered together and away from low arousal moods such as “amiable” or “wistful.” Another two clusters shared by both plots are also in line with common sense: “rollicking,” “cheerful,” “passionate” and “rousing”; “literate,” soothing” and “bittersweet”.

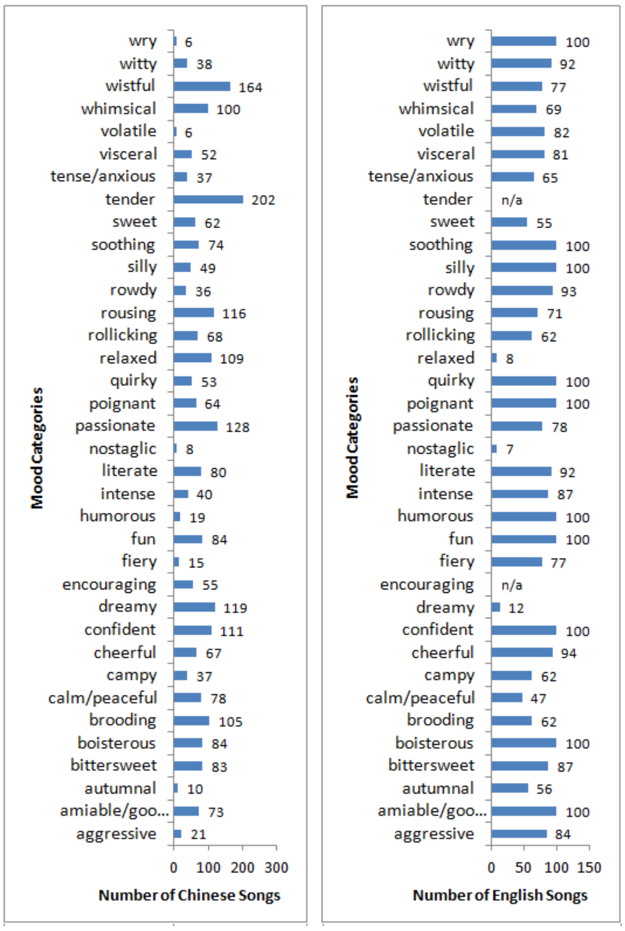
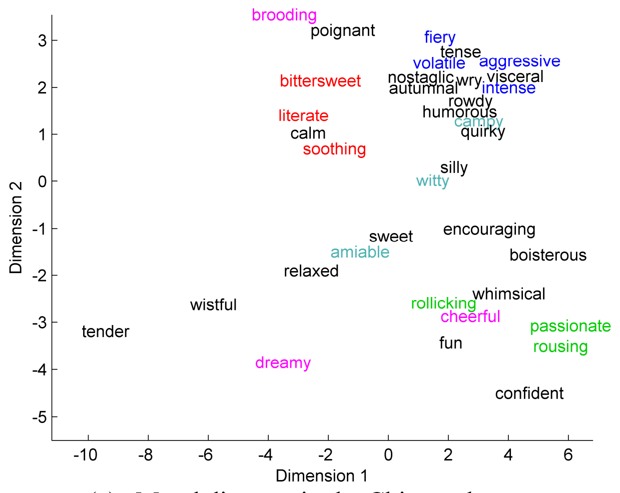
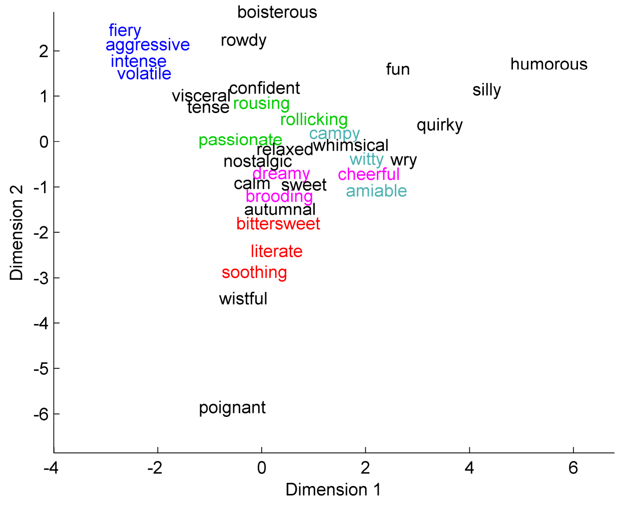


Figure 2. Song distribution across categories.

The two plots in Figure 3 also differ in several ways. “Dreamy,” “brooding” and “cheerful” are close to one another in the English dataset but are separated out in the Chinese dataset which seems more intuitive. As another example, the English set puts “campy” and “witty” together with “cheerful” and “amiable” while the Chinese set separates them apart. This difference might be related to the cultural contexts. The Chinese experts interpreted the first two words as neutral and the last two as positive. However, an English native speaker said he would associate the four terms to “fun” although the first two terms were more of a kind of “dark” fun. In sum, the relative distance among mood labels in the Chinese set, although somewhat different from that in the English set, seems agreeable with the semantic meanings of the terms.



(a) Mood distance in the Chinese dataset



(b) Mood distribution for the English dataset

Figure 3. Projection of mood categories to a 2-D space.

From the comparisons of song distributions across mood categories as well as the relative distance between them, we can see that the mood categories used to describe English songs are generally applicable to Chinese songs with exceptions of radical moods such as “fiery” and “volatile.”

4. CLASSIFICATION EXPERIMENTS

To answer research questions 2 and 3, we conducted mood classification experiments on Chinese and English songs. To build a dataset of English songs, we collected the “Top song” lists from allmusic.com for mood labels used in this study and then obtained 30-second audio previews from 7digital.com which boasts itself as “a leading digital media delivery company.” A total of 1,520 English song clips were collected.

All experiments were set up as binary classifications for each mood category, without considering possible correlations between categories. Positive examples of a mood category are songs labelled with that category while negative examples are randomly selected from songs labelled with other categories. Positive and negative examples are balanced both in training and test sets. The classification is carried out by SVM, with RBF kernel and the two parameters *C* and *γ* tuned. Each experiment was repeated 20 times and average accuracy values are reported. In our evaluation, we only used the mood classes with more than 20 positive examples in both Chinese and English datasets. Throughout the experiments, the datasets were split into 50% training and 50% test.

4.1 Effectiveness of Acoustic Features

In view of the complexity of mood perception, it is difficult to find a universal feature representation that well characterizes every mood. In addition, the perceptions of different moods in music are usually associated with different patterns of acoustic cues [5]. We therefore extracted acoustic features that represent various perceptual dimensions of music listening and trained classifiers using features of each perceptual dimension separately.

A total of six feature sets were examined in this study, as summarized in Table 3. They were extracted using the MIR toolbox [8] and the PsySound toolbox [1]. These features have been used extensively in previous work on mood classification [7][17].

Feature	Type	Dim	Description
RMS	Energy	2	The mean and standard deviation of root mean square energy
PHY	Rhythm	5	Fluctuation pattern and tempo [8]
PCP	Pitch	12	Pitch class profile, the intensity of 12 semitones of the musical octave in Western twelve-tone scale [8]
TON	Tonal	6	Key clarity, musical mode (major/minor), and harmonic change (e.g., chord change) [8]
MFCC	Timbre	78	The mean and standard deviation of the first 13 MFCCs, delta MFCCs, and delta delta MFCCs
PSY	Timbre	36	Psychoacoustic features including the perceptual loudness, volume, sharpness (dull/sharp), timbre width (flat/rough), spectral and tonal dissonance (dissonant/consonant) of music [1]

Table 3. Feature representations adopted in our study.

Figure 4(a) shows the average classification accuracy of the binary classification tasks on the two datasets. The six audio descriptors performed well in both sets and even better for the Chinese songs. The relative performances among the feature sets are similar in both datasets: timbre descriptors performed better than energy or rhythm related descriptors. The fact that PCP and TON features worked well for Chinese songs reflects that the composition of contemporary Chinese pop songs is influenced by the Western twelve-tone scale. In addition, we find that PSY performed the best for both datasets with an average accuracy of 74.7% and 65.8%, respectively. The performance differences between PSY and the first four features sets are significant (pair-wise *t*-test, $p < 0.001$). This shows that the psychoacoustic features seem to be generally applicable to Chinese songs [1].

It is interesting that significantly better performance (pair-wise *t*-test, $p < 0.001$) is obtained for Chinese songs than for English songs. Although both datasets were annotated by experts (recruited by allmusic.com and by us, respectively), the allmusic.com song lists contain “tier1” (more representative) and “tier2” (less representative) songs. Our inclusion of tier2 songs may have introduced noise to the English dataset. In contrast, all the Chinese songs were annotated with the same criteria. The performance difference may also result from the fact that the excerpts of the Chinese songs were segments with strong emotion, whereas the English excerpts were provided by 7digital which may not represent the full songs annotated by allmusic.com experts.

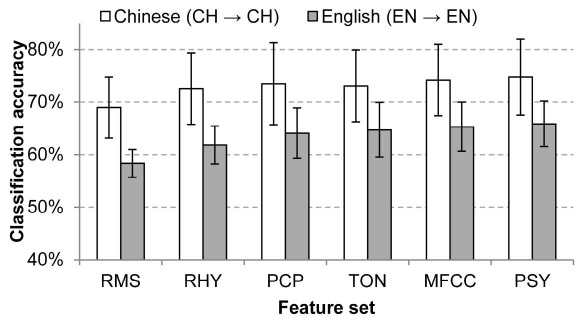
4.2 Cross-cultural Applicability of Classifiers

In this subsection, we report the result of using classifiers trained from English songs to classify Chinese songs, and vice versa. This set of experiments is designed to study the cross-cultural applicability of classification models which has rarely been addressed in the literature. Figure 4(b) shows the performances across the six feature sets.

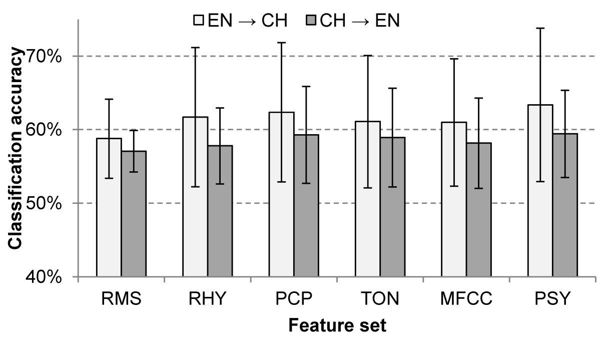
PSY was again the best performing feature set, with average accuracies of 63.3% and 59.4% for Chinese and English test data, respectively. The performance differences between PSY and other feature sets are significant when Chinese songs are used as the test set (pair-wise *t*-test, $p < 0.001$). While the performances are comparable to those in the literature [3][17], they are significantly worse than those of last experiments where training and testing data were drawn from the same dataset (pair-wise *t*-test, $t = 5.92$ for Chinese songs; $t = 5.09$ for English songs, $df = 25$, $p < 0.001$). This means the datasets in the two cultures have significant difference. Whenever possible, it is better to use songs in the same culture to train classification model. However, in cases when training data from the same culture as test data are not available, it is still an acceptable alternative to use classification models built with data in the other culture.

Using English songs as training data to classify Chinese songs performed significantly better than the other way around (pair-wise *t*-test, $t = 3.70$, $df = 25$, $p = 0.001$). This may be because there were more English songs making larger training sets and/or mood classification of Chinese songs seemed to be easier (c.f. Section 4.1).

To further examine possible differences in acoustic characteristics between English and Chinese songs, we applied MDS to project the PSY features of the songs to a 2-D space (Figure 5). The fact that the two datasets overlap to a large extent indicates that Chinese songs and English songs in our datasets have similar perceptual timbre quality (as depicted by the PSY features [1]). This may partially explain the fact that PSY features performed well in both English and Chinese songs as well as the cross-cultural experiments.



(a)



(b)

Figure 4. Average accuracy of different feature sets for mood classification of (a) intra-cultural classification using Chinese and English datasets and (b) inter-cultural classification using the other set as training data.

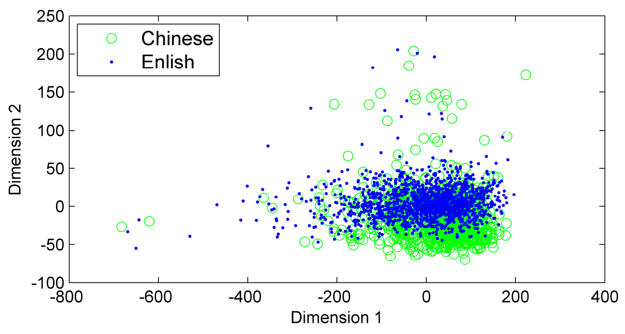


Figure 5. Visualizing PSY features of the two datasets.

5. CONCLUSIONS AND FUTURE WORK

This study investigates the cross-cultural applicability of mood categories, acoustic features and classification models in the case of English and Chinese songs. Results show that mood categories found in English songs are generally well applicable to Chinese songs except for several categories representing radical moods. It also seems feasible to apply feature descriptors developed for English songs to represent the audio content of Chinese songs, possibly due to the overlap of Psychoacoustic timbre features in both datasets. Our cross-cultural evaluation showed significant degradation of classification performance compared to the result of within-culture evaluation, although the absolute accuracy values are still comparable to the state-of-the-art in the literature.

In future work, we will examine the cross-cultural applicability of audio features and classification models on individual mood categories. We also plan to explore the problem of predicting the valence and arousal values of Chinese songs and investigate whether techniques that worked for Western music will work for Chinese music.

6. REFERENCES

[1] D. Cabrera: "Psysound: A computer program for psychoacoustical analysis," in *Proc. Australia Acoustic Society Conf.*, 1999.

[2] X. Hu and J. S. Dowine: "Improving Mood Classification in Music Digital Libraries by Combining Lyrics and Audio," in *Proc. ACM/IEEE Joint Conf. Digital Libraries* (*JCDL*), pp. 159–168, 2010.

[3] X. Hu, J. S. Dowine, C. Laurier, M. Bay, and A. F. Ehmann: "The 2007 MIREX audio mood classification task: Lessons learned," in *Proc. ISMIR*, 462–467, 2008.

[4] Y. Hu, X. Chen, and D. Yang, "Lyric-based song emotion detection with affective lexicon and fuzzy clustering method," in *Proc. ISMIR*, 2009.

[5] P. Juslin, "Cue utilization in communication of emotion in music performance: Relating performance to perception," *J. Experimental Psychology*, vol. 16, no. 6, 2000.

[6] P. N. Juslin and P. Laukka, P.: "Expression, perception, and induction of musical emotions: a review and a questionnaire study of everyday listening," *J. New Music Research*, vol. 33, no. 3, pp. 217–238, 2004.

[7] Y. E. Kim *et al*: "Music emotion recognition: A state of the art review," in *Proc. ISMIR*, 2010.

[8] O. Lartillot and P. Toiviainen: "MIR in Matlab (II): A toolbox for musical feature extraction from audio," in *Proc. ISMIR*, pp. 127–130, 2007.

[9] C. Laurier, J. Grivolla, and P. Herrera, "Multimodal music mood classification using audio and lyrics," in *Proc. Int. Conf. Machine Learning and Applications*, pp. 1–6, 2008.

[10] L. Lu, D. Liu, and H. J. Zhang: "Automatic mood detection and tracking of music audio signals," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 5–18, 2006

[11] R. R. McCrae, P. T. Costa, and M. Yik: "Universal aspects of Chinese personality structure," in M. H. Bond (Ed.) *The Handbook of Chinese Psychology*, pp. 189–207. Hong Kong: Oxford University Press, 1996.

[12] J. A. Russell: "A circumspect model of affect," *J. Psychology and Social Psychology*, vol. 39, no. 6, 1980.

[13] E. M. Voorhees and D. K. Harman: *TREC: Experiments in Information Retrieval Evaluation*, MIT Press, 2005.

[14] W. Wu and L. Xie: "Discriminating mood taxonomy of Chinese traditional music and Western classical music with content feature sets," in *Proc. IEEE Congress on Image and Signal Processing*, pp. 148–152, 2008.

[15] Y. Xia, L. Wang, K. Wong, and M. Xu: "Sentiment vector space model for lyric-based song sentiment classification," in *Proc. ACL*, pp. 133–136, 2008.

[16] Y.-H. Yang and H.-H. Chen: "Prediction of the distribution of perceived music emotions using discrete samples," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2184–2196, 2011.

[17] Y.-H. Yang and H.-H. Chen: "Machine recognition of music emotion: A review," *ACM Trans. Intelligent Systems and Technology*, vol. 3, no. 3, 2012.

TOWARDS A (BETTER) DEFINITION OF THE DESCRIPTION OF ANNOTATED MIR CORPORA

Geoffroy Peeters
STMS IRCAM-CNRS-UPMC
Paris, France

Karën Fort
INIST-CNRS & Université Paris 13,
Sorbonne Paris Cité, LIPN
Nancy, France

ABSTRACT

Today, annotated MIR corpora are provided by various research labs or companies, each one using its own annotation methodology, concept definitions, and formats. This is not an issue as such. However, the lack of descriptions of the methodology used—how the corpus was actually annotated, and by whom—and of the annotated concepts, i.e. what is actually described, is a problem with respect to the sustainability, usability, and sharing of the corpora. Experience shows that it is essential to define precisely how annotations are supplied and described. We propose here a survey and consolidation report on the nature of the annotated corpora used and shared in MIR, with proposals for the axis against which corpora can be described so to enable effective comparison and the inherent influence this has on tasks performed using them.

1. INTRODUCTION

The use of annotated data usually corresponds to increasing performances in a field of research, as has been seen in the cases of speech and language processing. The accessibility of novel annotated data usually corresponds to the initiation of a number of research activities in a field. This is the case of music genre, chord recognition, and music structure in music information retrieval (MIR). For this reason, annotated data can be considered to be a major issue in MIR. In MIR, there is currently no dedicated institution responsible for providing music corpora comparable to ELRA[1] or LDC[2] in the speech and natural language processing (NLP) community. Instead, corpora are provided individually by various research labs and companies. While recent years have seen a large increase in corpora creation initiatives (e.g. Isophonic, SALAMI[3], Billboard, and Quæro), each research lab or company uses its own annotation methodology, concepts definition, and format. This is not a problem in and of itself, but the lack of descriptions of the methodology used, i.e. how the corpus was actually annotated, or of the concepts annotated, i.e. what is actually described, presents problems with respect to the sustainability, usability, and sharing of corpora. Therefore, it is essential to define exactly what and how annotations of MIR corpora should be supplied and described. We propose here an avenue to improve this situation by defining a methodology for describing MIR corpora and the implicit or explicit assumptions made during their creation. It should be noted that similar initiatives have been taken in the speech and NLP community to favor sharing and exchange of corpora (see for example [1], [2] [3]) leading to descriptions close to the one proposed here.

[1] `http://www.elra.info/`
[2] `http://www.ldc.upenn.edu/`
[3] Structural Analysis of Large Amounts of Music Information

2. DEFINING AN ANNOTATED MIR CORPUS

In the following, by annotated corpora, we mean "musical audio data with annotations". Such corpora can be used for research purposes to derive knowledge or train systems, or for benchmarking and evaluation projects, both internal and public, as in MIREX[4]. Creating an annotated MIR corpus involves:

(A) choosing or creating a set of audio items (denoted by "raw corpus" in the following),
(B) creating and/or attaching related annotations, and
(C) documenting and storing the results to ensure sustainability and sharing.

While these points may seem obvious, each of them involves making choices that in the aggregate will define what exactly the corpus is about, what use it is for, and what the underlying assumptions behind it are. In the following, we provide insights about the choices that must be explicitly or implicitly made for each of these points, and the implications of those choices. Figure 1 summarizes the various aspects of the proposed description.

2.1 (A) Raw Corpus

In the case of "audio MIR," the annotations describe audio items[5], which we denote here by the term "raw corpus", as opposed to the "annotated corpus". The choice of these audio items defines the domain, or musical area, for which the **results derived from** the annotations—results

[4] MIREX: Music Information Retrieval Evaluation eXchange
[5] Whether the annotations are distributed with or without the audio items, the following remains true.

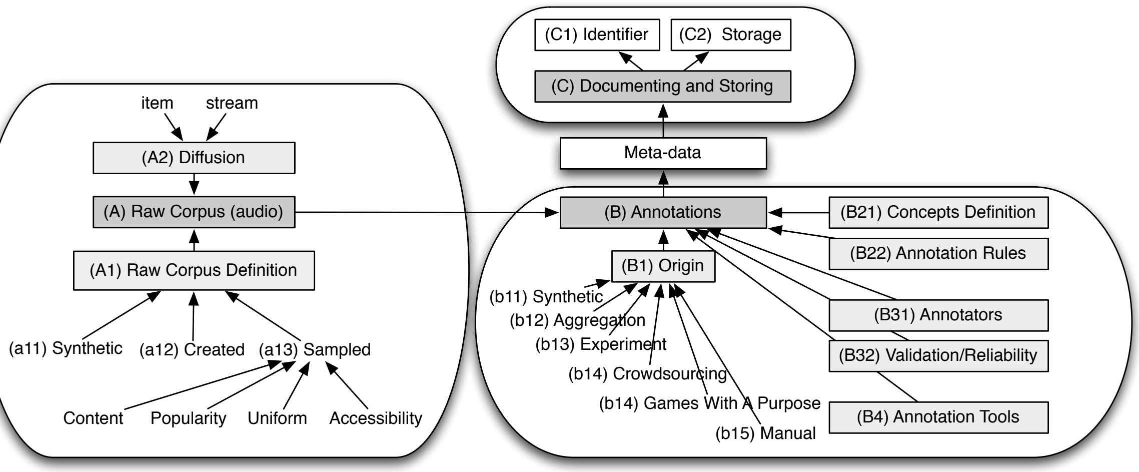


Figure 1. Decomposing the creation of a MIR annotated corpus into the tasks and sub-tasks involved.

from an experiment, training, or evaluation—are valid. For example, results derived from the music genre defined for the Tzanetakis test-set [4] do not generalize to the Million-Song test-set. The choice of audio items also determines the domain for which the **concepts defined by** the annotations are valid. This is specific to the way annotation is performed in the MIR field: while in other domains, concepts are first defined, and then used for the annotation of items, in MIR, the concepts are (in most cases) defined by the annotations themselves. For example, "music genre" is not defined textually, but rather is defined by its application to a specific test-set, such as Tzanetakis or Million Song. The same is true for "chords," whose meaning may differ in the context of the data to which it is applied: in the Beatles [5], it refers to guitar chords, but when applied to Billboard songs, it is a reduction of the overall harmonic content. Because of this, special care must be taken when selecting audio items for an annotated MIR corpus.

It is clear that results obtained from experiments on (i) synthesized MIDI files, (ii) audio recorded for the purposes of an experiment, or (iii) audio as sold by online music services, do not have the same impact. Note that this in no way means that one is better than the others.

To help describe the choices made so far in MIR, we propose to distinguish between three categories:

- a11 - artificial audio items made specifically for the purpose of representing an annotation
- a12 - real audio items recorded specifically for the purpose of the creation of a corpus
- a13 - real audio items sampled from the real world

2.1.1 *(a11) Corpus of Synthetic Items*

This kind of corpus is specific to the music research community, based on the assumption that, within certain limits, rendering a MIDI file can create a music audio signal. Examples of this are [6] for the creation of a multi-pitch estimation corpus, [7] for the case of chords, and the MIREX corpus used for key estimation.

There are several *advantages* to this approach. It allows (i) having close to perfect annotations very easily, since audio can be partly considered to be a direct instantiation of the annotation; (ii) having full control over the audio rendering process, such as testing the influence of instrument changes, reverberation or delay; and (iii) rapidly creating a large corpus. Its major *drawback* is the lack of realism due to (i) the absence of interpretation by musicians; (ii) the absence of realism due to sound variations, propagation, and capture by microphone; and (iii) the absence of "production" as made in recording studios.

2.1.2 *(a12) Corpus of Created Real Items*

The second trend consists in creating specific audio items for the purpose of research. The first corpora prepared in this way were built for "instrument samples" research—McGill [8] and Studio-On-Line [9]. In this case, the annotation—pitch, instrument name, and playing mode—is added during the recording session. Corpora for multi-pitch, source separation, and event recognition have also been created, such as the ENST Drum database [10], containing audio and video, and the MAP database [11], using a Yamaha disklavier for automatic pitch annotations. The most well-known and used corpus in MIR research is such a data set—the RWC corpus [12, 13].

The *advantages* of this approach are that it allows (i) complete specification of the underlying content property, (ii) easy creation of the annotations at the same time as the occurrence, and (iii) distribution of the corpus with no restrictions, as the creator of the corpus usually owns the audio copyright. The main *drawback* of this is again the lack of realism of the resulting audio items—e.g., RWC is a very valuable resource but does not sound like iTunes music. This is partly due to the recording conditions that a lab can afford—expensive compressors and enhancers remain in the big production studios. Also, the music composition used is often prototypical. All of this frequently creates a bias when using these corpora outside of the context of the experiment for which they were built.

2.1.3 (a13) Corpus of Sampled Real Items

The last trend corresponds to what has long been known as "using a private music collection". These could not be shared, mostly due to copyright issues. Today, because of the possibility of referencing audio items by ID (CD reference or Musicbrainz/ EchoNest/ Amazon/ 7-Digital ID), there is a major trend toward these corpora. The main *advantage* of this type of corpus is that it represents exactly the music people listen to or buy, with artistic interpretation and professional sound production. It also allows the evaluation of concepts that are well-established in the literature for their applicability to everyday music (see the case of the "chord" concept). The major drawback of this type of corpus is the cost of the annotations, which involve either human annotation (by dedicated people or by crowdsourcing) or data aggregation (for example, aggregating guitar-tab collections or music-recommendation sites).

However, underlying a corpus created by sampling the real world lies a major question: how was the sampling done? For which reasons or purposes were the specific music tracks selected? This is actually rarely described, with the exception of [14], which provides an in-depth description of the sampling for the Billboard corpus. We distinguish here between four trends:

Specific-content sampling: The sampling is done in order to highlight specific content characteristics. An example of this is the corpus proposed by [15] for music structure. It consists of a selection of tracks from Eurovision (the European Song Contest), i.e. pop songs with a typical pop structure. Another is the corpus proposed by [5] for chord annotation, which consists primarily of a selection of tracks from The Beatles, essentially made of guitar chords. While this perfectly fits the purpose of their annotations, care must be taken with respect to the validity of the concepts (e.g. the specific definition of structure or chords) outside of the context of these corpora.

Popularity-oriented sampling: The sampling is done according to what people were or are reading, listening to, or watching the most. An example is given in [14], in which the sampling is performed based on the Billboard charts. However, in this case, some music genres might be over-represented.

Uniform sampling: The sampling is done in a uniform way according to a description grid. The dimensions of this grid, as in our project, may represent music genre/style, year, or country[6]. In each resulting cell of the grid, the most popular audio items are selected. In this case, some music styles can be over-represented.

Accessibility-oriented sampling: The last trend consists in selecting items because they are freely available (e.g. Magnatune and Internet Archive), without any other considerations.

2.1.4 (A2) Type of Media Diffusion

Apart from the choice of the sampling process, the type of media diffusion also needs to be decided during the process of corpus creation. Corpora can represent isolated music tracks, but may also include items as diverse as music inside a TV/Radio audio stream (as in the corpus of [16]), the audio part of a video clip or User-Generated-Content videos, a live recording, a bootleg, or a DJ-remix. This implies different audio qualities, and also the possible presence of interfering sounds such as speech, applause, and the ambient atmosphere of live performances.

[6] These meta-data can be provided, e.g. by AMG. It should be noted, however, that the source of meta-data can create a bias.

2.1.5 Definition of the Media Coding Properties

Finally, the audio properties also have to be described, in terms of such variables as frequency bandwidth; the presence of drops, noise, pops, hisses or clicks (due for example to media trans-coding from vinyl); and the number of channels—mono, stereo, or multi-channel.

2.2 (B) Attaching Annotations to Audio Items

Although this is probably the most important aspect of an annotated corpus, it is often the one that is least described (except if the annotations were the subject of a dedicated publication, as in the case of the results of a listening experiment [17]). The main points to detail are the following:

- Where do the annotations come from?
- What do they represent? How are they defined?
- What is their reliability?

2.2.1 (B1) Origin of the Annotations

The central question is the origin of the annotations. We distinguish here between four different cases:

Automatic annotations

• (b11): The annotations are obtained by the **synthesis** parameters [6] (a11), as scores given during the recording process or analysis of the individual tracks of the recordings [13] (a12). In this case, the generative process of the music defines both the labels used for the annotation and the annotation itself. Its reliability is very high.

• (b12) The annotations are obtained by **aggregation** of diverse extant content. Examples of this are the Million Song Test-Set [18] and the use of Guitar-Tab in [19]. In this case, each annotation and its definition and reliability are defined by its provider: Last-FM data are obtained through crowdsourcing, Echo-Nest data are algorithm estimations, and MusicXMatch contains official lyrics.

Manual annotations:

• (b13): The annotations are the results of an **experiment**. In this case, the definition of the annotation is provided by the guidelines of the experiment. The reliability of the annotation is derived from the experimental results, either in a summarized form (e.g. two major peaks of the tempo histogram in [17]) or from the whole set of annotations, letting the user decide the way to summarize it (e.g. perception of tempo and speed in the case of [20]).

• (b14): **Crowdsourcing**, in particular **Games With A Purpose** (GWAP). In this case, annotations are obtained using various game processes [21–24]. The labels used for the annotation are either determined before the game, providing an existing frame of reference; or determined by the users during the game, allowing free input. In both cases, the definitions of the labels are not provided (although they may be inferred by another gamer's choices), but rather are defined by the use that gamers make of them. In this context, when a reliability measure of the annotation is proposed, it is usually derived from the number of occurrences of a label [24].

• (b15): Traditional **manual** human annotations. Examples of these are [5,25,26].

(b13), (b14) and (b15) are the most interesting for us here, since they involve thinking about the definition of the annotation concepts and the techniques for performing the annotations and measuring their reliability. Manual annotation is (very) costly, so the annotation process should ensure quality and reusability. In the field of natural language processing, the authors of [27] show that "corpora that are carefully annotated with respect to structural and linguistic characteristics and distributed in standard formats are more widely used than corpora that are not".

2.2.2 (B2) Definitions

(B21) Concepts: The term *annotation* refers both to the process of adding a note or a label to a flow of data (such as audio music, speech, text or video) and to the result of this process—the notes themselves, anchored in the source flow. The annotations are all the more useful to the extent that they are designed for a specific application [28]. Depending on the final application, the labels may not carry the same semantics. The semantics may even be completely different—for example, annotating football matches with the intent of producing an automatic summary [29] is very different from annotating football matches for purposes of linguistic analysis. In speech and natural language processing, saying that we may find as many annotation models as there are annotation projects is not too far from reality. In MIR, it seems that the same concepts are always used, with different meanings that are sometimes only implicit.

In the case of manual human annotations, the concepts to be annotated must be defined. The absence of definition is clearly a problem for a set of tasks in MIR (beat [7], chord [8], and structure [9], to name just a few). Recently, efforts have been made to clarify the concepts being annotated through dedicated papers or through the on-line availability of so-called "annotation guides" [15,26]. Those efforts should be encouraged. It must be noted that the use of annotation guidelines has been considered part of "best practices" in speech and natural language processing for some time, following the trend in this direction in corpus linguistics [28].

(B22) Rules: Beyond the definition of the concepts being annotated, the annotations are performed using a set of rules. This set of rules should also be described. For example, what is the temporal precision used for segment annotations? Which type of dictionary was used for the labels? Are there equivalences between labels? To exemplify the difference between *concept* and *rules used to annotate this concept*, consider an experiment in a recent project to annotate beat/tempo. Two different rules were used. The first was to do annotation of beats and then infer the tempo curve from that; the second was to adjust a tempo curve so as to align a sequencer grid to an audio track, and then infer beat positions. The two methods describe the same concept, but lead to different results (data not shown).

2.2.3 (B3) Actors and Quality

(B31) Who are the annotators? Annotators may be students or researchers, creating a corpus that will directly fit their research, with the model of their algorithm in mind while annotating; musicians, with a strong ability to apply the concepts with respect to detailed musical structure, sometimes losing sight of overall perception; or everyday people. This choice influences the way the annotation is performed.

(B32) Reliability of the annotation? Although they are considered to be able to generate "gold standards", humans are not perfect annotators. The definitions of the concepts to be annotated might not have been defined clearly, they may not fit the content of a given audio file, there might be several plausible possibilities for a particular annotation, or the annotator may lose concentration. The question of the reliability of the annotation is therefore another major issue. For this reason, it is common practice to do cross-validation of the annotations. This can be done by applying either or both of two scenarios. In the first scenario, an annotated track is validated or corrected by a second annotator. In the second scenario, the same track is annotated independently by at least two annotators. The resulting annotations are then compared by computing the inter-annotator agreement (using the Kappa coefficient [31] or other measures [10]). A decision is then made whether the annotation is sufficiently reliable, or whether it should be redone using the same definitions and rules, or whether the definitions or rules should be modified. In speech and natural language processing, computing the intra-annotator agreement (agreement of an annotator with him/herself as the project progresses) is also considered to be good practice and allows the detection of potential issues with the annotators [33]. This is already done in sound perception experiments, and could be extended to annotation projects.

Overall, the methodology used should be documented and detailed. In speech and natural language processing, the typical methodology includes early evaluation of the annotation guidelines using inter-annotator agreement, the update of these guidelines with the help of the annotators' feedback, regular checking, continuous use of inter- and

[7] Given that beat is mostly a perceptual concept, what is the metrical level being annotated?

[8] In the case of chord annotations, what is the definition of chords? Are we considering the perceived chord of the background accompaniment? Do we also consider the lead vocal? Are the chords derived from the guitar part?

[9] The case of music structure is even less defined. What is a chorus? A segment? Why could a segment not be further divided into sub-segments or grouped into meta-segments? Considering this, the proposal made in [30] to store the various possible annotations is worth mentioning.

[10] It must be noted, however, that the resulting coefficient of agreement (Cohen's Kappa or others) is far from being wholly sufficient as a metric when used in isolation, and should be accompanied by details of the choices that were made to compute it. In this respect, the contingency table provides more interesting information about the annotation reliability than the inter-annotator agreement itself [32].

intra-annotator agreement and/or precision measures (going so far as the so-called “agile annotation” [34]), and a final evaluation of the resource that has been produced.

2.2.4 (B4) Annotation Tools

Caution is necessary in selecting the appropriate annotation tool, as the limitations of the tool will impact the annotation model. For example, there may be relations that are impossible to annotate, or the interface may contain a feature that is difficult to access and hence seldom used.

2.3 (C) Documenting and Storing the Results to Ensure Sustainability and Sharing

Corpus sharing and distribution does not simply require putting all of the audio data and annotations into an archive file. From our point of view, it implies providing information on all of the above-mentioned points (A* and B*). We provide here some additional recommendations for improving the distribution process.

2.3.1 (C1) Corpus Identification

Currently, most corpora in MIR research or MIREX benchmarking have no identifier (except RWC, Isophonic, or Million Song Test-Set). They are referred to as “the corpus used in the publication of [reference]”. A unique identifier should be assigned to each corpus, including versioning of the annotations and annotation guidelines (see [35]). This could take the form of a simple URI (example of this would be corpus:MIR:qmul:2004:beatles:chords:version1.0) or used the more elaborated Vocabulary of Interlinked Datasets [11] . This would solve some ambiguity issues, such as when a corpus is updated over time (for example the SALAMI corpus), or when one set of annotations is revised by another lab and later included in a new corpus (for example the structure annotations of The Beatles).

2.3.2 (C2) Storage of the Created Annotations

Annotations must be sustainable. We therefore recommend that the storage of the data make their semantics explicit. Up to this point in time, many annotations of local-in-time concepts such as beat, chord, and structure were done in formats where the semantics is implicit in the corpus. In particular, the so-called “.csv” or “.lab” formats (one row for time, one row for labels) would not be sustainable outside of the context of a given corpus [12] . RDF (as used by QMUL [5]) or XML (as used by [30]) seem good choices. For the later, the MPEG-7 xml shema [36] already proposes a full-range of description with inherent semantic and the possibility to define new semantics using Classification Schemes (CS). Whatever choice, the definition of and the reference to a controlled list of labels is necessary. It also allow to define the width of the description-space [13] .

Providing a precise reference to the audio items being described is also crucial. Considering that recent annotated corpora were distributed without audio media, this is clearly a major issue. Several linkage mechanisms between annotations and audio media have been proposed so far: CD reference (as in Isophonic), Musicbrainz ID (as in the Million Song) or the EchoNest ID, Amazon ID, 7-Digital ID. The reference should also allow referencing time inside the files. The example of the alignment problem of the Beatles annotations to the various possible audio instances is notorious in the MIR community. Inclusion in the annotation of time-stamped identification, such as is provided by audio-ID techniques, would help.

(C1) Corpus ID:	corpus:MIR:AIST:RWC:2006:version1.0
(A) Raw Corpus	
(A1) Definition:	(a12) created real items; 315 tracks created for the specific purpose of having a copyright-free test-set for MIR research representative of the various genres, styles, instrumentation, vocal types (see [12] for details)
(A2) Type of media diffusion:	full tracks stereo high-quality
(B) Annotations	
(B1) Origin:	(b11) synthetic—obtained during creation and (b15) manual annotations
(B21) Concepts definition:	only defined by the annotation rules
(B22) Annotation rules:	- Standard MIDI Files (SMF) transcribed by ear, - Lyrics of songs obtained during creation, - Beat/ downbeat annotated using metronome clicks of recording and manual editing, - Melody line annotated using fundamental frequency estimation on the melody track and manual editing, - Chorus sections method is not indicated, - Audio synchronized MIDI Files using the annotated beat positions
(B31) Annotators:	a music college graduate with absolute pitch
(B32) Validation/ reliability:	not indicated
(B4) Annotation tools:	“Music Scene Labeling Editor”
(C) Documents and Storing	
(C2) Audio identifier and storage:	RWC-specific audio identifiers, audio files are available through audio CDs, annotations available through archive files in CSV format

(C1) Corpus ID:	corpus:MIR:LastFM:Tempo:2011:version1.0
(A) Raw Corpus	
(A1) Definition:	(a13) sampled real items. Sampling method: somehow uniform—4006 tracks chosen “essentially at random” among several thousands
(A2) Type of media diffusion:	30s extract of music items
(B) Annotations	
(B1) Origin:	(b13) Experiment and (b14) Crowd-Sourcing
(B21) Concepts definition:	the concepts are defined by the results of the experiments, itself defined by the instructions provided to the annotators: ”tap along to each except”, ”describe its speed on a three point scale”, compare two tracks in terms of speed.
(B22) Annotation rules:	defined by the experiment protocol (see [20] for details)
(B31) Annotators:	2141 users of Last-FM (not all tracks are annotated by all the annotators)
(B32) Validation/ reliability:	for each track, all the annotations are provided, it is let to the user of the corpus to compute inter-annotator agreement
(B4) Annotation tools:	Web-interface
(C) Documents and Storing	
(C1) Audio identifier and storage:	no audio identifiers are provided (except the artist, album and track name); annotations distributed as an archive file accessible through an URL, files in TSV format (Tab Separated Values File).

Table 1. Application of the proposed description to the corpus of [12, 13] and [20]

[11] http://www.w3.org/TR/void/

[12] Consider the question of how a user will interpret the “1” label ten years from now.

[13] This would for example make it possible to decide whether a C-Maj chord is really a C-Maj or a reduction of a C-Maj7+9 chord.

3. EXAMPLES OF DESCRIPTIONS

As examples of the application of the proposed description, we illustrate in Table 1 its use for the (short) description of two corpora [12,13] and [20]. It should be noted that these descriptions are solely based on the information provided with the distributed corpora and the respective publications and should ideally be complemented and corrected by the respective authors themselves. Based on this, a comparative table of the corpora can easily be made [14].

4. CONCLUSION

MIR should benefit from the "best practices" that have been evolving for decades in the speech and natural language processing communities. Among these practices, we attempt here to provide insights into the choices currently made when creating a MIR annotated corpus, their implications, and the resulting necessity to better describe them when distributing an annotated corpus. We presented them in the form of a numbered list—A*, B*, C*—to highlight the fact that all of these choices must be described. Considering the importance that the distribution of annotated corpora will have to the development of MIR research, we hope that providing this list will facilitate the sharing and re-use of annotated corpora.

Ackowledgements: This work was party supported by the Quaero Program funded by Oseo French State agency for innovation and by the MIReS project funded by EU-FP7-ICT-2011.1.5-287711. Many thanks to Kevin B. Cohen for proof-reading.

5. REFERENCES

[1] F. Schiel, C. Draxler, *et al.*, "Production and validation of speech corpora," *Bavarian Archive for Speech Signals. Munchen: Bastard Verlag*, 2003.

[2] A.-J. Li and Z.-g. Yin, "Standardization of speech corpus," *Data Science Journal*, vol. 6, no. 0, pp. 806–812, 2007.

[3] M. Wynne, ed., *Developing Linguistic Corpora: a Guide to Good Practice*. Oxford: Oxbow Books, 2005.

[4] 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.

[5] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Klozali, D. Tidhar, and M. Sandler, "OMRAS2 metadata project 2009," in *Proc. of ISMIR (Late-Breaking News)*, (Kobe, Japan), 2009.

[6] C. Yeh, N. Bogaards, and A. Roebel, "Synthesized polyphonic music database with verifiable ground truth for multiple f0 estimation," in *Proc. of ISMIR*, pp. 393–398, 2007.

[7] T. Fujishima, "Realtime chord recognition of musical sound: a system using common lisp music," in *Proc. of ICMC*, (Beijing, China), pp. 464–467, 1999.

[8] F. Opolko and J. Wapnick, "McGill university master samples CD-ROM for SampleCell VOLUME 1," 1991.

[9] G. Ballet, R. Borghesi, P. Hoffman, and F. Lévy, "Studio online 3.0: An internet "killer application" for remote access to ircam sounds and processing tools," in *Proc. of JIM*, (France), 1999.

[10] O. Gillet and G. Richard, "Enst-drums: an extensive audio-visual database for drum signals processing," in *Proc. of ISMIR*, pp. 156–159, 2006.

[11] V. Emiya, R. Badeau, and B. David, "Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1643–1654, 2010.

[12] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "Rwc music database: Popular, classical, and jazz music databases," in *Proc. of ISMIR*, (Paris, France), pp. 287–288, 2002.

[13] M. Goto, "Aist annotation for the rwc music database," in *Proc. of ISMIR*, (Victoria, Canada), pp. 359–360, 2006.

[14] J. A. Burgoyne, J. Wild, and I. Fujinaga, "An expert ground-truth set for audio chord recognition and music analysis," in *Proc. of ISMIR*, (Miami, USA), 2011.

[15] F. Bimbot, E. Deruty, S. Gabriel, and E. Vincent, "Methodology and resources for the structural segmentation of music pieces into autonomous and comparable blocks," in *Proc. of ISMIR*, (Miami, USA), 2011.

[16] M. Ramona, S. Fenet, R. Blouet, H. Bredin, T. Fillon, and G. Peeters, "A public audio identification evaluation framework for broadcast monitoring," *Journal of Experimental and Theoretical Artificial Intelligence (Special Issue on Event Recognition)*, 2012.

[17] D. Moelants and M. F. McKinney, "Tempo perception and musical content: What makes a piece slow, fast, or temporally ambiguous?," in *International Conference on Music Perception and Cognition*, Evanston, IL, 2004.

[18] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proc. of ISMIR*, (Miami, USA), 2011.

[19] M. McVicar and T. De Bie, "Enhancing chord recognition accuracy using web resources," in *Proc. of the 3rd international workshop on Machine learning and music*, pp. 41–44, ACM, 2010.

[20] M. Levy, "Improving perceptual tempo estimation with crowd-sourced annotations," in *Proc. of ISMIR*, (Miami, USA), 2011.

[21] Y. E. Kim, E. Schmidt, and L. Emelle, "Moodswings: A collaborative game for music mood label collection," in *Proc. of the International Symposium on Music Information Retrieval*, pp. 231–236, 2008.

[22] D. Turnbull, R. Liu, L. Barrington, and G. Lanckriet, "A game-based approach for collecting semantic annotations of music," in *Proc. of ISMIR*, (Vienna, Austria), 2007.

[23] E. L. M. Law, L. v. Ahn, R. Dannenberg, and M. Crawford, "TagATune: A game for music and sound annotation," in *Proc. of ISMIR*, (Vienna, Austria), 2007.

[24] M. I. Mandel and D. P. W. Ellis, "A web-based game for collecting music metadata," in *Journal of New Music Research*, vol. 37, pp. 151–165, Taylor & Francis, 2008.

[25] C. Harte, M. Sandler, S. Abdallah, and E. Gomez, "Symbolic representation of musical chords: A proposed syntax for text annotations," in *Proc. of ISMIR*, (London, UK), pp. 66–71, 2005.

[26] 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. of ISMIR*, (Miami, USA), 2011.

[27] K. B. Cohen, L. Fox, P. V. Ogren, and L. Hunter, "Corpus design for biomedical natural language processing," in *Proc. of the ACL-ISMB workshop on linking biological literature, ontologies and databases: mining biological semantics*, pp. 38–45, 2005.

[28] G. Leech, *Developing Linguistic Corpora: a Guide to Good Practice*, ch. Adding Linguistic Annotation, pp. 17–29. Oxford: Oxbow Books, 2005.

[29] K. Fort and V. Claveau, "Annotating football matches: Influence of the source medium on manual annotation," in *Proc. of LREC*, (Istanbul, Turkey), May 2012.

[30] E. Peiszer, T. Lidy, and A. Rauber, "Automatic audio segmentation: Segment boundary and structure detection in popular music," in *Proc. of LSAS (Learning the Semantics of Audio Signals)*, (Paris, France), 2008.

[31] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[32] K. Fort, C. François, O. Galibert, and M. Ghribi, "Analyzing the impact of prevalence on the evaluation of a manual annotation campaign," in *Proc. of LREC*, (Istanbul, Turkey), May 2012.

[33] U. Gut and P. S. Bayerl, "Measuring the reliability of manual annotations of speech corpora," in *Proc. of Speech Prosody*, (Nara, Japan), pp. 565–568, 2004.

[34] H. Voormann and U. Gut, "Agile corpus creation," *Corpus Linguistics and Linguistic Theory*, vol. 4(2), pp. 235–251, 2008.

[35] D. Kaplan, R. Iida, and T. Tokunaga, "Annotation process management revisited," in *Proc. of LREC*, pp. 365 – 366, May 2010.

[36] MPEG-7, "Information technology - multimedia content description interface - part 5: Multimedia description scheme," 2002.

[14] It could not be included due to space constraints.

MODELING MUSICAL MOOD FROM AUDIO FEATURES AND LISTENING CONTEXT ON AN IN-SITU DATA SET

Diane Watson
University of Saskatchewan
diane.watson@usask.ca

Regan L. Mandryk
University of Saskatchewan
regan.mandryk@usask.ca

ABSTRACT

Real-life listening experiences contain a wide range of music types and genres. We create the first model of musical mood using a data set gathered in-situ during a user's daily life. We show that while audio features, song lyrics and socially created tags can be used to successfully model musical mood with classification accuracies greater than chance, adding contextual information such as the listener's affective state or listening context can improve classification accuracy. We successfully classify musical arousal with a classification accuracy of 67% and musical valence with an accuracy of 75% when using both musical features and listening context.

1. INTRODUCTION

Musical mood – the emotion expressed by a piece of music – is conveyed to a listener through a variety of musical cues in the form of auditory features. These auditory features (e.g., mode, rhythm, articulation, intensity and timbre of a musical track) have previously been used to model musical mood with fairly good classification results [1-2]. However, current high performing models of musical mood have two main problems: first, music is constrained to a single genre (usually Western classical or Western popular); and second, the data is collected and labeled in laboratory contexts. Previous work has shown that the data sets used in previous research modeling musical mood do not correspond to real-life listening experiences in a number of ways [3]. First, people listen to music of various genres in their daily life; second, music is listened to as part of social activities or in a public venue; third, music is attended to as a secondary activity while driving, working, or exercising; finally, people listen to music for various reasons, such as to relax, to entertain, or to influence emotion [3].

Previous high-performing musical mood models, based on data from a single genre and gathered in a laboratory setting may fail when applied to data sets gathered in daily life. Systems implementing these previous models – such as music recommender systems – may also fail when using data collected from real-life listening experiences, which may lead to negative user experiences. However, musical mood classifiers built on a broad data set, containing several genres and labeled during real-life activities rather than in a laboratory, may be unusable in real systems if they yield weak classification results.

To solve the problem of building good musical mood classifiers that are effective for real-life listening experiences, we include context-sensitive features in addition to the previously used auditory features. Our data set of real-life listening experiences was gathered through an experience-sampling study using smartphones. Participants were handed phones for a period of two weeks. Phones would randomly poll the user about once per hour and ask them to fill out a survey collecting musical mood, the listener's affective state and the context of the listening experience. Genre, title and artist were optionally captured. Previous analysis of our data set shows that real-life listening experiences are far from the homogenous data sets used in current models, and cover a wide range of genres, artists, and songs [3]. In the present paper, we used our naturally-gathered data set to model musical mood using Bayesian networks and feature sets including musical features (audio features, song lyrics, socially-created tags), the affective state of the listener, and listening context. Listening context included reason for listening, activity, location, social company, level of choice over the song and mental associations.

In this paper we make two main contributions. First, we successfully model musical mood from a data set gathered in-situ during a user's daily life; we are the first to do so. Second, we show that while musical features (audio features, song lyrics and socially created tags) can successfully model musical mood with classification accuracies better than chance, adding contextual information, such as the listener's affective state or the listening context of the musical experience, can further improve classification accuracies. We successfully classify musical arousal with a classification accuracy of 67% and musical valence with an accuracy of 75% when using both musical features and listening context.

2. RELATED WORK

2.1 Affective State

It is well documented that music can induce specific affective experiences in the listener. Affective state, or the emotion or mood a person is experiencing, can be described using either a categorical or dimensional approach. The categorical approach breaks emotions into discrete labeled categories (e.g., happiness, fear, joy) [4]. In contrast, the dimensional approach, which we use in this paper, represents affective state using two orthogonal dimensions: arousal and valence [5]. Arousal can be described as the energy or activation of an emotion. Low arousal corresponds to feeling sleepy or sluggish while high arousal corresponds to feeling frantic or excited. Valence describes how positive or negative an emotion is. Low valence corresponds to feeling negative, sad or melancholic and high valence to feeling positive, happy or joyful. Most categorical emotions can be described by Arousal-Valence (A-V) space (e.g., angry in Figure 1).

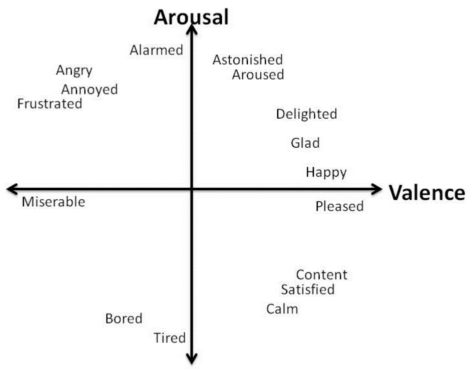


Figure 1 shows A-V space labeled with several of the categorical emotions.

2.2 Musical Mood

Musical mood, the emotion expressed by a piece of music, is to some degree perceived consistently across different listeners and even different cultures. Studies by Juslin and Sloboda have shown that listeners of different musical training classify musical mood into the same categories [6]. Fritz et al. found that the Mafa natives of Africa – without any exposure to Western music – categorized music into the same three basic emotional categories as Westerners [7]. Musical mood is frequently measured in arousal and valence [8] and we have used this approach in this paper. It should be noted that the affective state induced in the listener is not necessarily the same as the musical mood of the music [9], [10]. For example, an individual who is herself feeling frustrated (i.e., mood of the listener) can still perceive a piece of music as calm (i.e., musical mood).

2.3 Musical Mood Classification

Musical mood can be manually categorized by the listener, but researchers have also algorithmically classified musical mood using audio features extracted from the musical track. Work by Juslin [11] has identified seven musical features that are important in the interpretation of musical mood. He asked performers to play the same musical scores in such a way as to express four different musical moods (anger, sadness, happiness and fear) and then had listeners rate the strength of each mood. He found that performers and listeners used the same features to identify each mood, but weighted their importance differently. These features are:

- *Mode:* Mode refers to the key of the music. (e.g. A-)
- *Rhythm:* Rhythm is the pattern of strong and weak beat. It can be described through speed (tempo), strength, and regularity of the beat.
- *Articulation:* Articulation refers to the transition and continuity of the music. It ranges from legato (connected notes) to staccato (short abrupt notes).
- *Intensity / Loudness:* Intensity is a measure of changes in volume.
- *Timbre / Spectrum:* Timbre describes the quality of the sound. It is often defined in terms of features of the spectrum gathered from the audio signal.

Musical mood has previously been modeled using only audio features. Lu et al. classified classical music into the four quadrants of A-V space using a collection of audio features with an accuracy of 86.3% [1]. Their algorithm also detected places within the song where the mood changed. Experts labeled musical mood. Feng et al. classified Western popular music into four moods using only two features: tempo and articulation. They achieved a precision of 67% and a recall of 66% [2]. They do not specify how they gathered musical mood.

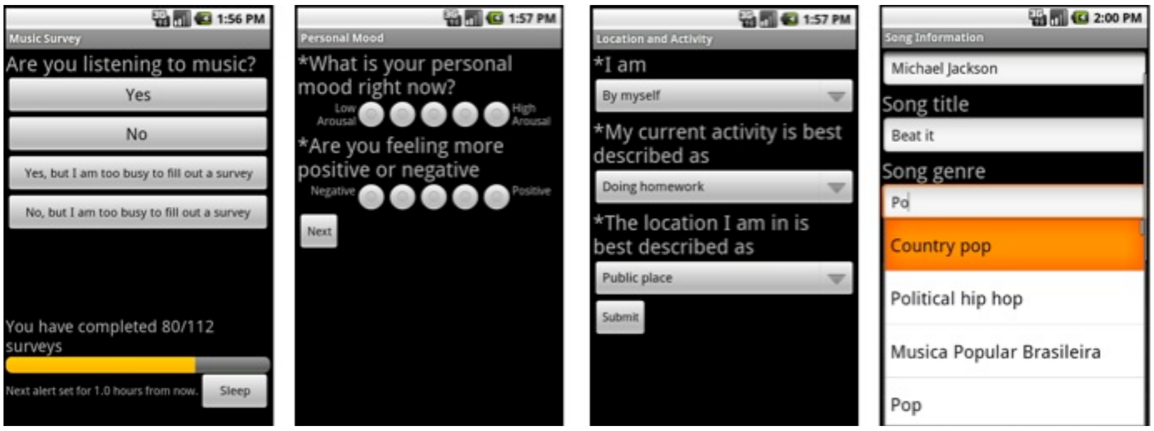
Some effort has been made to incorporate other musical context with audio features to improve classification. Yang et al., working with a set of Western Rock music, made small gains in their classification rates by adding lyrics to the audio features (from 80.7% to 82.8%) [12]. Musical mood was gathered in a laboratory study. Bischoff et al. integrated socially created tags with audio features, and while their classification rates were low due to problems with their ground truth data, they achieved better results using tags and audio features than audio features alone [13]. Their poor results may be due to the fact they were using a diverse, online, data set with multiple genres. Musical mood was specified in this data set by users of the AllMusic site.

2.4 Music Recommenders

Many commercial music recommender systems exist (e.g., Last.fm, Pandora, Apple's Genius, StereoMoods). In 2010, Han et al. created COMUS, a context-based music recommender system that accounts for mood, situation and musical features [14]. Their system was limited to recommending music for only one listening purpose – to transition between emotional states – and assumed a prior explicit knowledge about how a specific individual changes their music habits depending on situation.

3. EXPERIENCE SAMPLING SOFTWARE

To gather an in-situ data set of musical mood and listening context, we surveyed participants using an experience-sample methodology [15]. We created an application that ran on Android 2.1 smartphones, which generated custom surveys from XML files. Participants were asked to carry the phone with them at all times. While it would be possible to create a plug-in for an existing computer media player such as iTunes, we wanted to capture listening experiences in all contexts. For example, some activities, such as exercising, do not usually occur simultaneously with computer use. Participants were not required to use the phone as a media player as this would



Are you listening to music?

You have completed 80/112 surveys

Next alert set for 1.0 hours from now.

*What is your personal mood right now?

*Are you feeling more positive or negative

*I am: By myself

*My current activity is best described as: Doing homework

*The location I am in is best described as: Public place

Michael Jackson

Song title: Beat it

Song genre: Po

Figure 2 shows screenshots of the experience-sampling software. Participants answered a short survey about their affective state, listening context and the music they were listening too.

further limit listening contexts (e.g., music playing in the background at a restaurant). The tradeoff is that we could not automatically capture song title, artist, or audio features such as tempo.

The program would query the user randomly (approximately hourly) by vibrating the phone. A participant could fill out a survey or dismiss the program by indicating they were too busy. Surveys were completed in less than five minutes and were filled out regardless of whether participants were listening to music. This was done to encourage survey completion. Participants were paid per number of surveys completed, between 5 and 40 CAD. To obtain the maximum payout, 112 surveys were required, which is roughly 8 surveys per day. A progress bar in the software provided feedback about how many surveys had been completed.

Four types of information were collected: musical mood, affective state, musical context and listening context. See Figure 2 for screenshots of the experience-sampling application.

Musical Mood: Participants were asked to describe the musical mood of the song they were listening to using two five-point differential scales. They were asked to rate the arousal of the music by selecting one of five radio buttons between low arousal and high arousal. Similarly, they rated the valence of the music on a scale between sad and happy. Definitions were given to participants before the study and available from a help menu.

Affective State: Participants were asked to describe their personal arousal and valence using five-point differential scales similar to musical mood.

Artist, Title and Genre: Artist and title could optionally be entered in free-text fields that autocompleted to previously entered answers. A genre field was provided that autocompleted to a list of common genres taken from Wikipedia, but also allowed participants to enter their own genre.

Listening Context: Participants were asked questions describing their current listening context. Participants selected their current activity from a list (waking up, bathing, exercising, working, doing homework, relaxing, eating, socializing, romantic activities, reading, going to sleep, driving, travelling as a passenger, shopping, dancing, getting drunk, other). These activities were taken from [8], which lists the most common activities to occur in conjunction with music. Participants also selected their location (home, work, public place, other) and social company (by myself, with people I know, with people I do not know). Participants selected their reason for listening (to express or release emotion, to influence my emotion, to relax, for enjoyment, as background sound, other) as well as whether or not they choose the song (yes, yes as part of a playlist, no). A text field was provided for participants to enter any terms or phrases they associated with the song.

4. DATA SET GATHERED IN-SITU

Twenty participants, (14 male) with an average age of 25, used the experience-sampling software for two weeks.

For a full description of the data set, see [3]. Here we summarize for the purposes of guiding the development of our musical mood classifiers. In total 1803 surveys were filled out; 610 of those surveys were completed when the participant was listening to music. Only the results of the music surveys are included in this paper.

Participants had an average arousal of 2.28 (SD=0.92) on our 5-pt scale (0 low, 2 neutral, 4 high) and average valence of 2.64 (SD=0.90). The music they were listening to had an average arousal of 2.64 (SD=1.05) and average valence of 2.66 (SD=1.14).

The most common activities while listening to music were working (37%) and relaxing (21%). Users also listened to music while eating (6%), driving (5%), travelling (as a passenger)(5%), other (5%), and socializing (4%). Participants were by themselves 57% of the time, with people they knew 37% and with people they did not know 6%. They were at work 39% of the time, at home 38%, in a public place 21% and in other locations 2%.

The most common reason for listening was to use the music as background sound (46%) or enjoyment (25%). Participants chose the song 74% of the time; 50% of the time it was as part of a playlist.

Participants entered 102 unique song genres a total of 486 times. Genres were coded into their parent genre and the most common genres were pop (28%), rock (23%), electronic (14%), jazz (7%), hip-hop & rap (6%), other (5%), modern folk (4%) and country (3%). The remaining genres were classical, traditional/indigenous music, soundtrack, blues, easy listening and R&B.

Participants entered musical associations for 335 songs. These were then coded into themes, from a list partially taken from [8]. Participants mostly described emotions (45%), lyrics or instruments (20%), imagery (15%), or specific people, locations or memories (7%).

Songs were not limited to Western genres or even the English language. At least 14% of the songs with artist and title specified were non-English; however, all participants listened to at least some English music.

5. CLASSIFICATION FEATURES

To create classifiers of musical mood, we included musical features used in previous work, but also added context-based features from our data set.

5.1 Musical Features

Songs were downloaded from iTunes and other sources where possible using the artist and title specified.

5.1.1 Audio Features

Audio features describing the mode, rhythm, articulation, and timbre of the music were extracted using MIRtoolbox [16] and Matlab.

Mode: These features included the most probable key of the music as well as an estimation of whether the key was major or minor.

Rhythm: These features included an estimation of tempo (number of beats per minute) and pulse clarity (relative strength of the beat, related to how easily a listener can perceive the tempo[17]).

Articulation: These features included the attack slope, (an indicator of how aggressively a note is played) as well as the Average Silence Ratio (ASR)[2].

Timbre: These features were taken from the audio spectrum and include brightness (amount of energy above a cutoff point in the spectrum), rolloff (the frequency such that 85% of total energy is contained below that frequency), spectral flux (average distance between the spectrum of successive frames), spectral centroid (the frequency around which the spectrum is centered), MFCC (description of the sound separated into different bands), low energy (percentage of frames with less than average energy), and average sensory roughness. Sensory roughness corresponds to when several sounds of nearly the same frequency are heard, causing a “beating” phenomenon. High roughness corresponds to harsher music with more “beating” oscillations.

5.1.2 Lyrics

Lyrics were downloaded from various sources using the artist and title. Some included mark-ups indicating non-word sounds or names of singers responsible for a section of lyrics. Only English lyrics were collected. Songs that were mainly English but included a few foreign words were included. Some songs contained notations indicating that a section was repeated (e.g., “x2”). These were manually removed and replaced with the repeated text. Lyrics were analyzed using the Linguistic Inquiry Word Count Tool (LIWC) [18], a textual analysis tool that provides a word count in 80 categories and the output of LIWC was used as the feature set.

5.1.3 Tags

Socially created tags from the website Last.fm were downloaded and analyzed using LIWC. This output was used as features.

5.2 Affective Features

This included the personal arousal and valence of the listener on a 5-point scale.

5.3 Listening Context

Listening context included: reason for listening, activity, location, social company, and level of choice over the song. The associations were categorized (see section 4) and this category was included as a feature. Associations were also analyzed using LIWC.

6. MODEL RESULTS

6.1 Feature Sets

We used a number of feature combinations in creating our models, which can be summarized as three feature sets.

Musical Features: Our first feature set used audio features, lyrical features, and tag features, as these features were used in previous models based on laboratory-gathered data sets of a single genre. There were 198 different features in this set.

Musical Features + Affective Features: Our second feature set used all the musical features but added personal arousal and valence for a total of 200 different features.

Musical Features + Listening Context: Our third feature set combined musical features with the listening context collected in our study for a total of 296 features.

6.2 Models of Musical Mood

Due to “in the wild” nature of the study, musical arousal and musical valence had an uneven distribution of responses. Participants were much more likely to indicate that they were listening to songs with high arousal and high valence. To prevent over fitting of the model due to class skew, musical arousal and musical valence were binned into two levels, low, and high. Neutral instances were ignored. Since only songs with song titles could be downloaded and audio features extracted, instances without a song title were also ignored. Also, undersampling, a technique that selects a random number of instances to obtain an equal distribution, was used. This lowered the total number of instances from 610 to 122 when modeling musical arousal and 156 when modeling musical valence. To avoid any effects caused by the specific set of random instances chosen, this process was completed five times, and the average accuracies of all runs are reported.

All models were created in Weka [19] using Bayes Net classifiers, Markov Estimation and tenfold cross validation. We modeled musical arousal and musical valence separately, using each feature set. See Figure 3 for classification accuracies.

Musical Features: Using only musical features (audio features, lyrics and tags), musical arousal has a classification accuracy of 59.5% (SD=3.1, kappa=0.1984). Musical valence has an accuracy of 53.0% (SD=6.8, kappa=0.0604). While both models are higher than chance (50%), a one sample t-test shows that only musical arousal (t_4=6.74, p<0.01) was significantly higher. However, the results are lower than previously reported classification accuracies of homogenous lab-based data sets.

Musical Features + Affective State: When we combined affective features (personal arousal and valence) with musical features, musical arousal has a classification accuracy of 60.3% (SD=4.6, kappa=0.2121). Musical valence has an accuracy of 60.2% (SD=4.6, kappa=0.2074). Both musical arousal (t_4=4.99 p<0.01) and musical valence (t_4=4.70, p<0.01) performed significantly better than chance (50%), and we achieved improved classification accuracies of musical arousal and musical valence by using a combination of affective and musical features.

Musical Features + Listening Context: When we combined musical features with listening context features, musical arousal has a classification accuracy of 67.4% (SD=1.7, kappa=0.3437). Musical valence has an accuracy of 75.7% (SD=1.5, kappa=0.5133). Both musical arousal (t_4=23.54, p <0.0001) and musical valence (t_4=38.75, p<0.0001) performed significantly better than chance (50%), and we achieved gains in classification accuracy in both models over using only musical features or musical features and affective features combined.

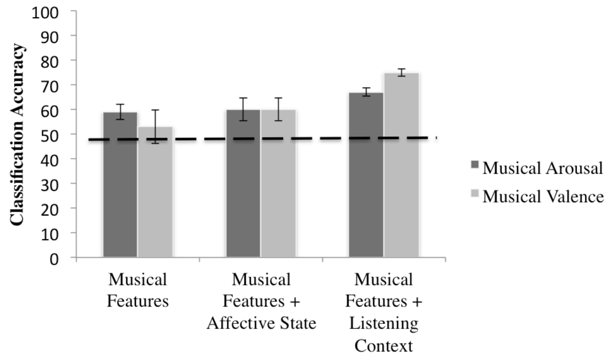


Figure 3 shows the classification accuracies for each feature set. The dotted line shows chance (50%).

7. DISCUSSION

Our experience sampling study collected in-situ data that reflects real-life listening experiences. Unlike previous models, our data included multiple genres and different listening contexts.

We have shown that musical mood can be successfully modeled from in-situ data, although with a lower classification accuracy than previous attempts. Adding affective state to the model resulted in an improvement in classification accuracy while modeling musical valence; adding listening context to the model resulted in improvements in both musical arousal and musical valence. Our results show that listening context is an important aspect of modeling musical mood, when using real-life data.

7.1 Importance of Context

It may be possible that context is important when modeling musical mood because participants rate musical mood differently depending on their context. For example, a user may rate the same song differently depending on whether they are working alone or cooking with friends. We cannot confirm this with our data set, as one would need the same songs played in a variety of listening contexts – in our study, songs and artists were only encountered once on average.

It is also possible that people listen to music with certain musical moods based on their context. For example, a user may generally choose to listen to music with high arousal when exercising and low arousal when eating dinner. In that case our model predicts the type of musical mood listeners want to listen to, based on context, which is useful for automatically generating playlists.

Similarly, participants may rate musical mood differently depending on their affective state. This is a tricky relationship to investigate as the music itself has a hand in inducing an affective state in a listener. Any correlation found between musical mood and affective state does not show directionality of the relationship.

To examine the relationships between listening context, musical mood, and affective state, we could provide users with representative samples in a music library. By listening to (and rating) the same song in a variety of contexts and affective states, the relationship between these three factors might be made clear.

7.2 Limitations

There are several limitations with our study. The first is that participants are unlikely to answer a survey during some activities (e.g., driving). Second, all categories in our data may not be mutually exclusive (e.g., reading while running on the treadmill). Third, the number of participants and length of the study may have been too small to collect a fully representative sample of listening context. Finally, previous studies have assumed that people listen to music with four emotional categories (happy, sad, fear, anger) [11]; however, in our study we found that people tended to listen to happy music. The other three emotions may not be equally represented when capturing in-situ data [3].

While a classification accuracy of 75% is much improved over a random classifier, or one based on auditory features, a music recommender suggesting songs with the wrong mood a quarter of the time may result in a negative user experience. This can be circumvented in a few ways. First a music recommender can select tracks from a personal music library; users are more likely to enjoy their own music even if the recommendation is off. Second, a playlist rather than a single song could be recommended so that a majority of the music recommended is suitable. Third, combining this model with existing recommendation systems that use clustering of similar genres and artists could further improve existing prediction rates. Finally, we could improve the classification rates and avoid possible overfitting caused by the small number of instances in our models by collecting a more comprehensive data set.

8. FUTURE WORK

Based on the results of this work, we will create a context-aware music recommender system. This system will take in the context of the listening experience and use this context to compile a playlist. Based on our models, the system will recommend a musical mood listeners are likely to enjoy, and will create playlists of songs with this specific musical mood, (based on a data set labeled from musical features). The system could also make suggestions of songs for purchase the user might enjoy. We will evaluate the predictions through a user study, conducted in-situ, to preserve the importance of context.

To create the underlying model for this music recommender, a larger in-situ data set will be collected. The study will run for a longer time period (i.e., months) with a larger pool of participants. Participants will receive bonuses for filling out genre, title and artist and will be asked to provide a copy of their music library at the end of the study for audio feature processing. This larger, more comprehensive data set will help improve classification accuracies.

9. CONCLUSIONS

We successfully model musical mood from a data set gathered in-situ during a user's daily life. We show that musical features (audio features, song lyrics and socially created tags) can successful model musical mood with classification accuracies better than chance. We successfully classify musical arousal with a classification accuracy of 59% and musical valence with an accuracy of 53% when using only musical features on an in-situ data set.

Adding contextual information, such as the listener's affective state or the listening context of the musical experience can further improve classification accuracies. We successfully classify musical arousal and musical valence with a classification accuracy of 60% when using both musical features and affective state. We classify musical arousal with a classification accuracy of 67% and musical valence with an accuracy of 75% when using both musical features and listening context.

10. ACKNOWLEDGEMENTS

This work was funded in part by the GRAND NCE.

11. REFERENCES

[1] L. Lu, D. Liu, and H.J. Zhang, "Automatic mood detection and tracking of music audio signals," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 1, pp. 5 – 18, Jan. 2006.

[2] Y. Feng, Y. Zhuang, and Y. Pan, "Popular music retrieval by detecting mood," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, 2003, pp. 375–376.

[3] D. Watson and R. L. Mandryk, "An In-Situ Study of Real-Life Listening Context," in *SMC 2012*.

[4] P. Ekman, *Basic Emotion*. John Wiley & Sons, Ltd., 2005.

[5] J. A. Russell, "Core affect and the psychological construction of emotion," *Psychological Review*, vol. 110, no. 1, pp. 145–172+, 2003.

[6] P. Juslin and J. Sloboda, *Music and Emotion: Theory and Research*. Oxford University Press, 2001.

[7] T. Fritz, S. Jentschke, N. Gosselin, D. Sammler, I. Peretz, R. Turner, A. D. Friederici, and S. Koelsch, "Universal Recognition of Three Basic Emotions in Music," *Current Biology*, vol. 19, no. 7, pp. 573 – 576, 2009.

[8] P. Juslin and P. Laukka, "Expression, Perception, and Induction of Musical Emotions: A Review and a Questionnaire Study of Everyday Listening - Journal of New Music Research," *Journal Of New Music Research*, vol. 33, no. 3, pp. 217–238, 2004.

[9] K. . Scherer and M. R. Zentner, "Emotional Effects Of Music: Production Rules," in *Music and Emotion:Theory and Research*, New York, NY, USA: Oxford University Press, 2001, pp. 361–392.

[10] M. R. Zentner, S. Meylan, and K. Scherer, "Exploring 'musical emotions' across five genres of music.," in ICMPC, Keeyle, UK, 2000.

[11] P. N. Juslin, "Cue utilization in communication of emotion in music performance: Relating performance to perception.," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 26, no. 6, pp. 1797–1812, 2000.

[12] D. Yang and W. Lee, "Disambiguating Music Emotion Using Software Agents," in *ISMIR* Barcelona

[13] K. Bischoff, C. Firan, R. Paiu, W. Nejdl, C. Laurier, and M. Sordo, "Music Mood and Theme Classification-a Hybrid Approach," in *ISMIR*, Kobe, Japan, 2009.

[14] B.-J. Han, S. Rho, S. Jun, and E. Hwang, "Music emotion classification and context-based music recommendation," *Multimedia Tools Appl.*, vol. 47, pp. 433–460, May 2010.

[15] R. Larson and M. Csikszentmihalyi, "The Experience Sampling Method.," *New Directions for Methodology of Social & Behavioral Science*, vol. 15, pp. 41–56, 1983.

[16] O. Lartillot and P. Toiviainen, "A Matlab Toolbox for Musical Feature Extraction from Audio," in *Proceedings of the 10th International Conference on Digital Audio Effects*, Bordeaux, France, 2007.

[17] O. Lartillot, T. Eerola, P. Toiviainen, and J. Fornari, "Multi-Feature Modeling of Pulse Clarity: Design Validation and Optimization," presented at the ISMIR 2008, 2008.

[18] J. W. Pennebaker, M. E. Francis, and R. J. Booth, "Linguistic Inquiry and Word Count LIWC 2001," *Word Journal Of The International Linguistic Association*, pp. 1–21, 2001.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations*, vol. 11, no. 1, 2009.

ANALYZING DRUM PATTERNS USING CONDITIONAL DEEP BELIEF NETWORKS

Eric Battenberg
University of California, Berkeley
Dept. of Electrical Engineering and Computer Sciences
ericb@eecs.berkeley.edu

David Wessel
University of California, Berkeley
Center for New Music and Audio Technologies
wessel@cnmat.berkeley.edu

ABSTRACT

We present a system for the high-level analysis of beat-synchronous drum patterns to be used as part of a comprehensive rhythmic understanding system. We use a multi-layer neural network, which is greedily pre-trained layer-by-layer using restriced Boltzmann machines (RBMs), in order to model the contextual time-sequence information of a drum pattern. For the input layer of the network, we use a conditional RBM, which has been shown to be an effective generative model of multi-dimensional sequences. Subsequent layers of the neural network can be pre-trained as conditional or standard RBMs in order to learn higher-level rhythmic features. We show that this model can be fine-tuned in a discriminative manner to make accurate predictions about beat-measure alignment. The model generalizes well to multiple rhythmic styles due to the distributed state-space of the multi-layer neural network. In addition, the outputs of the discriminative network can serve as posterior probabilities over beat-alignment labels. These posterior probabilities can be used for Viterbi decoding in a hidden Markov model in order to maintain temporal continuity of the predicted information.

1. INTRODUCTION

Deep belief networks (DBNs) have shown promise in many discriminative tasks, such as written digit recognition [6] and speech recognition [8]. In addition, the generative nature of DBNs makes them especially well-suited for stochastic generation of images or sequences [5, 11].

In this paper, we apply DBNs to the analysis of drum patterns. The drum pattern analysis system presented here is to be part of a complete live drum understanding system, which is also composed of a drum detection front-end [1] and a low-level multi-hypothesis beat tracker. The goal of the drum understanding system is to go beyond simple beat tracking by providing additional high-level rhythmic information, such as time signature or style information, while being robust to expressive embellishments and dynamic song structure, such as tempo fluctuations or time signature changes. The pattern analysis system we present here can help achieve these goals, not only by providing the desired high-level information, but also by communicating with the low-level beat tracker to help it correct beat period and phase errors.

To demonstrate the effectiveness of this model, we focus on a specific discriminative task: identifying the alignment of beats within a measure. Measure alignment information is particularly important to high-level drum pattern analysis since each beat has a specific meaning depending upon the musical style. For example, song transitions typically occur on the first beat of a measure, and in rock music and relate styles, beats 2 and 4 typically feature an accented snare drum *back beat*.

Previous work on beat-measure alignment has focused on simple heuristic rules. In [7], Klapuri presents a beat tracker that determines beat-measure alignment by correlating multi-band onset patterns with two different back beat measure templates. In [3], Goto addresses beat alignment by detecting chord change locations and by alignment with 8 drum pattern templates. Approaches like these work well for the typical pop song but are ineffective when presented with exotic rhythm styles or many types of progressive music. To deal with these situations, rather than accruing a large list of hand-written heuristic rules, we can automatically encode a large amount of musical knowledge into the *distributed state-space* [2] of a deep belief network, which we introduce in the next section.

2. DEEP BELIEF NETWORKS

2.1 The Restricted Boltzmann Machine

The deep belief network is a probabilistic multi-layer neural network composed of *restricted Boltzmann machines*, or RBMs [2, 5]. The RBM, as shown in Figure 1, is a two layer probabilistic graphical model with undirected connections between visible layer units, v_i, and hidden layer units, h_j. The "restricted" part of the name points to the fact that there are no connections between units in the same layer. This allows the conditional distribution of the units of one layer given all the units of the other layer to be com-

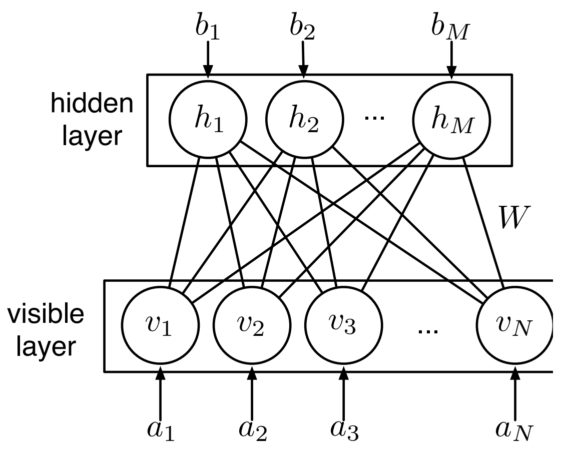


Figure 1. A restricted Boltzmann machine with N vis units and M hidden units.

pletely factorial, i.e.

$$P(\mathbf{v}|\mathbf{h}) = \prod_i P(v_i|\mathbf{h})$$
$$P(\mathbf{h}|\mathbf{v}) = \prod_j P(h_j|\mathbf{v})$$

The RBM is a probabilistic energy-based model, meai the probability of a specific configuration of the visible hidden units is proportional to the negative exponentia of an energy function, $E(\mathbf{v},\mathbf{h})$

$$P(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z}$$

Where $Z = \sum_{\mathbf{v},\mathbf{h}} \exp(-E(\mathbf{v},\mathbf{h}))$ is a normalizing (stant referred to as the *partition function*. Note that cause Z is difficult to compute, it is typically intractabl compute the joint distribution $P(\mathbf{v},\mathbf{h})$.

For binary-valued visible and hidden units, the en function, $E(\mathbf{v},\mathbf{h})$, can be written as:

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{a}^{\mathrm{T}}\mathbf{v} - \mathbf{b}^{\mathrm{T}}\mathbf{h} - \mathbf{v}^{\mathrm{T}}W\mathbf{h}$$

Where $\mathbf{a}$ and $\mathbf{b}$ are vectors containing the visible and hidden unit biases, respectively, and W is the weight matrix that connects the two layers.

The goal in training an RBM is to maximize the likelihood of the training data under the model, $P(\mathbf{v})$. The actual log-likelihood gradient is difficult to compute because it involves the intractable partition function Z; however, stochastic estimates of the gradient can be made by drawing Gibbs samples from the joint distribution $P(\mathbf{v},\mathbf{h})$ using the factorial conditional distributions in (5),(6).

$$P(v_i = 1|\mathbf{h}) = \bar{\sigma}(a_i + \textstyle\sum_j W_{ij}h_j) \quad (5)$$
$$P(h_j = 1|\mathbf{v}) = \bar{\sigma}(b_j + \textstyle\sum_i W_{ij}v_i) \quad (6)$$

Where $\bar{\sigma}(x)$ is the logistic sigmoid function:

$$\bar{\sigma}(x) = \frac{1}{1+e^{-x}} \quad (7)$$

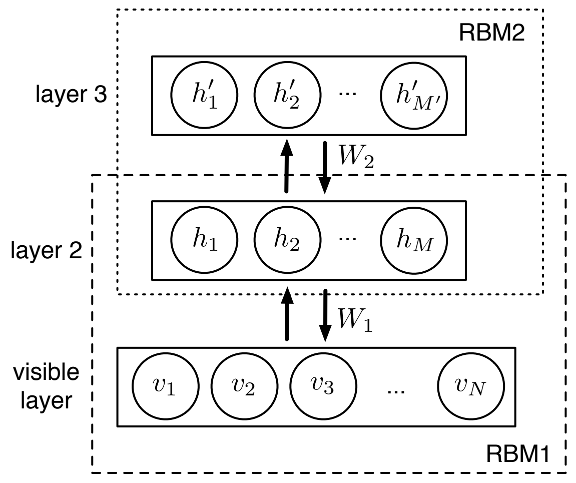


Figure 2. A 3-layer deep belief network comprised of 2 RBMs

The Gibbs sampling Markov chain can take quite a long time to produce actual samples from the joint distribution, so in practice the chain is started at a training example and run for a small number of iterations. Using this estimate of the log-likelihood gradient, we are instead minimizing a quantity referred to as the *contrastive divergence* between the training data and the model [2, 5]. Contrastive divergence updates for the RBM parameters are shown below:

$$\Delta W_{ij} \propto \langle v_i h_j \rangle_0 - \langle v_i h_j \rangle_k \quad (8)$$
$$\Delta a_i \propto \langle v_i \rangle_0 - \langle v_i \rangle_k \quad (9)$$
$$\Delta b_j \propto \langle h_j \rangle_0 - \langle h_j \rangle_k \quad (10)$$

Where $\langle\cdot\rangle_k$ denotes the value of the quantity after k iterations of Gibbs sampling, and for $k = 0$, v_i is simply the training data and h_j is a sample from (6) given the training data. Typically, these updates are performed using multiple training examples at a time by averaging over the updates produced by each example. This helps to smooth the learning signal and also helps take advantage of the efficiency of larger matrix operations. As $k \to \infty$, these updates approach maximum likelihood learning.

2.2 Stacking RBMs

A deep belief network is formed when multiple RBMs are stacked on top of each other as shown in Figure 2. After training a first-level RBM using the training data, we can perform a deterministic up-pass by setting the hidden units to their real-valued activation probabilities using (6) for each visible training vector. This is the same as what is done in the up-pass in a deterministic neural network. These deterministic hidden unit values are then used as the visible data in a subsequent higher-level RBM, which is also trained using contrastive divergence learning. This RBM stacking continues until the network reaches the desired depth. This greedy layer-by-layer training approach is a useful procedure for learning a set of non-linear features in an unsupervised manner [4], and it has been shown

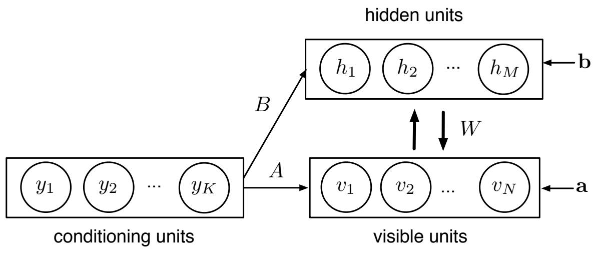


Figure 3. A conditional restricted Boltzmann machine. The correct label unit activations are provided as part of the visible unit data during training.

to be a beneficial pre-training procedure when followed by discriminative backpropagation [6].

2.3 The Conditional Restricted Boltzmann Machine

The conditional restricted Boltzmann machine (CRBM) takes the RBM a step further by adding directed connections between additional visible units, y_i, and the existing visible and hidden units, as shown in Figure 3. These additional units can represent any type of additional information, including visible data from the recent past. Because of this, the CRBM is an effective generative model of time sequence data [11]. This fact is what motivated our use of the CRBM to model drum patterns.

The directed connections, which are represented by weight matrices A and B, replace the bias terms, $\mathbf{a}$ and $\mathbf{b}$ in (5),(6), with dynamic bias terms, $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$.

$$\hat{\mathbf{a}} = \mathbf{a} + A\mathbf{y} \quad (11)$$
$$\hat{\mathbf{b}} = \mathbf{b} + B\mathbf{y} \quad (12)$$

Where $\mathbf{y}$ is a vector containing the conditioning data. This modified RBM models the distribution $P(\mathbf{v}, \mathbf{h}|\mathbf{y})$, and the learning rules in (8)–(10) are unchanged except for the addition of the dynamic bias terms to the sampling expressions. The learning rules for the conditional weight matrices also have a familiar form:

$$\Delta A_{ij} \propto \langle v_i y_j \rangle_0 - \langle v_i y_j \rangle_k \quad (13)$$
$$\Delta B_{ij} \propto \langle h_i y_j \rangle_0 - \langle h_i y_j \rangle_k \quad (14)$$

Note that the y_j above are simply the training values and are not stochastically sampled in any way.

3. MODELLING AND ANALYZING DRUM PATTERNS

3.1 Bounded Linear Units

Drum patterns are not simply a series of ones and zeros, onset or no onset. Most drum patterns contain an appreciable sonic difference between accented and unaccented notes on every drum or cymbal, and it is these differences which give drum patterns their character. In order to effectively model drum patterns using the CRBM, we must modify the binary-valued visible units to be real-valued.

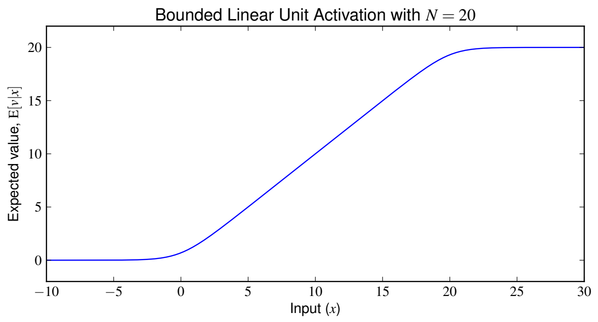


Figure 4. Bounded linear unit activation function

There are many options for getting real-valued visible activations out of RBMs; in fact, it has been shown that every distribution in the exponential family is a viable candidate [13]. A popular choice is the Gaussian distribution due to its familiarity and ubiquity; however, the unboundedness of Gaussian samples does not translate well to the space of dynamic levels possible within a drum pattern.

In order to model the bounded nature of playing dynamics, we use a modified version of the *rectified linear units* (RLUs) described in [9]. RLUs are constructed from a series of binary units with identical inputs but with fixed, increasing bias offsets. If the bias offsets are chosen appropriately, the expected value and variance of the number of active units out of these N tied binary units with common input x is:

$$\mathrm{E}[v|x] = \log(1 + e^x) - \log(1 + e^{x-N}) \quad (15)$$
$$\mathrm{Var}(v|x) = \bar{\sigma}(x) - \bar{\sigma}(x - N) \quad (16)$$

As can be seen in Figure 4, (15) yields a sigmoidal curve that saturates when $x > N$, bottoms out when $x < 0$, and is approximately linear in between. In the linear region, the variance is equal to one, so the value of N is chosen to achieve the desired level of noisiness in the samples, and the training data can be rescaled to the interval $[0, N]$. In this work, we have chosen $N = 20$, so that a value of 20 represents the loudest possible drum strike, while zero represents the absence of a drum strike. To sample from these *bounded linear units* (BLUs), instead of actually sampling from N binary units with stepped offsets, we approximate their total output with:

$$P(v|x) \sim \Big[\mathcal{N}\big(\mathrm{E}[v|x], \mathrm{Var}(v|x)\big)\Big]_0^N \quad (17)$$

where $\mathcal{N}(\cdot)$ is a normal distribution with mean and variance provided by (15) and (16), and $[\,\cdot\,]_0^N$ snaps values outside of the interval $[0, N]$ to the boundaries of the interval. Because these BLUs are constructed from logistic binary units, all of the RBM learning rules from Section 2 are still valid; the only thing that changes is how we sample from the visible BLUs.

3.2 Label Units

If bounded linear units give us a way to get drum onset information into the network, label units are how we get information out of the network. A standard approach to multi-class classification with neural networks is to use a

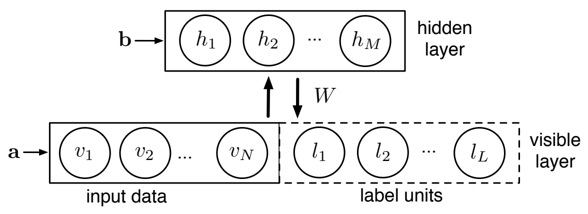


Figure 5. An RBM with an added group of visible label units.

group of softmax output units, which assigns a value to each of its units (each with input x_i) based on the softmax activation function shown in (18). This activation function is convenient for classification because the activation values of the group sum to one, which allows the output values to be interpreted as posterior class probabilities given the input data.

$$\mathcal{S}_{\max}(x_i, \mathbf{x}) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{18}$$

In the realm of RBMs and deep learning, a different approach can be used which entails providing the ground truth class labels as part of the visible data during training. This approach has been shown to be more effective than using a separate softmax output layer in certain cases [6], and it indeed achieves better results for our application. Instead of adding the label units to a separate output layer, we augment the visible layer in the top-level RBM of a deep belief network with a group of softmax label units, as shown in Figure 5. This allows us to train the top-level RBM using the label units as visible data, by turning on only the correct label unit during training. Once this labeled RBM has been trained, we can compute the posterior activation probability under the model of each of the label units given the data, $P(l|\mathbf{v})$, using (19) and (20) (see [5]):

$$\mathcal{F}(\mathbf{v}|l) = -\sum_i v_i^{(l)} a_i - \sum_j \log\left(1 + \exp\left(x_j^{(l)}\right)\right) \tag{19}$$

$$P(l|\mathbf{v}) = \frac{e^{-\mathcal{F}(\mathbf{v}|l)}}{\sum_k e^{-\mathcal{F}(\mathbf{v}|k)}} \tag{20}$$

Where $x_j^{(l)} = b_j + \sum_i W_{ij} v_i^{(l)}$, and $v_i^{(l)}$ denotes the visible data but with only unit l of the label unit group activated. This calculation is tractable due to the typically small number of label units being evaluated.

3.3 Modelling Drum Patterns

In our drum pattern analysis network, we always start with a CRBM at the first layer. This CRBM models the current drum beat or subdivision using one BLU visible unit per drum or cymbal. In our experiments, we use a minimal three-drum setup: bass drum, snare drum, and hi-hat, but this can be expanded to work with any percussion setup. The conditioning units, y_j, of the CRBM contain drum activations from the recent past. In our experiments, $\mathbf{y}$ is fed with drum activations from the most recent two measures (or 32 subdivisions given a 4/4 time signature with sixteenth note subdivisions).

Subsequent layers use binary visible units instead of BLUs. Intermediate layers of the DBN can be made up of either additional CRBMs or standard RBMs, and the final layer must have visible label units to represent the classifier output. Using an intermediate-layer CRBM allows the layer to take into account past hidden unit activations of the layer below it, which allows it to learn higher-level time dependencies. In doing so, it increases the past time context that the top-level layer sees, since the past hidden unit activations of a first-level CRBM have been conditioned on the past relative to themselves. In order to make a fair comparison between DBNs that use different numbers of CRBM layers, we must make sure that the top layer always has access to the same amount of visible first-layer data from the past.

In our experiments, we train the label units to detect the current sixteenth note beat subdivision within the current 4/4 measure. In the next section, we give details on the configuration and training of the various DBNs that we test for this task.

4. TRAINING THE SYSTEM

4.1 Training Data

The dataset consists of 173 twelve-measure sequences comprising a total of 33,216 beat subdivisions, each of which contains bass drum, snare drum, and hi-hat activations. The data was collected using electronic Roland V-drums [1], quantized to exact sixteenth note subdivisions, and converted to a subdivision-synchronous drum activation matrix.

The sequences were intended to span a sizeable, but by no means complete, collection of popular rhythmic styles. There is a strong rock bias, with many beats featuring a prominent back beat; however, also included are more syncopated styles such as funk and drum ‘n’ bass as well as the Brazilian styles samba and bossa nova. We use a randomized 70/20/10 split for training, testing, and validation data, respectively.

4.2 DBN Configurations

We test four DBN configurations. For each of the four network architectures, we tested multiple hidden unit configurations and have chosen to present only those which performed best on the test data for each architecture. They are as follows:

1. *1-layer: Labeled-CRBM*
   3 visible data units + 16 visible label units, 100 hidden units, and 32 past subdivisions of context (96 conditioning units)

2. *2-layers: CRBM → Labeled-RBM*
   Each with 100 hidden units. The CRBM again has a context of 32 subdivisions.

[1] `http://rolandus.com`

3. *2-layers: CRBM → Labeled-CRBM*
   With 100 and 200 hidden units respectively. Each CRBM has a context of 16.

4. *3-layers: CRBM → CRBM → Labeled-RBM*
   With 100, 200, and 100 hidden units respectively. Both CRBMs have a context of 16 subdivisions.

4.3 DBN Training

Each non-labeled layer was trained using contrastive divergence with $k = 1$ (CD-1) for 300 sweeps through the training data with an update batch size of 100. The order of the training data was randomized in order to smooth the learning.

Top-level labeled layers were trained with the correct visible label unit switched on and the other label units switched off. We pre-trained each labeled layer using CD with $k = 1$ for 150 epochs and then k was linearly increased from 1 to 10 for an additional 150 epochs.

After pre-training each layer, we used discriminative backpropagation to fine-tune the network by backpropagating the cross-entropy label unit error to the lower layers [6]. Backpropagation was run for 400 epochs, but in the end we used the model parameters which produced the lowest cross-entropy validation error during training.

This type of training relies heavily on multiplying large matrices, which can be done considerably faster using highly data-parallel graphics processing units (GPUs). We use Gnumpy [12], a Python module which provides Numpy-like [2] bindings for matrix operations on Nvidia GPUs. Using an Nvidia Tesla C2050 GPU, training the single-layer model (#1) took around 20 minutes, while the 3-layer model (#4) took around 30 minutes. The typical split between pre-training time and backpropagation time was around 60%/40%.

4.4 Viterbi Decoding

In addition to simply classifying each subdivision individually, we can take into account additional sequential context by providing the label probabilities as posterior state probabilities in a hidden Markov model (HMM) [10]. In order to maximize coherence between successive beat subdivision estimates, we assign a high probability of a transition to the next successive beat and give an equal division of the remaining probability to other transitions. Since our system is designed for live use, we use strictly causal Viterbi decoding to estimate the current beat subdivision.

5. RESULTS

5.1 Independent Subdivision Classification

Here we present the classification results for beat-measure alignment. The test data contains 16 beat subdivisions per 4/4 measure, so we use 16 separate label units in the training. We were concerned with the prevalence of half-note symmetry in most back-beat-oriented drum patterns. For

[2] `http://numpy.org`

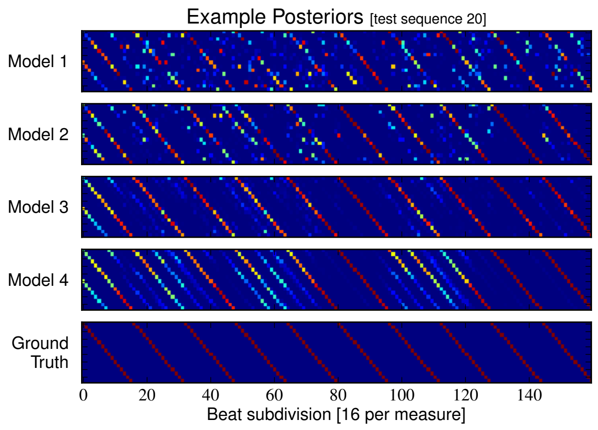


Figure 6. Example posteriors subdivision probabilities from the four models and the ground truth labels. The columns in each matrix show the posterior probability of each label for a particular beat subdivision.

example, distinguishing between the first quarter note and the third quarter note of many basic rock patterns is virtually impossible without additional contextual information. Even though this phenomenon caused the majority of classification errors, the networks seemed to do well on the whole despite it.

Table 1 shows the classification results for each model. The single layer model was significantly outperformed by all multi-layer models, and adding a third layer did not seem to provide additional benefit on our test data. Example posterior label probabilities for each model are shown in Figure 6.

DBN Configuration	Train Accuracy	Test Accuracy
L-CRBM	97.2	76.2
CRBM→L-RBM	94.9	80.8
CRBM→L-CRBM	91.0	**83.7**
CRBM→CRBM→L-RBM	89.6	81.1

Table 1. Subdivision classification accuracy for each network configuration

5.2 With Viterbi Decoding

Now we present the classification results when using Viterbi decoding. We were concerned there would be a tendency for the high sequential state transition probability to increase the number of classification errors in the presence of half-note offset ambiguities; however, the decoding only seemed to help classification. Strong half-note ambiguities seemed to provide strong enough evidence for both alternatives that the original independent classification decisions were typically unaffected by the Viterbi decoding.

As shown in Figure 7, increasing the sequential transition probability increases the overall beat classification accuracy; however, in a real-world application, one cannot simply set this probability to one or else the decoder could

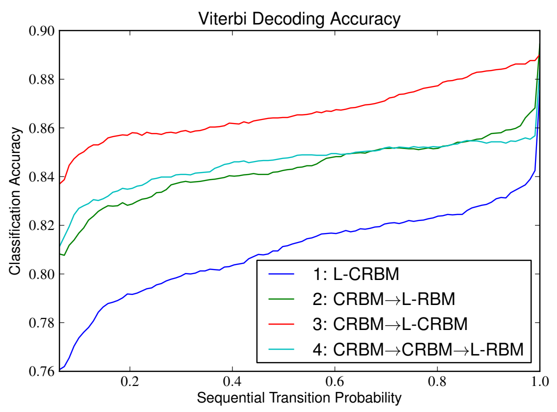


Figure 7. Viterbi decoding classification accuracy with increasing sequential transition probability

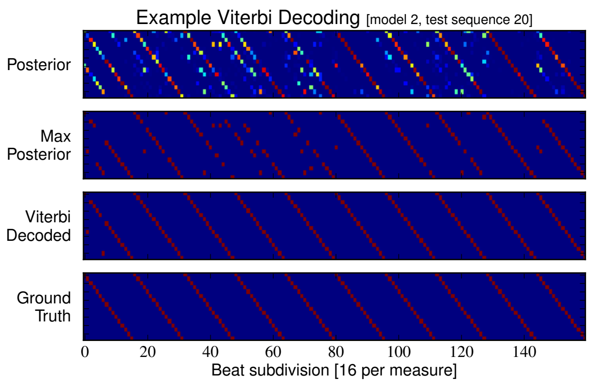


Figure 8. Row 1: Example posterior probabilities. Row 2: Independent classifications. Row 3: Viterbi decoded classifications. Row 4: Ground truth labels.

possibly be locked into an incorrect beat-measure alignment for the entire song. The decoder must also be allowed to adjust its estimates when beats are purposely left out by the drummer. Therefore, this parameter should be set with the end use case in mind. Example viterbi decoding results are shown in Figure 8.

6. DISCUSSION

Our results show the benefit of using a multi-layer neural network that is pre-trained as a deep belief network for analyzing drum patterns; however, it is likely that the actual optimal network configuration will be highly dependent on the diversity of the drum patterns in the dataset.

The results in Table 1 show an inverse relationship between training and test accuracy, which suggests overfitting was occurring during backpropagation. Subsequent work should focus on a more robust evaluation of the results using a larger dataset, cross-validation, and more attention to regularization techniques. In addition, comparison with existing drum pattern analysis methods is necessary.

Although, we do not objectively evaluate the use of these models for generating drum patterns, it is important to note that because the RBM is inherently a generative model, these networks are especially well-suited to serve as stochastic drum machines. Even a single labeled-CRBM works well for this purpose, and turning on the label unit of the desired subdivision during Gibbs sampling helps increase the metric stability of the generated patterns.

This type of model has significant potential to be of use in many music information retrieval and computer music tasks. We plan to explore the ability of the model to discriminate between rhythmic styles, discern between different time signatures, or to detect rhythmic transitions or fills. We also plan to do a more in-depth evaluation of the generative abilities of the model as well as to pursue methods which will allow interactive improvisation between human and computer performers.

Additional information as well as the dataset and code used in this work will be made available at:
`http://www.eecs.berkeley.edu/~ericb/`

7. ACKNOWLEDGEMENTS

Research supported by Microsoft (Award #024263) and Intel (Award #024894) funding and by matching funding by U.C. Discovery (Award #DIG07-10227). Additional support comes from Par Lab affiliates National Instruments, Nokia, NVIDIA, Oracle, and Samsung.

8. REFERENCES

[1] E Battenberg, V Huang, and D Wessel. Toward live drum separation using probabilistic spectral clustering based on the itakura-saito divergence. *Proc. AES Conference on Time-Frequency Processing*, 2011.

[2] Y Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2009.

[3] M Goto. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159–171, 2001.

[4] P Hamel and D Eck. Learning features from music audio with deep belief networks. *Proc. of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 339–344, 2010.

[5] G Hinton and S Osindero. A fast learning algorithm for deep belief nets. *Neural Computation*, 2006.

[6] G. E. Hinton. To recognize shapes, first learn to generate images. *Progress in brain research*, 165:535–547, 2007.

[7] AP Klapuri, AJ Eronen, and JT Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Speech and Audio Processing*, 14(1):342, 2006.

[8] A Mohamed and G Dahl. Deep belief networks for phone recognition. *NIPS Workshop on Deep Learning for Speech Recognition and Related Applications*, 2009.

[9] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. *Proc. 27th International Conference on Machine Learning*, 2010.

[10] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[11] G Taylor and G Hinton. Two Distributed-State Models For Generating High-Dimensional Time Series. *Journal of Machine Learning Research*, 2011.

[12] T Tieleman. Gnumpy: an easy way to use GPU boards in Python. Technical Report 2010-002, University of Toronto, 2010.

[13] M Welling and M Rosen-Zvi. Exponential family harmoniums with an application to information retrieval. *Advances in Neural Information Processing Systems*, 2005.

N-GRAM BASED STATISTICAL MAKAM DETECTION ON MAKAM MUSIC IN TURKEY USING SYMBOLIC DATA

Erdem Ünal
TÜBİTAK-BİLGEM
unal@uekae.tubitak.gov.tr

Barış Bozkurt
Bahçeşehir University
baris.bozkurt@bahcesehir.edu.tr

M. Kemal Karaosmanoğlu
Yildiz Technical University
kkara@yildiz.edu.tr

ABSTRACT

This work studies the effect of different score representations and the potential of n-grams in makam classification for traditional makam music in Turkey. While makams are defined with various characteristics including a distinct set of pitches, pitch hierarchy, melodic direction, typical phrases and typical makam transitions, such characteristics result in certain n-gram distributions which can be used for makam detection effectively. 13 popular makams, some of which are very similar to each other, are used in this study. Using the leave-one-out strategy, makam models are created statistically and tested against the left out music piece. Tests indicate that n-gram based statistical modeling and perplexity based similarity metric can be effectively used for makam detection. However the main dimension that cannot be captured is the overall progression which is the most unique feature for classification of close makams that uses the same scale notes as well as the same tonic.

1. INTRODUCTION

The makam/maqam/mugam concept is very central to music of a very large geographical region from Balkans to Kazakhstan, Iran, and North Africa. Automatic classification of makam is hence very important for music information retrieval technologies though not widely studied.

Computational studies on makam music can be very broadly classified into two categories based on the type of data being processed: symbolic or audio. While some works such as [1], [6], [11] propose systems for makam recognition from audio data, works on symbolic data appear to be much more limited, probably due to lack of machine readable data. In [16], Şentürk and Chordia use Variable-Length Markov Models (VLMM) to predict the melodies in the *uzunhava* (long tune) form, a melodic structure in Turkish folk music. In [2] (which is a shorter version of [8]), Alpkoçak and Gedik present the first and only study on n-grams for makam recognition. Unfortunately, due to several deficiencies, reliability of their results is questionable. The paper presents classification results without cross-validation, uses limited and questionable data (20 pieces for each of 10 makam where data is represented with 12 notes in an octave while today's notation uses 24 notes in an octave). Due to such deficiencies, a new work needs to be conducted exploring the potential of n-grams in automatic makam recognition from symbolic data. Our main contributions in this study are: in addition to the 12-TET (Tone Equal Temperament) representation used in [2], [9], we also used data represented using the official theory of makam music in Turkey (TMMT) which uses 24 tones (unequally spaced) in an octave, and holding a larger database, and challenging makam sets, we were able to test the potential of n-gram based statistical approach in makam recognition more reliably. We also tested makam detection performance using comma level intervallic movements, showing how this system can be used in real life applications using audio data only.

In the MIR literature, makam recognition can be considered, to some level, as a key finding or a mode finding problem. However, there appears to be important differences between the concepts of key, mode and makam (a detailed discussion can be found in [3]). In the makam system, different makams can be constructed using the same set of pitches, the same set of tetrachord - pentachord formulation and the same tonic. Two examples are presented in Figure 1; the scale for makam *Hüseyni* and makam *Muhayyer* (top figure) and makam *Uşşak* and makam *Beyati* (bottom figure). Then, pitch hierarchy, melodic direction, typical phrases and typical makam transitions appear to be the discriminating features for makams having the same set of pitches and tonic.

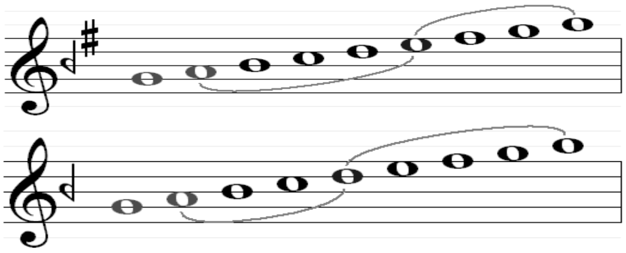


Figure 1. Scale used for makam Hüseyni and makam Muhayyer (top), makam Beyati and makam Uşşak

The listed characteristics have important influences on the pitch-class distribution of a given piece in a given makam, as in the case of key or mode in Western music

[13]. For that reason, the processing of pitch (class) histograms appears as the most common approach for computational studies of makam music [8]. N-grams can be considered to be an extension to this approach, where distributions of fixed length note-sequences are used in addition to single note pitch distributions.

The n-grams approach [10], from the text retrieval domain, have been widely used in computational studies on Western music. Various applications exist including indexing [6], query processing and music similarity computation [7], [17]. This study targets filling the gap for makam music in Turkey in the context of makam recognition. We first present the data, then details of implementation, results and discussions.

2. DATA

The current notation system (The Arel Theory (notation)) [4] used for TMMT assumes 24 notes in an octave. Although highly-criticized, almost all scores being used today are written in that system. While a large number of scanned scores are available on internet, machine readable data is very limited. Recently, we have announced the largest symbolic database of TMMT containing 1700 pieces in 155 makams [12]. Due to the availability at the time of the experiments, this study uses the following subset from [12]:

Makam name	*Total # of Songs*	*Total # of Notes*
Beyati	39	16,172
Hicaz	112	45,905
Hicazkar	48	17,950
Hüseyni	70	28,292
Hüzzam	63	26,842
Kürdilihicazkar	49	20,993
Mahur	51	22,037
Muhayyer	51	21,718
Nihavent	79	31,143
Rast	83	32,636
Saba	42	17,255
Segah	75	26,757
Uşşak	85	31,704
TOTAL	**847**	**339,404**

Table 1. Makam coverage and note statistics

The makam selection is based on three criteria: commonness, similarity and having sufficient number of samples in the database. For a classification study, it is beneficial to include similar classes and study the effects of such similarities in the classification performance. We have included in our set, makam couples which are stated to be differing only in melodic progression namely *Uşşak* - *Beyati* and *Muhayyer* - *Hüseyni* [14]. These couples share the same set of pitches, the same tonic and dominant (which is considered to be the boundary of tetrachord-pentachord division of the octave) as shown in Figure 1. As previous classification work on audio data showed, *Hüseyni* is also confused with *Uşşak* [8]. Therefore, the set includes challenging examples of classes.

2.1 Arel representation compared to 12-TET

In this work we test our system with two different representations of the symbolic data. The first one is the official theory of TMMT [4] and the second is the well-known 12-TET representation. The tonal space of the two systems are compared in Figure 2.

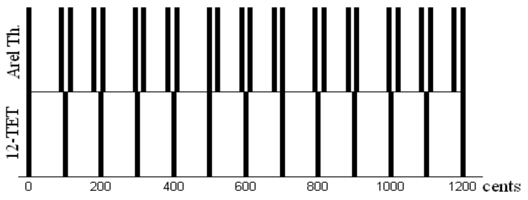


Figure 2. Tonal spaces of the Arel Theory Notation and 12-TET.

Being based on Pythagorean tuning, the 24 tones of Arel Theory are indeed close to 12-TET tones. While being a better representation (than 12-TET) for TMMT, Arel system is known to be insufficient in representing the practice. In this work we use the Arel Theory representation due to its wide use and the 12-TET representation, to be able to compare our results with [2],[9].

Arel theory uses two different but close formulations to represent notes and musical intervals, the first one being frequency ratios (such as 3/2, 9/8, etc.) and second one being the interval in integer multiples of Holdrian-commas (obtained by equal 53 divisions of an octave). The second is indeed a quantized version of the first and is more practical in explaining accidentals of the notation system as in Figure 3.

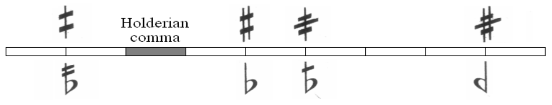


Figure 3. Accidentals used in the Arel Theory notation system

As shown in Figure 3, a whole-tone is composed of 9 Holdrian commas (will be referred as comma hereafter) in the Arel Theory system. In a machine readable format, it is convenient to name the notes using the comma steps such as *B4b1*, which corresponds to *B4* with a flat of single comma size, where *B4b4* would have a flat of 4 comma size. Alternatively each note can be represented using its distance to a reference note, for example *C1*, in commas. Such a representation makes it possible to easily obtain interval sizes (by simply subtracting the values assigned to each note as a distance in commas to a common reference) between consecutive notes which can further be used in modeling the progression (as in Section 4.4).

3. STATISTICAL MODELING

3.1 N-gram models

N-grams are widely used in computational linguistics, probability, communication theory and computational biology as well as music information retrieval [6], [7], [17]. N-grams predict X_i based on $X_{i-(n-1)}$, ..., X_{i-1}. In theory this is the information calculated by $P(X_i|_{Xi-(n-1)}$, ..., $X_{i-1})$. Given sequences of a certain set, one can statistically model this set by statistically counting the sequences that belong to it. In this study, according to the given note sequences that belong to the same makam, n-grams will be used to statistically model the pitch and intervallic space, as well as short melodic motifs to define makams.

The main hypothesis to be tested here is that, the short-time melodic contour and the frequency of makam specific notes are selective features for defining makams. This is why n-gram models are selected for training makam models using the Arel Theory notation. Given a microtonal notation sequence, using perplexity, the system will define how well the input sequence can be generated by the makam models in the database. The makam model that has the maximum similarity score is selected as the output of the system.

3.2 Smoothing

In practice, it is necessary to *smooth* the probability distributions by assigning non-zero probabilities to unseen words or n-grams. The reason is that models derived directly from the n-gram frequency counts have severe problems when confronted with any n-grams that have not been seen before which is called the "zero frequency problem". Different smoothing techniques are introduced in order to solve this problem [10]. Written-Bell smoothing technique available in the SRILM toolkit is used in our experiments [15].

3.3 Perplexity

Perplexity is a metric that is widely used for comparing probability distributions. The perplexity of a random variable *X* can be stated as the perplexity of the distribution over its possible values of *x*. Given a proposed probability model *q* (in our case: a makam model), evaluating *q* by asking how well it predicts a separate test sequence or set *x1, x2, ...,xN* (in our case: a microtonal note sequence) also drawn from *p*, can be performed by using the perplexity of the model *q*, defined by:

$$2^{\sum_{i=1}^{N}\frac{1}{N}log_2 q(x_i)} \qquad (1)$$

For the test events, we can see that better models will assign better probability scores thus a lower perplexity score which means it has a better potential to compress that data set. The exponent is the cross entropy per definition:

$$H(p,q) = -\sum_x p(x) log_2 q(x) \qquad (2)$$

The cross entropy thus the perplexity is the similarity measure between the test input and the makam models in the database. For each of the makam models defined, the system calculates the similarity metric to evaluate which makam is the most similar to the input sequence given.

4. EXPERIMENTAL SETUP

4.1 Leave-one-out

The experimental setup can be found in Figure 4, explaining how the leave-one-out strategy is inherited.

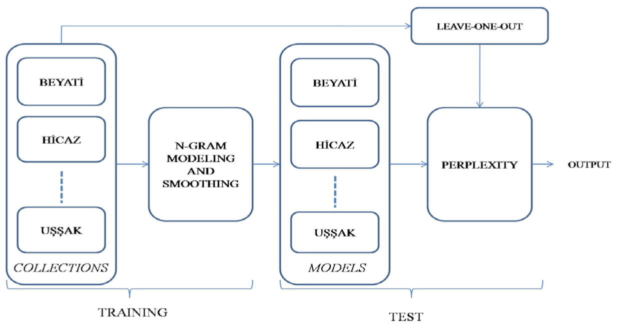


Figure 4. The leave-one-out experimental setup

There are 13 makam classes. Each of them has unequal number of music pieces. Since our approach is statistical, for each class, it is desirable to have a training set and a separate test set. This approach is feasible in case there is enough data. Since machine-readable microtonal notation is very hard to find in makam music, the "leave-one-out" strategy will be used in this experimental setup in order to avoid the negative effect of unequal set sizes. For each iteration of the experiment, one music piece will be selected as the input and the leftover music pieces will be used for training the genuine and the imposter makam classes. Using a probabilistic evaluation metric (perplexity), the system will calculate a similarity measure between the input and already built makam models.

4.2 Evaluation

Given a note sequence, the perplexity will estimate how well this sequence can be statistically generated by the makam models in our search space. Between each of the makam models, and the input sequence, the system calculates a similarity measure, and the makam that produces the maximum similarity measure becomes the output of the system. The performance criteria of the experimental procedure is binary, which is either a success or failure. The matching performance of the entire system, which is the accuracy (Recall), will be given as a proportion of successes over the total test trials in terms of Total Average (Tot-Ave) and the Weighted Average (W-Ave).

byati	hicaz	hczkr	hsyni	huzzm	krdhz	mahur	muhyr	nhvnt	rast	saba	segah	Ussak	Ref	Rcl.
24	0	0	3	0	0	0	1	0	0	0	0	11	**byati**	61.5
0	112	0	0	0	0	0	0	0	0	0	0	0	**hicaz**	100
0	0	48	0	0	0	0	0	0	0	0	0	0	**hczkr**	100
1	0	0	50	0	0	0	13	0	0	0	0	0	**hsyni**	71.4
0	0	0	0	62	0	0	0	0	0	0	1	0	**huzzm**	98.4
0	0	0	0	0	49	0	0	0	0	0	0	0	**krdhz**	100
0	0	0	0	0	0	51	0	0	0	0	0	0	**mahur**	100
2	0	0	11	0	0	0	35	0	3	0	0	0	**muhyr**	68.6
0	0	0	0	0	0	0	0	78	1	0	0	0	**nhvnt**	98.7
0	0	0	1	0	0	0	0	0	76	0	0	4	**rast**	91.6
0	0	0	1	0	0	0	0	0	0	41	0	0	**saba**	97.6
0	0	0	1	2	0	0	0	0	0	0	70	0	**segah**	95.9
16	0	0	11	0	0	0	4	0	3	0	2	49	**ussak**	57.6
55.8	100	100	64.1	96.9	100	100	63.6	100	91.6	100	95.9	70	Prc.	

Table 2. Confusion Matrix for Arel Theory Notation (n=3)

4.3 Tests with the Arel Theory Notation

In information retrieval, the output of such kind of systems are evaluated given 2 different measures. Given a music piece, the Recall (Rcl) suggests, how many of the queries for each of the makams are correctly found. On the other hand, precision is how many of the retrieved makams belong to the correct reference makam class. Precision becomes more meaningful when there is equal number of test trials from each makam classes. As seen from Table 3, 4 makams has perfect recall rate which are *Hicaz, Hicazkar, Kürdilihicazkar* and *Mahur*. The makam which shows the worst performance is *Beyati* as the recall rate is 61.5% and it is confused with *Uşşak*, the most (for n=3).

The confusion matrix also suggests that there are concrete similarities between these makam sets: *Beyati - Uşşak* and *Hüseyni - Muhayyer*. In theory these makam couples use exactly the same microtonal note sets as well as the tonics and the effect of this similarity can be practically seen in our experiments.

Table 3 shows the change in Recall metric when the order of the n-grams increased from 1 to 3. Also, the last column shows which n-gram shows the best performance with respect to Recall. As seen from results, for the makam, *Hicazkar, Hüseyni, Rast* and *Segah*, increasing the order of n-grams from 1 to 2 or 3, improves the makam detection performance of the classifier. For *Hicaz, Hüzzam, Mahur, Nihavent* and *Saba* increasing the order of the n-grams did not have any positive influence. On the other hand, increasing N has negative effect on the performance of the classifier for the makams *Beyati, Muhayyer* and *Uşşak*.

There might be a number of reasons for performance fluctuation within different makams. The one that we believe the most important is the unequal number of notes for training each makam classes. Even though a smoothing technique is used, the frequency of widely seen sequences become more dominant for makams that have few training samples (such as *Beyati, Hüzzam, Segah* and *Uşşak*), which makes these makams harder to be distinguished from the ones that are similar.

	n=1	n=2	n=3	Best-N
Beyati	64.1	56.4	61.5	1
Hicaz	100	99.1	100	1,3
Hicazkar	97.9	100	100	2,3
Hüseyni	50	64.3	71.4	3
Hüzzam	98.4	98.4	98.4	1,2,3
Kürdilihicakar	100	100	100	1,2,3
Mahur	100	100	100	1,2,3
Muhayyer	80.4	68.6	68.6	1
Nihavent	98.7	98.7	98.7	1,2,3
Rast	74.7	92.8	91.6	2
Saba	97.6	97.6	97.6	1,2,3
segah	93.2	97.3	95.9	2
Uşşak	68.2	62.4	57.6	1
Tot-Avg	86.3	87.9	88.2	3
W-Avg	86.4	87.4	87.8	3

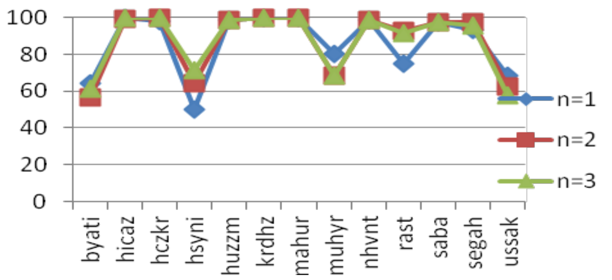


Table 3. Change in Recall w.r.t. n-gram order for data using the Arel Theory notation

4.4 Tests with microtonal intervals

Considering that the real world application of this system will operate with audio inputs, and it is known that a direct transcription of audio to the microtonal sequence used above is not easy, tests on data represented as sequence of microtonal intervals are also applied. The interval between consecutive notes are computed in commas as explained in Section 2.1 This also ensures that the system functionality is independent of the starting note of the music piece, the type or the tuning of the instrument that plays the piece.

	n=1	n=2	n=3	Best-N
Beyati	56.4	64.1	59	2
Hicaz	61.6	81.2	93.8	3
Hicazkar	66.7	85.4	85.4	2,3
Hüseyni	30	50	60	3
Hüzzam	69.8	87.3	85.7	2
Kürdilihicazkar	38.8	71.4	77.6	3
Mahur	80.4	90.2	94.1	3
Muhayyer	51	47.2	43.1	1
Nihavent	70.9	92.4	97.5	3
Rast	69.9	89.2	88	2
Saba	97.6	97.6	97.6	1,2,3
Segah	69.9	94.5	94.5	2,3
Uşşak	21.2	51.8	56.5	3
Tot-Avg	58.9	77.3	80.6	3
W-Avg	60.3	77.1	79.4	3

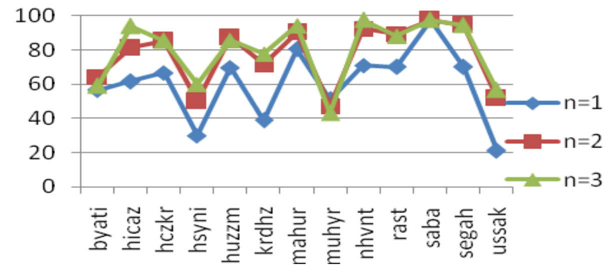


Table 4. Change in Recall w.r.t. n-gram order with data represented as microtonal intervals

The basic goal in this experimental setup is to achieve at least a close performance to the test explained in 4.3, and thus the cost of losing absolute note level information can be tested over the system performance. The makam detection performance for the microtonal representations can be seen in Table 4.

4.5 The Baseline: 12-TET Input Tests

Finally we evaluated the performance of our system on data represented using 12-TET since it is the representation used in the only available system in literature that does makam detection using n-grams [2], [9]. In addition to important differences in the implementation, modeling and evaluation, this study uses and compares different data representations, where in [2], [9] only the 12-TET representation is used. Since the basic strategy is building n-grams for both systems, we ran our experimental setup on the same database using the leave-one-out technique.

Table 5 shows the results with respect to increasing n-grams per each makam in the database. Since neither the evaluation nor the modeling technique is clearly explained in [2], [9], the standard modeling and smoothing techniques in our system was used when implementing the baseline (i.e. the system using the 12-TET representation). As seen from the results, the best performing n-gram order is 3, similar to results gathered from Arel theory tests. However, the system using the Arel Theory notation outperforms the baseline for both the Weighted Average and the Total Average.

	n=1	n=2	n=3	Best-N
Beyati	66.7	56.4	61.5	1
Hicaz	98.2	100	100	2,3
Hicazkar	91.7	97.9	97.9	2,3
Hüseyni	42.9	58.6	72.9	3
Hüzzam	98.4	98.4	98.4	1,2,3
Kürdilihicakar	100	100	98	1,2
Mahur	70.6	82.4	74.5	2
Muhayyer	80.4	70.6	66.7	1
Nihavent	97.5	97.5	97.5	1,2,3
Rast	72.3	75.9	80.7	3
Saba	97.6	97.6	97.6	1,2,3
segah	75.3	84.9	89	3
Uşşak	69.4	54.1	56.5	1
Tot-Avg	81.7	82.8	84.5	3
W-Avg	81.6	82.6	83.9	3

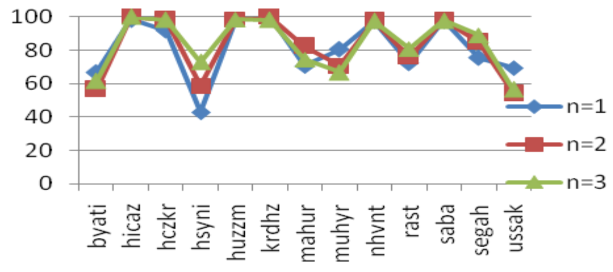


Table 5. Change in Recall w.r.t. n-gram order for 12 TET

The overall comparison of the performance of all the tests can be seen in Table 6. For *n*=3 where the best performance for all the systems were achieved, we observe that by using the Arel Theory notation as opposed to 12-TET, an improvement of 3.7% is achieved.

Recall	n=1	n=2	n=3
Arel Theory	86.3	87.9	88.2
12-TET	81.7	82.8	84.5
Delta (in commas)	58.9	77.3	80.6

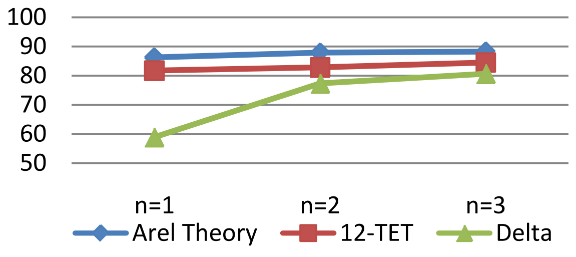


Table 6. Overall Performance Comparison.

5. DISCUSSION AND CONCLUSION

In this work, we implemented a perplexity based makam detection system on symbolic data of TMMT. N-gram based statistical makam models were built using the SRILM toolkit. Necessary smoothing was performed in order to compensate the negative effect of unequal set sizes.

Experimental set up was designed using the leave-one-out approach. For each of the test trials, one song from the database was chosen as the input. The rest of the pieces were used for modeling the makam classes.

Three different experimental setups were created. The tests with data represented using Arel Theory showed that the overall recall performance of the system is 88.2%. Increasing the order of n-grams boosted the classification performance as expected. However, the effect is different for different makams. We observed that increasing the n-gram order did not help when trying to distinguish makams that use the same scale such as *Beyati-Uşşak* and *Muhayyer-Hüseyni*. The second experimental setup was for a real application case, where there is no direct note level transcription. For this test, the data is represented as intervals (in commas) between consecutive notes. This experiment was designed to provide reference information for research on makam detection directly from audio where exact note level transcription is not available. For audio, due to different instrumentation, and tuning, the only reliable information is the intervallic movement. The result showed that the makam detection accuracy is %80.6 using with n order 3. Note that, higher order n grams did not improve the experimental results beyond n=3 because of data sparsity.

For comparison with a previous study [2], [9], both the data representation in [2], [9] and additional representations (Arel and interval representation) are tested and compared. It is observed from tests using a large dataset, and challenging makam couple sets, that a system using Arel Theory representation outperforms a system using the 12-TET representation on average 3.7% percent. Increasing the n-gram order beyond 3 did not improve the performance of the tests due to lack of data.

Future work includes defining global tonal features that could help distinguishing makams having the same microtonal scale and tonic such as *Uşşak - Beyati* and *Hüseyni - Muhayyer*. Features related to global progression of the melody could give clues that cannot be captured by n-grams that concentrate more on local short length movements.

6. ACKNOWLEDGEMENT

This work was funded in part by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 267583 (CompMusic) and in part by TÜBİTAK ARDEB grant no: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 European Union or the TÜBİTAK.

7. REFERENCES

[1] S. Abdoli,. "Iranian Traditional Music Dastgah Classification," *ISMIR, Florida*,2011

[2] 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.

[3] T. Aoyagi "Makam Rast: Intervallic Ordering, Pitch Hierarchy, Performance and Perception of a Melodic Mode in Arab Music," *University of California*, 2001.

[4] H. S. Arel, "Türk Musikisi Nazariyatı Dersleri, Hazırlayan Onur Akdoğu," *Kültür Bakanlığı Yayınları /1347, Ankara*, p.70. 1991,

[5] N. Darabi, N. Azimi and H. Nojumi, "Recognition of Dastgah and Makam for Persian Music with Detecting Skeletal Melodic Models," *The second annual IEEE BENELUX/DSP Valley Signal Processing Symposium* 2006

[6] S. Doraisamy, "Polyphonic Music Retrieval: The N - gram Approach," *PhD Thesis, University of London,* 2004.

[7] S. Downie "Evaluating a simple approach to music information retrieval: Conceiving melodic n-grams as text," *PhD thesis, University of Western Ontario, 1999.*

[8] A. C. Gedik, and B. Bozkurt, "Pitch-frequency histogram-based music information retrieval for Turkish music," *Signal Processing*, 90(4), 1049-1063, 2010.

[9] A.C. Gedik, C. Işıkhan, A. Alpkoçak and Y. Özer, "Automatic Classification of 10 Turkish Makams," *International Congress on Representation in Music & Musical Representation*, *İstanbul*, 2005.

[10] H. S. Heaps "Information Retrieval: Computational and Theoretical Aspects," *Academic Press*, 1978.

[11] L. Ioannidis, E. Gómez, and P. Herrera, "Tonal-based retrieval of Arabic and Middle-East music by automatic makam description," *CBMI*, 2011.

[12] M. K. Karaosmanoğlu, "A Turkish makam music symbolic database for music information retrieval: SymbTr," *submitted to ISMIR*, 2012.

[13] C.L. Krumhansl, "Cognitive Foundations of Musical Pitch," *Oxford University Press, NewYork*, 1990.

[14] I. H. Özkan, "Türk musikisi nazariyatı ve usulleri: kudüm velveleleri,". *Ötüken Neşriyat*, 2006.

[15] A. Stolcke: "Srilm – an Extensible Language Modeling Toolkit," *Proceedings of the International Conference on Spoken Language Processing,* 2002.

[16] S. Şentürk, and P. Chordia, "Modeling Melodic Improvisation in Turkish Folk Music Using Variable-length Markov Models," *ISMIR*, *Florida*, 2011.

[17] E. Unal, E. Chew, P. Georgiou and S. Narayanan, "Perplexity based cover song identification System for short length queries," *ISMIR, Florida*, 2011

EVALUATING THE ONLINE CAPABILITIES OF ONSET DETECTION METHODS

Sebastian Böck, Florian Krebs and Markus Schedl
Department of Computational Perception
Johannes Kepler University, Linz, Austria

ABSTRACT

In this paper, we evaluate various onset detection algorithms in terms of their online capabilities. Most methods use some kind of normalization over time, which renders them unusable for online tasks. We modified existing methods to enable online application and evaluated their performance on a large dataset consisting of 27,774 annotated onsets. We focus particularly on the incorporated preprocessing and peak detection methods. We show that, with the right choice of parameters, the maximum achievable performance is in the same range as that of offline algorithms, and that preprocessing can improve the results considerably. Furthermore, we propose a new onset detection method based on the common *spectral flux* and a new peak-picking method which outperforms traditional methods both online and offline and works with audio signals of various volume levels.

1. INTRODUCTION AND RELATED WORK

Onset detection, the task of finding musically meaningful events in audio signals, is fundamental to many applications: Real-time applications such as automatic score followers [7] can be enhanced by incorporating (online) onset detectors that look for note onsets in a live performance, while (offline) onset detection is used increasingly to improve digital audio workstations with a view to event-wise audio processing.

Many different methods of solving this task have been proposed and evaluated over the years. Comprehensive overviews of onset detection methods were presented by Bello et al. in [2] and Collins in [6] (with special emphasis on psychoacoustically motivated methods in the latter). Dixon proposed enhancements to several of these in [9]. All methods were evaluated in an *offline* setting, using a normalization over the whole length of the signal or applying averaging techniques which require future information.

For *online* onset detection, only few evaluations have been carried out: Brossier et al. [5] compared four onset functions based on spectral features and proposed a method for dynamic thresholding in online scenarios, using a dataset of 1,066 onsets for evaluation. Stowell and Plumbley [18] proposed *adaptive whitening* as an improvement to short-time Fourier transform (STFT) based onset detection methods and evaluated eight detection functions using a dataset of 9,333 onsets. Glover at al. [12] applied linear prediction and sinusoidal modeling to online onset detection, but used a relatively small dataset of approximately 500 onsets for evaluation.

These traditional onset detection methods usually incorporate only spectral and/or phase information of the signal, are easy to implement, and have modest computational cost. In contrast, methods based on machine learning techniques (e.g., neural networks in [11,15]) or on probabilistic information (e.g., Hidden Markov models in [8]) depend on large datasets for training and are in general computationally more demanding, which makes them unsuited for online processing.

The onset detection process is usually divided into three parts (as shown in Figure 1): signal preprocessing, computation of the actual onset detection function (ODF), and peak detection.

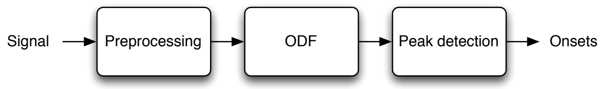


Figure 1. Basic onset detection workflow.

There are generally two normalization steps that require special attention in an online context: The first can be found in the preprocessing step where many implementations normalize the audio input prior to further processing.

The second and more widespread use of normalization is in the peak detection stage, where the whole ODF is normalized before being processed further. An exception to this rule are some machine learning approaches like the neural network-based methods, since their detection function can be considered as a probability function which already has the range [0..1]. Furthermore, most offline methods use smoothing or averaging over (future) time to compute dynamic thresholds for the final peak-picking.

This paper is structured as follows: We combine the ODFs described in Section 2.2 with different preprocessing methods from Section 2.1 and evaluate them on the dataset described in Section 3.1 using the peak-picking method given in Section 2.3.4. In Section 4 we discuss the results, and we give conclusions in Section 5.

2. COMPARED METHODS

Previously, onset detection algorithms used to work directly with the time signal $x(t)$. However, all current onset detection algorithms use a frequency representation of the signal. We used frames of 23 ms length (2048 samples at a sample rate of 44.1 kHz) that are filtered with a Hann window before transfer into the frequency domain by means of STFT. The hopsize between two consecutive frames was set to 10 ms, which results in a frame rate of 100 frames per second. The resulting spectrogram $X(n,k)$ (n denoting the frame and k the frequency bin number) was then processed further by the individual preprocessing and onset detection algorithms.

2.1 Preprocessing

2.1.1 Filtering

Scheirer [17] stated that, in onset detection, it is advantageous if the system divides the frequency range into fewer sub-bands as done by the human auditory system. Filtering has been applied by many authors (e.g. [6,14,17]), and neural network based approaches also use filter banks to reduce the dimensionality of the STFT spectrogram [11].

2.1.2 Logarithmic magnitude

Using the logarithmic magnitude instead of the linear representation was found to yield better results in many cases, independently of the ODF used [11,14]. λ is a compression parameter and was adjusted for each method separately. Adding a constant value of 1 results in only positive values:

$$X^{log}(n,k) = log(\lambda \cdot X(n,k) + 1) \tag{1}$$

2.1.3 Adaptive whitening

Proposed in [18], adaptive whitening normalizes the magnitudes $|X(n,k)|$ of each frequency bin separately by past peak values. The iterative algorithm (with r being a floor parameter and m the memory coefficient) is given as follows:

$$P_{n,k} = \begin{cases} max(|X(n,k)|, r, m \cdot P_{n-1,k}) & \text{if } n \geq 1 \\ max(|X(n,k)|, r) & \text{otherwise} \end{cases}$$

$$|X(n,k)| \leftarrow \frac{|X(n,k)|}{P_{n,k}} \tag{2}$$

2.2 Onset detection functions

We have chose to omit other common methods such as phase deviation (PD) [3], high frequency content (HFC) [16] or rectified complex domain (RCD) [9], since they exhibited inferior performance in our tests.

2.2.1 Spectral Flux

The spectral flux (SF) [16] describes the temporal evolution of the magnitude spectrogram by computing the difference between two consecutive short-time spectra. This difference is determined separately for each frequency bin, and all positive differences are then summed to yield the detection function.

$$SF(n) = \sum_{k=1}^{k=\frac{N}{2}} H(|X(n,k)| - |X(n-1,k)|) \tag{3}$$

with $H(x) = \frac{x+|x|}{2}$ being the half-wave rectifier function. Variants of this method use the L_2-norm instead of the L_1-norm or the logarithmic magnitude [14] (cf. Section 2.1.2).

2.2.2 Weighted Phase Deviation

Another class of detection function utilizes the phase of the signal [3, 9]. The change in the instantaneous frequency (the second order derivative of the phase $\varphi(n,k)$) is an indicator of a possible onset. In [9], an improvement to the phase deviation ODF called weighted phase deviation (WPD) was proposed. The WPD function weights each frequency bin of the phase deviation function with its magnitude.

$$WPD(n) = \frac{2}{N} \sum_{k=1}^{k=\frac{N}{2}} |X(n,k) \cdot \varphi''(n,k)| \tag{4}$$

2.2.3 Complex Domain

Another way to incorporate both magnitude and phase information (as in the WPD detection function) was proposed in [10]. First, the expected target amplitude and phase $X_T(n,k)$ for the current frame are estimated based on the values of the two previous frames assuming constant amplitude and rate of phase change. The complex domain (CD) ODF is then defined as:

$$CD(n) = \sum_{k=1}^{k=\frac{N}{2}} |X(n,k) - X_T(n,k)| \tag{5}$$

2.3 Peak detection

Illustrated in Figure 2 and common to all onset detection methods is the final thresholding and peak-picking step to detect the onsets in the ODF. Various methods have been proposed in the literature; we give an overview of the different components and modifications needed to make them suitable for online processing.

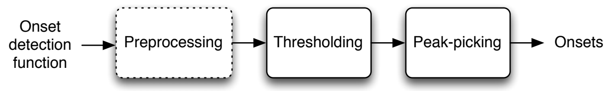


Figure 2. Peak detection process.

2.3.1 *Preprocessing*

The preprocessing stage of the peak detection process consists mainly of two components: smoothing of the peaky ODF and normalization. Both of them cannot be used in an online scenario. Instead, moving average techniques as outlined in Section 2.3.2 are applied to normalize the ODF locally. To prevent detecting many false positives due to a peaky ODF, the effect of smoothing can be approximated by introducing a minimal distance from the last onset w_5 as proposed in Section 2.3.4.

2.3.2 *Thresholding*

Before picking the final onsets from the ODF, thresholding is performed to discard the non-onset peaks. Most methods use dynamic thresholding to take into account the loudness variations of a music piece. Mean [9], median [3, 11, 18] or combinations [5, 12] are commonly used to filter the ODF. If only information about the present or past is used, the thresholding function is suitable for online processing.

2.3.3 *Peak-picking*

Two peak-picking methods are commonly used for final detection of onsets. One selects all local maxima in the thresholded detection function as the final onset positions. Since detecting a local maximum requires both past and future information, this method is only applicable to offline processing.

The other method selects all values above the previously calculated threshold as onsets and is also suitable for online processing. The downside of this approach is its relatively high false positive rate because the threshold parameter must be set to a very low level to detect the onsets reliably.

2.3.4 *Proposed peak detection*

We use a modified version of the peak picking method proposed in [9] to also satisfy the constraints for online onset peak detection. A frame n is selected as an onset if the corresponding $ODF(n)$ fulfills the following three conditions:

1. $ODF(n) = max(ODF(n - w_1 : n + w_2)$
2. $ODF(n) \geq mean(ODF(n - w_3 : n + w_4)) + \delta$
3. $n - n_{last\,onset} > w_5$

where δ is a fixed threshold and $w_1..w_5$ are tunable peak-picking parameters. For online detection, we set $w_2 = w_4 = 0$. Our online experiments experiments showed that, on average, onsets are detected one frame earlier than annotated in the dataset (using the values specified in Section 3.3). As we want to find the perceptual onset times (as annotated), we report the onset one frame later than detected. Note that this does not mean that we predict the onset, it only means that the onset can be recognized in the signal before it is perceived.

Unlike in previous studies [5, 12, 18] we do not use the same thresholding parameters for all ODFs. This is mainly because some of the ODFs have fewer peaks and hence need less averaging in the thresholding stage than others.

2.4 Neural network based methods

For reference, we compare the presented methods with two state-of-the-art algorithms, the *OnsetDetector* [11] and its online variant *OnsetDetector.LL* [4]:

OnsetDetector uses a bidirectional neural network which processes the signal both in a forward and backward manner, making it an offline algorithm. The algorithms showed exceptional performance compared to other algorithms independently of the type of onsets in the audio material, especially in its latest version tested during the MIREX contest in 2011 [1].

OnsetDetector.LL incorporates a unidirectional neural network to model the sequence of onsets based solely on causal audio signal information.

Since these methods show very sharp peaks (representing the propability of an onset) at the actual onset positions, the before mentioned peak detection method is not applied, and a simple thresholding is used instead.

2.5 New method

We propose a new onset detection method which is based on the spectral flux (cf. Section 2.2.1), drawing on various other author's ideas.

As a first step, we filter the linear magnitude spectrogram $|X(n,k)|$ with a filter bank. We investigated different types of filter banks (Mel, Bark, Constant-Q) and found that they all outperform the standard spectral flux. Since they all perform approximately equally well when using a similar number of filter bands, we chose a pseudo Constant-Q, where the frequencies are aligned according to the frequencies of the semitones of the western music scale over the frequency range from 27.5 Hz to 16 kHz, but using a fixed window length for the STFT. Overlapping triangular filters sum all STFT bins belonging to one filter bin (similarly to Mel filtering). The resulting filter bank $F(k,b)$ has $B = 82$ frequency bins with b denoting the bin number of the filter and k the bin number of the linear spectrogram. The filters have not been normalized, resulting in an emphasis of the higher frequencies, similar to the HFC method. The resulting filtered spectrogram $X_{filt}(n,b)$ is given by:

$$X_{filt}(n,b) = |X(n,k)| \cdot F(k,b) \tag{6}$$

Applying Equation 1 to the filtered linear magnitude spectrogram $X_{filt}(n,b)$ yields the logarithmic filtered spectrogram $X^{log}_{filt}(n,b)$. The final ODF O is then given by:

$$O(n) = \sum_{k=1}^{k=\frac{N}{2}} H\left(\left|X^{log}_{filt}(n,b)\right| - \left|X^{log}_{filt}(n-1,b)\right|\right) \tag{7}$$

where H is the half-wave rectifier function defined in Section 2.2.1.

3. EXPERIMENTS

To evaluate the methods described, we conducted three experiments: First, the methods were evaluated under online conditions: no future information was used to decide whether there is an onset at the current time point. Second, the same methods were evaluated under offline conditions (enabling prior data normalization or computing averages that incorporate future information) to determine the maximum performance achievable by each method. Third, we attenuated the volume of the audio data to an increasing degree to test the online methods' abilities to cope with signals of different volume without access to normalization.

3.1 Dataset

To evaluate the presented onset detection and peak-picking methods we use a dataset of real world recordings.

An onset is usually defined as the exact time a note or instrument starts sounding after being played. However, this timing is hard to determine, and thus it is impossible to annotate the real onset timing in complex audio recordings with multiple instruments, voices, and effects. Thus, the most commonly used method for onset annotation is marking the earliest time point at which a sound is audible by humans. This instant cannot be defined in pure terms (e.g., minimum increase of volume or sound pressure), but is a rather complex mixture of various factors.

The annotation process is very time-consuming because it is performed in multiple passes. First, onsets are annotated manually during slowed down playback. In the second pass, visualization support is used to refine the onset positions. Spectrograms obtained with different STFT lengths are used in combination to capture the precise timing of an onset without missing any onset due to insufficient frequency resolution. This multi-resolution procedure seems to be a good approach since the best onset detection algorithms also use this mechanism. If multiple onsets are located in close vicinity, they are annotated as multiple onsets.

The dataset contains 321 audio excerpts taken from various sources. 87 tracks were taken from the dataset used in [11], 23 from [2], and 92 from [13]. All annotations were manually checked and corrected to match the annotation style outlined above. The remaining 119 files were newly annotated and contain the vast majority of the 27,774 onsets of the complete set.

Although musically correct, the precise annotations (raw onsets) do not necessarily represent human perceptions of onsets. Thus, all onsets within 30 ms were combined into a single one located at the arithmetic mean of the positions [1], which resulted in 25,966 combined onsets used for evaluation. The dataset can be roughly divided into six main groups (Table 3.1).

Type of audio	Files	Raw onsets	Combined
Complex mixtures	193	21,091	19,492
Pitched percussive	60	2,981	2,795
Non-pitched perc.	17	1,390	1,376
Wind instruments	25	822	820
Bowed strings	23	1,180	1,177
Vocal	3	310	306
ALL	321	27,774	25,966

Table 1. Description of the used dataset: Pitched percussive (e.g., piano, guitar), non-pitched percussive (e.g., percussion), wind instruments (e.g., sax, trumpet), bowed string instruments (e.g., violin, kemence), monophonic vocal music and complex mixtures (e.g., jazz, pop, classical music)

3.2 Measures

For evaluation, the standard measures precision, recall, and F-measure were used. An onset is considered to be correctly detected if there is a ground truth annotation within ± 25 ms around the predicted position. This rather strict evaluation method (also used in [11] and [6] for percussive sounds) was chosen because it gives more meaningful results - especially in online onset detection - than an evaluation window of ± 50 ms as used in [2, 9, 18].

An important factor in the evaluation is how false positives and negatives are counted. Let us assume that two onsets are detected inside the detection window around a single annotation. If tolerant counting is used, no false positives are counted. Every single detection is considered a true positive, since there is an annotated onset within the detection window. This is often referred to as *merged* onsets. If counted in a strict way, all annotated onsets can only be matched once, i.e., two detections within the detection window of a single onset are counted as one true positive and one false positive detection.

Since many papers do not explicitly describe the criteria, it must be assumed that the results were obtained with the first method (usually yielding better results). In this paper, we evaluated the stricter way, but with combined annotated onsets (not to be confused with merged onsets). The combining of onsets leads to less false negative detections if the algorithm reports only a single onset where multiple ones are annotated. Since most of the algorithms are not capable reporting multiple consecutive onsets, this results in a more fair comparison.

3.3 Parameter selection

The peak-picking parameters $w_1 ... w_5$ and the fixed threshhold δ introduced in Section 2.3.3 were optimized by a grid search over the whole set for each method separately. As in [2, 9], we report the best performance for each method using the optimized global parameter set. For online detection ($w_2 = w_4 = 0$), the optimal values for w_3 were found to be between 4 and 12, $w_1 = 3$, and $w_5 = 3$. For the offline case, $w_2 = 3$, $w_4 = 1$ and $w_5 = 0$ yielded the best results (w_1 and w_3 were left unchanged). The adaptive whitening parameters $m = 10$ and $r = 0.005$ were found to be generally good settings and were used for all ODFs in the experiments. The compression parameter λ (Section 2.1.2) was chosen to be between 0.01 and 20. The neu-

[1] To better predict the perceived position of an onset, psychoacoustical knowledge must be applied. Since the masking effects involved depend on both loudness and frequency of an onset, they are not applied here. For the evaluation of onset detection methods as in this paper, the selected method of combination is adequate.

ral networks are trained and evaluated using 8-fold cross validation on disjoint training, validation, and test sets.

All parameters were optimized on the dataset and left unchanged for the unnormalized penalty task.

4. RESULTS AND DISCUSSION

4.1 Comparison of different ODFs

Table 2 lists the results for all algorithms working in online mode on the complete dataset using the peak detection method described in Section 2.3.4. It shows that application of adaptive whitening and use of a logarithmic magnitude both outperform the traditional methods without any preprocessing. Both preprocessing methods compress the magnitude and hence emphasize higher frequency bands that are important for detecting percussive onsets. Furthermore, our proposed method (*SF log filtered*) clearly outperforms all the other methods (apart from the reference *OnsetDetector.LL*). In particular, it is characterized by a high precision value due to the reduced number of false positives compared to the other methods. We believe that the filtering process reduces the spectrum to the most relevant components for onset detection. This may facilitate better distinction between signal changes that are arising from an onset and spurious, non-onset-related changes.

Online algorithm	% F-meas.	% Prec.	% Rec.
SF	74.5	76.3	72.8
SF aw	75.7	78.0	73.4
SF log	76.1	78.3	74.0
SF log filtered	**80.3**	**88.3**	73.5
CD	71.1	72.4	69.8
CD aw	75.8	76.4	**75.1**
CD log	74.1	77.4	71.1
WPD	69.7	68.8	70.6
WPD aw	71.4	70.8	72.0
WPD log	70.9	74.6	67.5
OnsetDetector.LL [4]	81.7	85.0	78.7

Table 2. F-measure, precision and recall of different onset detection algorithms using *online* peak-picking, where *aw* denotes adaptive whitening, *log* denotes the use of a logarithmic magnitude and *SF log filtered* is the method proposed in Section 2.5

.

Our tests showed that - if the parameters are properly chosen - the offline results are in the same range as the online results[2]. We deem this is a remarkable finding and think that the reasons for this behavior are the following: First, the audio tracks of the dataset have similar volume levels, which renders the normalization step less important. Second, when looking only at single independent frames, it seems reasonable that frames after the current onset frame do not carry much additional information. However, the superior results of the offline *OnsetDetector* (F-measure 86.6, precision 90.6, recall 83.0) suggest that using both past and future information contained in the magnitude spectrogram can be valuable to detect also the "harder" onsets (as reflected by the much higher recall value of this method).

[2] We observed an average gain in F-measure of 0.25% in offline mode

4.2 Unnormalized penalty

When dealing with unnormalized data, the investigated onset detection methods experience different levels of performance loss. As shown in Figure 3, our proposed onset detection method exhibits superior performance at all attenuation levels and is only beaten by the $OnsetDetector.LL$, that is unaffected by any volume changes. This shows the power of machine learning techniques that do not depend on predefined peak-picking threshholds. The methods using adaptive whitening score third, which seems reasonable as these methods include an implicit normalization using past frames. Computing the difference of two adjacent frames of the logarithmic spectrum (*SF log*) has the effect of dividing the magnitude at frame n by that at frame $n-1$, resulting in the relative magnitude change rather than the absolute difference. This makes the spectral flux obtained with logarithmic magnitudes more robust against absolute volume changes, compared to the standard variant (SF).

Finally, the compression parameter λ was found to influence greatly the shape of the performance curve (when using the logarithmic magnitude spectrum), especially at lower volume levels.

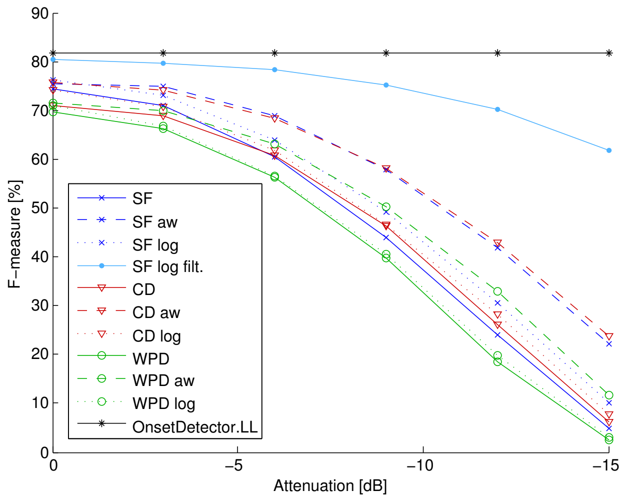


Figure 3. Performance of the online methods at different attenuation levels.

4.3 Remarks

In this paper, we give only results for the complete dataset. Results for subsets (organized by audio type and author) obtained with different detection window sizes can be found online at `http://www.cp.jku.at/people/boeck/ISMIR2012.html`.

5. CONCLUSIONS

In this paper we have evaluated various onset detection algorithms in terms of their suitability for online use, focusing on the preprocessing and peak detection algorithms. We have shown that using logarithmic magnitudes or adaptive whitening as a preprocessing step results in improved performance in all methods investigated. When the parameters for peak detection are chosen carefully, online methods can achieve results in the same range as those of offline methods.

Further, we have introduced a new algorithm which outperforms other preprocessing methods. It copes better with audio signals of various volume levels, which is of major importance for onset detection in real-time scenarios.

Apart from that, machine learning techniques like neural network based methods are much more robust against volume changes in online scenarios and are the methods of choice if enough training data is available.

6. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Funds (FWF) under the projects P22856-N23, TRP-109 and Z159 “Wittgenstein Award”. For this research, we have made extensive use of free software, in particular python and GNU/Linux. Further, we are grateful to the authors Bello and Holzapfel for making their onset dataset publicly available.

7. REFERENCES

[1] MIREX 2011 onset detection results. `http://nema.lis.illinois.edu/nema_out/mirex2011/results/aod/`.

[2] 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 Speech and Audio Processing*, 13(5):1035–1047, 2005.

[3] J. P. Bello, C. Duxbury, M. Davies, and M. B. Sandler. On the use of phase and energy for musical onset detection in the complex domain. *IEEE Signal Processing Letters*, 11(6):553–556, 2004.

[4] S. Böck, A. Arzt, F. Krebs, and M. Schedl. Online real-time onset detection with recurrent neural networks. In *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx)*, 2012.

[5] P. Brossier, J. P. Bello, and M. D. Plumbley. Real-time temporal segmentation of note objects in music signals. In *In Proceedings of the International Computer Music Conference (ICMC)*, 2004.

[6] N. Collins. A comparison of sound onset detection algorithms with emphasis on psychoacoustically motivated detection functions. In *Proceedings of the 118th AES Convention*, pages 28–31, 2005.

[7] R. B. Dannenberg. An on-line algorithm for real-time accompaniment. In *Proceedings of the 1984 International Computer Music Conference*, pages 193–198, 1984.

[8] N. Degara, M. Davies, A. Pena, and M. D. Plumbley. Onset event decoding exploiting the rhythmic structure of polyphonic music. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1228–1239, 2011.

[9] S. Dixon. Onset detection revisited. In *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx)*, pages 133–137, 2006.

[10] C. Duxbury, J. P. Bello, M. Davies, and M. B. Sandler. Complex domain onset detection for musical signals. In *Proceedings of the 6th International Conference on Digital Audio Effects (DAFx)*, 2003.

[11] F. Eyben, S. Böck, B. Schuller, and A. Graves. Universal onset detection with bidirectional long short-term memory neural networks. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 589–594, 2010.

[12] J. Glover, V. Lazzarini, and J. Timoney. Real-time detection of musical onsets with linear prediction and sinusoidal modeling. *EURASIP Journal on Advances in Signal Processing*, 2011(1):1–13, 2011.

[13] A. Holzapfel, Y. Stylianou, A.C. Gedik, and B. Bozkurt. Three dimensions of pitched instrument onset detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1517–1527, 2010.

[14] A. Klapuri. Sound onset detection by applying psychoacoustic knowledge. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 6, pages 3089–3092, 1999.

[15] A. Lacoste and D. Eck. A supervised classification algorithm for note onset detection. *EURASIP Journal on Applied Signal Processing*, pages 153–153, 2007.

[16] P. Masri. *Computer Modeling of Sound for Transformation and Synthesis of Musical Signals*. PhD thesis, University of Bristol, UK, 1996.

[17] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *The Journal of the Acoustical Society of America*, 103(1):588–601, 1998.

[18] D. Stowell and M. D. Plumbley. Adaptive whitening for improved real-time audio onset detection. In *Proceedings of the International Computer Music Conference (ICMC)*, 2007.

STRUCTURE-BASED AUDIO FINGERPRINTING FOR MUSIC RETRIEVAL

Peter Grosche[1,3], **Joan Serrà**[4], **Meinard Müller**[2,3], **and Josep Ll. Arcos**[4]
[1] Saarland University, [2] Bonn University, [3] MPI Informatik
[4] Artificial Intelligence Research Institute (IIIA-CSIC)
{pgrosche,meinard}@mpi-inf.mpg.de, {jserra,arcos}@iiia.csic.es

ABSTRACT

Content-based approaches to music retrieval are of great relevance as they do not require any kind of manually generated annotations. In this paper, we introduce the concept of *structure fingerprints*, which are compact descriptors of the musical structure of an audio recording. Given a recorded music performance, structure fingerprints facilitate the retrieval of other performances sharing the same underlying structure. Avoiding any explicit determination of musical structure, our fingerprints can be thought of as a probability density function derived from a self-similarity matrix. We show that the proposed fingerprints can be compared by using simple Euclidean distances without using any kind of complex warping operations required in previous approaches. Experiments on a collection of Chopin Mazurkas reveal that structure fingerprints facilitate robust and efficient content-based music retrieval. Furthermore, we give a musically informed discussion that also deepens the understanding of this popular Mazurka dataset.

1. INTRODUCTION

The rapidly growing corpus of digitally available audio material requires novel retrieval strategies for exploring large collections and discovering music. One outstanding instance of content-based music retrieval is *query-by-example*: Given a query in the form of an audio recording (or just a short fragment of it), the goal is to retrieve all documents from a music collection that are somehow similar or related to the query. In this context, the notion of similarity used to compare different audio recordings (or fragments) is of crucial importance and largely depends on the respective application. Typical similarity measures assess timbral, melodic, rhythmic, or harmonic properties [2].

A further key aspect of music is its structure. Indeed, the automatic extraction of structural information from music recordings constitutes a central research topic within the area of music information retrieval [10]. One goal of structure analysis is to split up a music recording into segments and to group these segments into musically meaningful categories, such as chorus or verse. The structure is a highly characteristic property for many musical styles. Folk songs and children songs, for example, typically exhibit a strophic form, where one tune is repeated over and over again with changing lyrics. Popular music typically consists of a number of repeating verses connected by a refrain. In classical music, the structure (or musical form) is often more complex and offers more variability.

Besides being characteristic for a certain musical style, the structure and, in particular, the relative duration of its elements is also a good descriptor for a specific piece of music—irrespective of specific realizations or performances. Furthermore, the structure is invariant to changes in instrumentation or key and therefore allows for identifying different performances of the same piece. So far, only a few approaches exist that exploit structural similarity to facilitate music retrieval [1, 4, 6, 7]. Typically, these approaches are based on *self-similarity matrices* (SSMs) which in general play an important role for analyzing musical structures [10]. For computing an SSM, an audio recording is first transformed into a sequence of feature vectors and then all elements of the sequence are compared in a pairwise fashion using a local similarity measure. Repeating patterns in the feature sequence appear as parallel paths in the SSM, see Figure 1a. Revealing structural properties, SSMs can in turn be used for analyzing structural similarities of performances. To this end, one requires a similarity measure that compares entire SSMs while being invariant to temporal variations. In [6, 7], the SSMs are compared using a similarity measure that is based on a two dimensional version of dynamic programming. The approach proposed by Bello [1] is also based on SSMs, but employs a *normalized compression distance* (NCD) to assess their similarity, without requiring any alignment operations. Originally proposed for comparing protein structures in bioinformatics, the NCD can be regarded as a measure of the *information distance* of two objects where the Kolmogorov complexity is approximated using a standard compression algorithm, see [1].

Inspired by the work of Bello, we describe in this paper a simple yet effective approach for measuring structural similarities of music recordings. As first contribution, we introduce the concept of *structure fingerprints* which are compact structural descriptors of music recordings. Analogous to [1, 6, 7], our fingerprints are also derived from self-similarity matrices while avoiding any explicit determination of structure. Specifically, we use a bivariate variant of a Parzen-Rosenblatt kernel density estimation method for representing a given SSM by a probability density function (pdf) [13]. This has the desired effect of smoothing out temporal variations in the performances. As a result, unlike previous approaches, we do not require any complex distance measure. Instead, recordings can be compared efficiently using, e. g., the Euclidean distance between fingerprints. As second contribution, we report on extensive experiments using a large collection of Chopin Mazurkas. In particular, we show that structure fingerprints facilitate content-based music retrieval solely based on structural information and exhibit a high degree of robustness against performance variations. This makes the presented approach particularly suited for supporting traditional retrieval systems that assess harmonic similarities [2, 5, 8, 12]. Finally, as third contribution, we provide a musically informed discussion of problematic pieces and recordings which also deepens the understanding of the Mazurka dataset.

The remainder of this paper is organized as follows. In Section 2, we introduce our approach to computing structure fingerprints. Then, in Section 3, we describe our retrieval experiment and give a quantitative as well as musically informed discussion of the results. Conclusions are given in Section 4.

2. STRUCTURE FINGERPRINTS

In this section, we introduce our strategy for computing structure fingerprints that capture characteristics of a musical piece and, at the same time, are invariant to properties of a specific performance. We first introduce the underlying feature representation (Section 2.1) and the SSM variant (Section 2.2). In particular, we introduce various enhancement strategies that absorb a large degree of temporal and spectral variations. Then, in Section 2.3 we explain in detail how the fingerprints are derived from the SSMs.

2.1 Feature Representation

We first convert a given music recording into a sequence of chroma features, which have turned out to be a powerful mid-level representation for relating harmony-based music [1, 2, 5, 8, 10, 12]. The term *chroma* refers to the elements of the set $\{\mathrm{C}, \mathrm{C}^\sharp, \mathrm{D}, \ldots, \mathrm{B}\}$ that consists of the twelve pitch classes as used in Western music notation. Representing the short-time energy content of the signal relative to the pitch classes, chroma features do not only account for the close octave relationship in harmony, but also introduce a high degree of robustness to variations in timbre and instrumentation [8]. Furthermore, normalizing the features makes them invariant to dynamic variations.

In our implementation, we use a variant of chroma features referred to as CENS[1] features [8]. As main advantage, CENS features involve an additional temporal smoothing and downsampling step which leads to an increased robustness of the features to local tempo changes [8]. This property is crucial for obtaining structure fingerprints that are invariant to local variations in the performances. In our implementation, the resulting feature representation has a resolution of 1 Hz (one feature per second), where each vector is obtained by averaging over 4 seconds of the audio.

[1] *Chroma Energy Normalized Statistics* features, provided by the Chroma Toolbox www.mpi-inf.mpg.de/resources/MIR/chromatoolbox

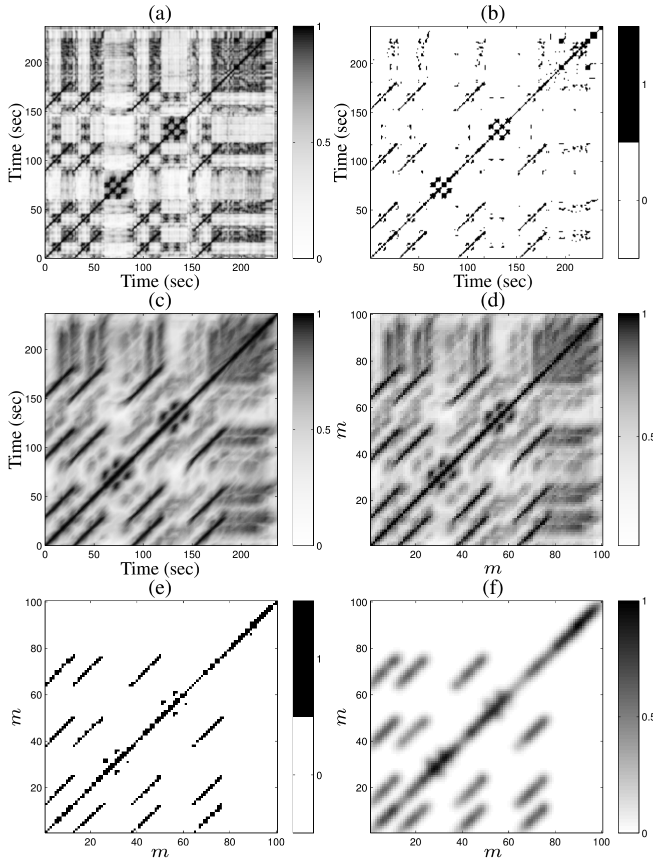


Figure 1: Computing structure fingerprints for an Ashkenazy (1981) performance of Chopin's Mazurka Op. 56 No. 1 with the musical form $A_1A_2BA_3CA_4D$. **(a)** SSM computed from CENS features. **(b)** Thresholded variant of (a) ($\kappa = 10$). **(c)** Path-structure enhanced SSM ($L = 12$). **(d)** Resampled SSM $\mathbf{S}_M^{\mathrm{fix}}$ ($M = 100$). **(e)** Thresholded variant of (d) ($\kappa = 10$). **(f)** Structure fingerprints (pdf estimated from (e), $\ell = 10$).

2.2 Self-Similarity Matrix

Let $X := (x_1, x_2, \ldots, x_N)$ be the feature sequence consisting of N normalized CENS features. Furthermore, let s be a similarity measure that allows for comparing two CENS vectors. In the following, we use the inner product between the normalized CENS vectors (cosine measure, which yields similarity values between 0 and 1). Then, a *self-similarity matrix* (SSM) is obtained by comparing all elements of X in a pairwise fashion [10]:

$$\mathbf{S}(n, m) := \mathbf{s}(x_n, x_m)$$

for $n, m \in [1 : N] := \{1, 2, \ldots, N\}$.

Figure 1a shows the resulting SSM for an Ashkenazy (1981) performance of Chopin's Mazurka Op. 56 No. 1

having the musical form $A_1A_2BA_3CA_4D$. The SSM reveals the repetitive structure (four repeating A-parts) in the form of diagonal paths of high similarity (dark colors).

2.2.1 Path-Structure Enhancement

Musical variations often lead to fragmented path structures of $\mathbf{S}$. To alleviate this problem, various matrix enhancement strategies have been proposed [1, 9, 12] with the idea to apply a smoothing filter along the direction of the main diagonal. This results in an emphasis of diagonal information and a denoising of other structures, see Figure 1c. In the presence of significant tempo differences, however, simply smoothing along the main diagonal may smear out important structural information. To avoid this, we use a strategy that filters the SSM along multiple gradients as proposed in [9]. In our experiments, we compute a simple moving average in windows corresponding to L seconds of audio and use five gradients covering tempo variations of -30 to $+30$ %. In the following, the enhanced SSM is again denoted as $\mathbf{S}$.

2.2.2 Resampling

A high degree of local tempo differences is already absorbed by the smoothing of the CENS features and the path-structure enhancement. Global differences in tempo of different performances of a piece of music, however, lead to SSMs that have different sizes. For deriving structure fingerprints that are invariant to such tempo differences, we apply the idea of [5] and introduce a simple resampling step that converts the $N \times N$ similarity matrix $\mathbf{S}$ into an $M \times M$ similarity matrix $\mathbf{S}_M^{\text{fix}}$, with M fixed to a suitable value:

$$\mathbf{S}_M^{\text{fix}}(n,m) := \mathbf{S}(\lfloor n\tfrac{N}{M}\rceil, \lfloor m\tfrac{N}{M}\rceil)$$

for $m, n \in [1 : M]$, where $\lfloor\cdot\rceil$ denotes rounding to the nearest integer. [2] Figure 1d shows an example for $\mathbf{S}_M^{\text{fix}}$.

2.2.3 Thresholding

We finally process the SSMs by suppressing all values that fall below a threshold. Analogous to [1, 12], we choose the threshold in a relative fashion by keeping $\kappa\,\%$ of the cells having the highest score. The motivation for this thresholding step is that only a certain amount of the cells of the SSM are expected to encode relevant structural information. The thresholding can then be regarded as some kind of denoising, where only relevant paths are retained, see Figure 1e. In the following, the resulting thresholded, resampled, and path-structure enhanced SSM is denoted as $\hat{\mathbf{S}}_M^{\text{fix}}$. Figure 1e also emphasizes the importance of the path-structure enhancement, as directly applying the thresholding operation on the original SSM does not lead to the desired denoising effect, see Figure 1b.

2.3 Probability Density Estimation

The four repeating A-parts of our Mazurka example are clearly revealed by $\hat{\mathbf{S}}_M^{\text{fix}}$ in the form of diagonal paths, see

[2] In our experiments, using linear or cubic interpolation did not lead to any improvements.

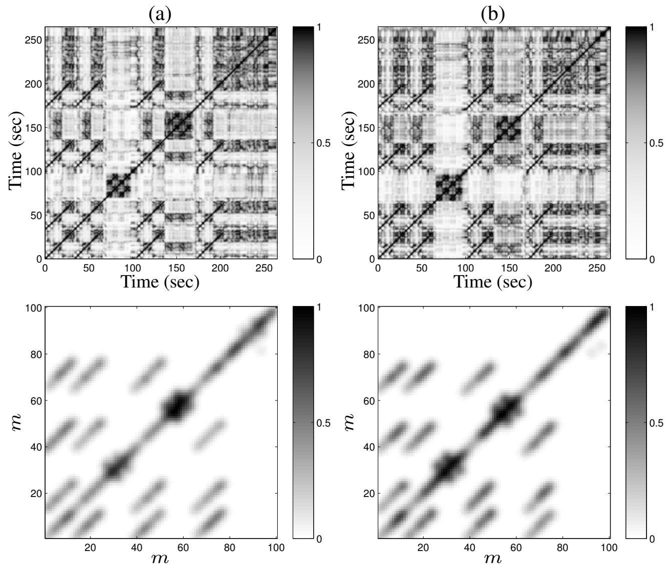


Figure 2: Original SSMs **(top)** and structure fingerprints **(bottom)** for two performances of Chopin's Mazurka Op. 56 No. 1. **(a)** Rubinstein (1966) and **(b)** Kushner (1989).

Figure 1e. However, as the structural information is contained in only a few cells of the thresholded SSM (in other words, the resulting matrix is sparse), small temporal variations in performances may lead to large distances when directly comparing these matrices in a pointwise fashion. As a result, some kind of tolerance to temporal variations is required in the similarity measure, as e. g., introduced by the similarity measures based on dynamic programming used in [6, 7] and the NCD used by Bello in [1].

Avoiding the additional complexity of such techniques, we consider $\hat{\mathbf{S}}_M^{\text{fix}}$ as a bivariate random sample of coordinates (n, m) for $n, m \in [1 : M]$ and our goal is to estimate the probability density function (pdf) producing this SSM. [3] The underlying assumption is that the pdf corresponds to the musical structure of the piece and that the bivariate random samples we observe are affected by variations in the realization of a specific performance. Analogous to [11], we employ a Parzen-Rosenblatt kernel density estimation method [13] that consist in convolving $\hat{\mathbf{S}}_M^{\text{fix}}$ with a two-dimensional Gaussian kernel of size ℓ. As a result, temporal variations in the performances are smoothed out. The choice of the value ℓ constitutes a trade-off between fingerprint characteristic (small value) and robustness to temporal variations (large value).

The resulting fingerprints (see Figure 1f) are an $M \times M$ representation [4] of the musical structure that features a high degree of robustness against properties of a specific performance. Figure 2 shows two further examples of fingerprints for the Mazurka Op. 56 No. 1.

3. STRUCTURE-BASED RETRIEVAL

In this section, we show how the structure fingerprints (SF) can be used to facilitate structure-based music retrieval.

[3] In the following, we use the term *pdf*, although for discrete random variables, the term *probability mass function* would be more appropriate.

[4] Note that this matrix is symmetric and only $M(M+1)/2$ entries are needed for representing the fingerprints.

Method	Dist.	Dataset	P	Sync.	MAP	T [sec]
Bello [1]	NCD	`Bello`	2919	No	0.767	>1000
SF	KL	`ORG`	2793	No	0.819	66.45
SF	ED	`ORG`	2793	No	0.816	0.58
SF	ED	`MOD`	2792	No	0.828	0.58
SF	ED	`MOD`	2792	Yes	0.958	0.58

Table 1: Overview of the results obtained for different methods and datasets. *Dist.* denotes the distance measure used, P the number of performances in the dataset, and T the run-time in seconds for computing $P \times P$ distances. [6] See Section 3.4 for a description of the dataset `MOD` and the column *Sync.* (indicating whether synchronized fingerprints are used).

We first describe the collection of Chopin Mazurkas (Section 3.1) and the retrieval scenario (Section 3.2). Then, we continue with a quantitative evaluation (Section 3.3) and give a musically informed discussion (Section 3.4).

3.1 Mazurka Collection

In our experiments, we use an audio collection comprising many recorded performances for each of the 49 Mazurka by Frédéric Chopin. Since different performances of a Mazurka typically share the same structure, this collection is a good choice for evaluating structural similarities. The dataset was assembled by the Mazurka Project [5] and has also been used by Bello in [1]. Note, however, that there are differences between our dataset (denoted as `ORG` in the following) and the one used in [1] (denoted as `Bello`). Actually, the datasets constitute a snapshot at different stages in the assembly process of the Mazurka Project which also results in a different number of performances (2793 for `ORG` and 2919 for `Bello`, see Table 1).

3.2 Retrieval Scenario

Using this dataset, we evaluate our structure fingerprints (SF) in a document-level retrieval scenario as in [1]. Given one performance of a Mazurka as query, the goal is to retrieve all other performances of the same Mazurka from the given dataset. To this end, we first compute the fingerprints for all P performances of the dataset. Using a suitable distance measure, we then derive the $P \times P$ matrix of pairwise distances between all performances, see Figure 6a. As the structure fingerprints are represented as densities, a natural choice of distance measure is the Kullback-Leibler divergence (KL). Additionally, in our experiments, we also use a simple Euclidean distance (ED). Finally, we rank the result with respect to ascending distances and express the retrieval accuracy by means of the *mean average precision* (MAP) measure as in [1, 12].

3.3 Quantitative Evaluation

First, we give a quantitative discussion of the results. Table 1 shows overall MAP values for the different methods and datasets. In [1], Bello reported $\text{MAP} = 0.767$ using his approach based on the NCD. Using the parameters $L = 10, \kappa = 20, M = 50, \ell = 5$ and the KL divergence, our approach leads to comparable, if not even slightly better results ($\text{MAP} = 0.819$). Note, however,

[5] mazurka.org.uk

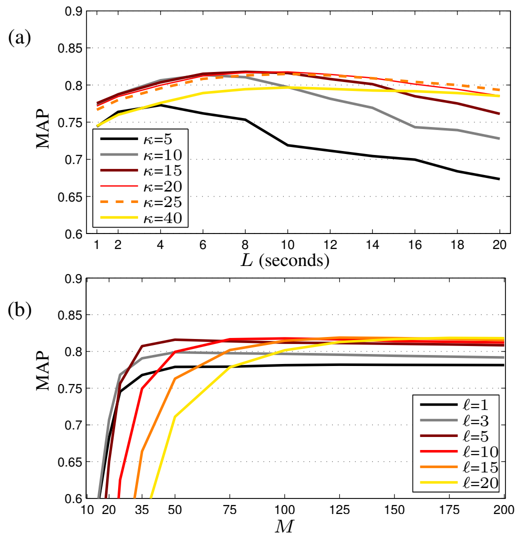


Figure 3: Parameter evaluation using `MOD` and Euclidean distances (ED). MAP values for different values of **(a)** the smoothing parameter L and threshold κ ($M = 100$, $\ell = 10$) and **(b)** of the fingerprint size M and kernel size ℓ ($L = 12$ sec, $\kappa = 20$).

that the results are not directly comparable due to the differences in the datasets. The results are insofar surprising, as our approach is not only conceptually much simpler, but also more explicit and, as it turns out, much more efficient. The last column of Table 1 indicates the run-time in seconds for computing the matrix of $P \times P$ pairwise distances. [6] Without knowing exact numbers for the NCD, our approach using KL seems to be at least one order of magnitude faster than [1]. Actually, when using the Euclidean distance (ED) instead of KL, the run-time of our approach can be improved significantly by two orders of magnitude (resulting in a run-time of just 0.58 seconds for computing all $P \times P$ distances), without any degradation of retrieval accuracy ($\text{MAP} = 0.816$).

We now continue with an evaluation of different parameter settings using ED (using KL lead to very similar findings). Figure 3a shows MAP values obtained on `ORG` as a function of the temporal smoothing parameter L (in seconds) and the relative threshold κ, see Section 2.2. Appropriate values for L constitute a trade-off between enhancement capability and level of detail. For the Mazurkas, a smoothing of 6-12 seconds seems to be reasonable, the actual choice of the parameter, however, is not crucial. For example, fixing $\kappa = 15$, one obtains $\text{MAP} = 0.815$ for $L = 6$ and $\text{MAP} = 0.816$ for $L = 10$. The threshold value κ constitutes a trade-off between retaining relevant structural information and denoising the SSMs. For the Mazurkas, 10%-25% seems to be a good compromise for capturing the repetitive structure. Again, the exact value is not crucial. For example, fixing $L = 6$, one obtains $\text{MAP} = 0.809$ for $\kappa = 25$ and $\text{MAP} = 0.814$ for $\kappa = 10$.

Figure 3b shows MAP values as a function of the fingerprint size M for different settings of the kernel density parameter ℓ. Interestingly, the size of the structure finger-

[6] Using a vectorized MATLAB implementation of ED, a C/C++ implementation of KL, and an Intel Xeon E3-1225 CPU. Run-times for the NCD are estimated from the indicators given in [1] and own experiments.

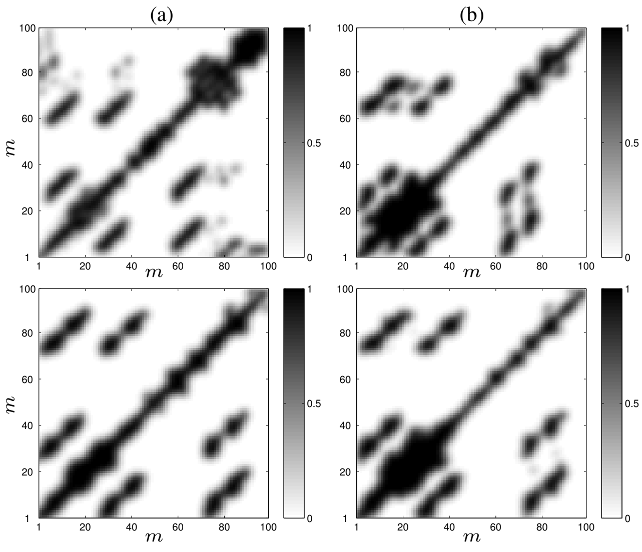


Figure 4: Structure fingerprints **(top)** and synchronized structure fingerprints **(bottom)** for performances of Chopin's Mazurka Op. 24 No. 2. **(a)** Merzhanov (2004) with applause at start and end of recording. **(b)** Smith (1975) with silence at the end.

prints can be reduced to $M = 50$ or even $M = 35$, while still retaining a high retrieval accuracy. The ratio of M and ℓ, however, is of crucial importance as it constitutes a trade-off between fingerprint characteristic and robustness against temporal variations in the performances. The settings $M = 50, \ell = 5$, and $M = 100, \ell = 10$, and $M = 200, \ell = 20$ yield almost identical retrieval results (MAP $= 0.816$, MAP $= 0.818$, and MAP $= 0.819$, respectively). Decreasing the size of the fingerprints, however, has the advantage of reducing the computational load.

Aside from the robustness to actual parameter settings, our approach turned out to be rather robust to implementation details. For example, very similar results were obtained by using, e. g., Cosine, Hellinger, and Battacharyya distances between SFs. Even an alternative implementation using different chroma features as well as delay coordinates and recurrence plots (instead of the enhanced SSMs) similar to [1, 11, 12], lead to almost identical results. This also indicates that our concept is generalizable.

3.4 Musically Informed Discussion

Our fingerprint-based approach allows for detecting musically interesting phenomena and inconsistencies in the Mazurka collection. A careful investigation of the retrieval results revealed three phenomena. Firstly, we discovered that there are 67 recordings in the dataset that are incorrectly assigned to one of the Mazurkas, although they actually are performances of another Mazurka.[7] Another recording of the collection did not correspond to any of the Mazurkas.[8] We corrected these errors and denote the modified dataset MOD. Repeating the retrieval experiment using the 2792 performances of MOD, the MAP value increases to 0.828, see Table 1 (fourth row).

[7] A majority (51 of the 67 recordings) affects Op. 41 consisting of four Mazurkas (No. 1 to No.4), where a permutation of the assigned numbers occurs.

[8] Labeled as a Rosenthal (1935) performance of Op. 50 No. 2.

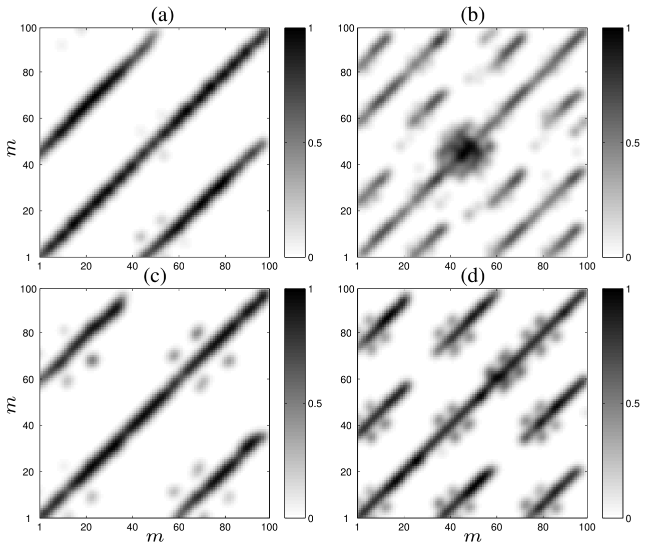


Figure 5: Structure fingerprints for four performances with differing structure of Chopin's Mazurka Op. 68 No. 4 ($L = 12$, $\kappa = 20$, $\ell = 10$, $M = 100$). **(a)** Niedzielski (1931). **(b)** Katin (1996). **(c)** Rubinstein (1952). **(d)** Rubinstein (1966).

Secondly, it turned out that many incorrectly retrieved performances exhibit a long passages of applause, silence, or spoken moderation at the beginning and/or end. Actually, such passages can be regarded as additional structural elements. As a result, the structure of these performances does not match to the structure of the other performances of the same Mazurka, see Figure 4. To quantify this phenomenon, we use music synchronization techniques [3, 8] for identifying musically corresponding time positions in all versions of a Mazurka and use this information to warp the fingerprints to a common time line. For additional segments appearing in one performance, there are no corresponding time positions in the other performances. As a result, such segments are basically not reflected in the resulting *synchronized fingerprints*, see Figure 4.[9] Using synchronized fingerprints to exclude the additional segments, we repeat our experiment using MOD and obtain MAP $= 0.958$, see Table 1 (last row).

The third phenomenon detected during our experiments are structural differences in the recordings. For instance, some pianists do not strictly stick to the score when performing a piece but omit (or sometimes even introduce) repetitions. Obviously, these structural differences lead to high distances as shown in Figure 6b for the Mazurka Op. 56 No. 1, where eight of the 42 performances exhibit a different structure.[10] The prime example for this effect is Mazurka Op. 68 No. 4, where the last bar in the score contains the marking *D. C. dal segno senza fine*. However, there is no *fine* marked in the score that would tell the pianist where to end. As a result, a performer may repeat the piece as often as he or she wants. This leads to many versions of the piece that differ significantly in structure as also revealed by the respective pairwise distances shown in Figure 6e. Figure 5 shows the fingerprints of four such

[9] This strategy has a similar effect as using a distance measure based on dynamic programming, as proposed in [6, 7].

[10] Actually, all eight musicians omit a repetition of the A-part, leading to the form $A_1BA_2CA_3D$ instead of $A_1A_2BA_3CA_4D$.

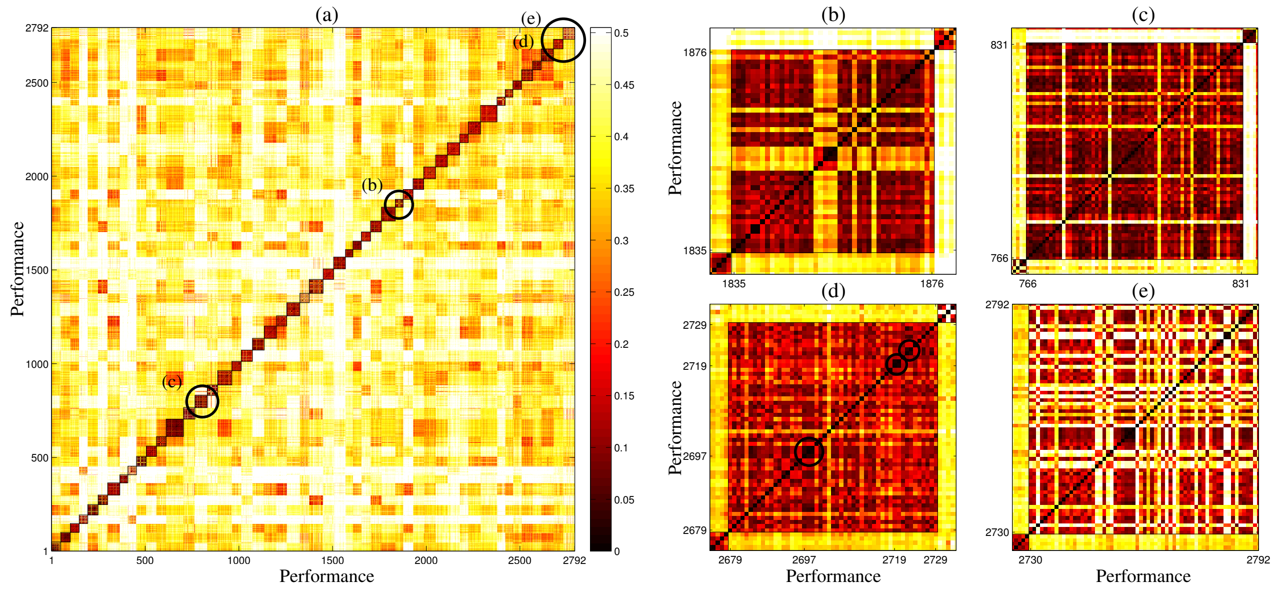


Figure 6: **(a)** Matrix of pairwise Euclidean distances for the 2792 performances of `MOD`. **(b)** Detail of the 42 performances of Op. 56 No. 1, see also Figure 2. **(c)** Detail of the 66 performances of Op. 24 No. 2, see also Figure 4. **(d)** Detail of the 51 performances of Op. 68 No. 3. **(e)** Detail of the 63 performances of Op. 68 No. 4, see also Figure 5.

versions, which, obviously, cannot be retrieved by a purely structure-based retrieval approach.

On the other hand, during our experiments we discovered performances that exhibit a surprisingly low distance, see, e. g., the squares of low distance on the main diagonal in Figure 6d. The low distance between the performances 2697-2699 is actually known as the “Hatto effect”: recordings released under the name of the pianist Joyce Hatto in 1993 (2697) and 2006 (2698) that are actually time-scaled copies of a 1988 recordings of Eugen Indjic (2699). Similarly, some performances appear repeatedly in the dataset as they were released multiple times. Examples for this effect are performances 2719 and 2720 (Rubinstein) as well as 2722 and 2723 (Smidowicz).

4. CONCLUSION

The concept of structure fingerprints presented in this paper allows for retrieving music recordings solely based on structural information. Using a combination of suitable enhancement strategies, our approach is robust as well as efficient. Furthermore, as our experiments reveal, the results obtained by our approach are at least comparable to state-of-the-art approaches without relying on complex distance measures. As further advantage of our approach, just using Euclidean distances between fingerprints opens the possibility of exploiting efficient index-based methods such as locality-sensitive hashing to scale the approach to even larger datasets. We showed that our methods are suited for systematically analyzing structural properties of entire music collections, thus deepening the musical understanding of the data. Obviously, the limits of structure-based retrieval are reached when the assumption of global structural correspondence between performances is violated.

Acknowledgments: The work by P. Grosche und M. Müller has been supported by the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University and the German Research Foundation (DFG MU 2686/5-1). J. Serrà and J. L. Arcos acknowledge 2009-SGR-1434 from Generalitat de Catalunya, TIN2009-13692-C03-01 from the Spanish Government, and EU Feder funds. J. Serrà also acknowledges JAEDOC069/2010 from Consejo Superior de Investigaciones Científicas.

5. REFERENCES

[1] J. P. Bello. Measuring structural similarity in music. *IEEE Trans. on Audio, Speech and Language Processing*, 19(7):2013–2025, 2011.

[2] M. A. Casey, R. Veltkap, 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, 2008.

[3] S. Ewert, M. Müller, and P. Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[4] J. Foote. ARTHUR: retrieving orchestral music by long-term structure. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, USA, 2000.

[5] P. Grosche and M. Müller. Toward characteristic audio shingles for efficient cross-version music retrieval. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 473–476, Kyoto, Japan, 2012.

[6] T. Izumitani and K. Kashino. A robust musical audio search method based on diagonal dynamic programming matching of self-similarity matrices. In *Proc. of the International Conference on Music Information Retrieval (ISMIR)*, pages 609–613, Philadelphia, USA, 2008.

[7] B. Martin, M. Robine, and P. Hanna. Musical structure retrieval by aligning self-similarity matrices. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 483–488, Kobe, Japan, 2009.

[8] M. Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.

[9] M. Müller and F. Kurth. Enhancing similarity matrices for music audio analysis. In *Proc. of the Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 437–440, Toulouse, France, 2006.

[10] J. Paulus, M. Müller, and A. P. Klapuri. Audio-based music structure analysis. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 625–636, Utrecht, The Netherlands, 2010.

[11] J. Serrà, M. Müller, P. Grosche, and J. L. Arcos. Unsupervised detection of music boundaries by time series structure features. In *Proc. of the AAAI Int. Conf. on Artificial Intelligence*, 2012. In Press.

[12] J. Serrà, X. Serra, and R. G. Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11(9):093017, 2009.

[13] J. S. Simonoff. *Smoothing Methods in Statistics*. Springer, 1996.

BRIDGING PRINTED MUSIC AND AUDIO THROUGH ALIGNMENT USING A MID-LEVEL SCORE REPRESENTATION

Özgür İzmirli, Gyanendra Sharma
Center for Arts and Technology
Computer Science Department
Connecticut College
{oizm, gsharma}@conncoll.edu

ABSTRACT

We present a system that utilizes a mid-level score representation for aligning printed music to its audio rendition. The mid-level representation is designed to capture an approximation to the musical events present in the printed score. It consists of a template based note detection front-end that seeks to detect notes without regard to musical duration, accidentals or the key signature. The presented method is designed for the commonly used grand staff and the approach is extendable to other types of scores. The image processing consists of page segmentation into lines followed by multiple stages that optimally orient the lines and establish a reference grid to be used in the note identification stage. Both the audio and the printed score are converted into compatible frequency representations. Alignment is performed using dynamic time warping with a specially designed distance measure. The insufficient pitch resolution due to the reductive nature of the mid-level representation is compensated by this pitch tolerant distance measure. Evaluation is carried out at the beat level using annotated scores and audio. The results demonstrate that the approach provides an efficient and practical alternative to methods that rely on symbolic MIDI-like information through OMR methods for alignment.

1. INTRODUCTION

Music can be represented in mainly three forms: audio, symbolic (such as MIDI) and printed. Historically these forms of data have remained disparate in archives and have only been associated through metadata. More recently the field of music information retrieval has been actively exploring ways to bridge the content across their different forms of existence. MIR systems dealing with large music collections depend on basic operations such as searching, matching and alignment. These operations are required to not only work with audio or MIDI formats but they should be capable of handling multi-format data including printed and hand-written scores. Finding matches and similarities across representations is of interest because these will pave the way to building integrated systems that have broad implications in research and education.

Traditional libraries contain vast collections of music on paper as well as recorded audio but lack the fine-level connection between the two formats. Incorporation of methods that connect the different representations can result in applications being more capable and multimodal. Some applications include: score retrieval by audio example; structure and harmonic analysis by audio input; transcription of performance parameters from audio superimposed onto existing scores; score following in the literal sense – following the music automatically on the printed score.

Audio is sonically rich but sound mixtures are hard to analyze and separate automatically. Symbolic data on the other hand represents music very efficiently at the note level but contains very little timbral and expressive information. The visual nature of the printed score allows musicians to read, experience and analyze music in different ways and is an indispensible part of musicians' every day experience. Each representation type has its own advantages and by connecting them in meaningful ways we can achieve greater musical understanding as well as convenient access to many forms of representation. The different kinds of information in these representations can greatly leverage our overall understanding and aid us in searching with multiple perspectives. Today, conversion between these forms presents many challenges and can be performed with varying levels of success. It is, however, easier to bridge collections in different forms through fine-grain alignment.

In this paper, we present an approach to aligning audio and printed representations of music using a mid-level score representation. We will use the term *score* to denote the sheet or printed version throughout the paper and note that it is different from the usage in score following work where it is commonly used to depict the MIDI-like symbolic sequence. The proposed mid-level representation enables us to capture sufficient pitch and score position information to guide the alignment process. Key signatures and accidentals are ignored in the recognition and therefore a tolerant distance measure that compensates for this shortcoming is proposed. In the remainder of the paper the next section outlines related and previous work. Section 3 presents the mid-level representation which is followed by a section in which a distance measure is defined. We finally present an evaluation of the method on a small set of piano music and close with concluding remarks.

2. RELATED WORK

Since the introduction of optical music recognition (OMR), multiple works have been carried out in mapping and aligning different music representations, namely, the music score, audio recordings and MIDI. Multiple approaches have been employed to build state-of-the-art audio-to-score alignment algorithms. Some are based on graphical and statistical models such as the ones in [3,8,16] whereas some use the Dynamic Time Warping (DTW) algorithm to align the sequences of features extracted from both the audio and the score as in [7,12]. Work done in [13] carry out a multi-pass algorithm where they propose a method which estimates the onset times of individual notes in a post processing step to obtain an accurate audio-to-score alignment. Earlier works in audio-to-score alignment such as [14] employ the DTW algorithm and generate spectral approximations from the symbolic form in order to compute the local distance measures for the DTW.

In [8] Joder et al. propose a statistical model for music-to-symbolic score alignment where a hidden state model uses two features: chroma vectors, to model pitch content of the signal and spectral flux, to model note onsets. The approach employed in this work claims to have achieved a very precise alignment with a low complexity compared to other DTW systems. More recent work by Cont [3] discusses the use of hierarchical hidden Markov models for online and real-time audio-to-score alignment.

All these works approach the problem of alignment based on fully-notated MIDI score. To the authors' knowledge, alignment work solely based on the music sheet and its corresponding audio recordings without the use of intermediate MIDI format have not been formally used. Work on mapping, synchronization and identification of the music with audio recordings has been carried out by [5,6,11]. In [5] the authors discuss two different approaches in identifying the corresponding sections of an audio interpretation of a musical piece given the sections of the score for the same piece. The first approach where it is assumed that the performance sequence is known uses a semi-automatic approach using synchronization whereas, the second approach where the performance sequence is unknown uses matching techniques. However, OMR is used to obtain the symbolic score before employing any of the identification techniques.

Work has also been done in aligning semi-improvised music with its lead sheet [4]. This is generally more difficult as the lead sheet specifies only essential elements such as the melody, harmony, and a basic musical form. This work also stems from using the symbolic data obtained after the OMR techniques on the score.

A lot of work has been focused on solving specific problems of the OMR such as staff line detection [1] and recognizing musical symbols. Recently, in [17] the authors have employed template matching and grammatically formulated top-down models as a means of performing OMR on scanned sheet music. Since the purpose of this paper is precisely not to perform detailed OMR we refer the interested reader to two overviews of the state-of-the-art in optical music recognition [2, 18].

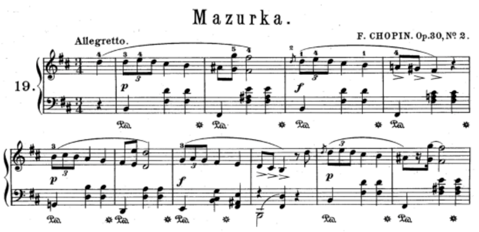
When audio is in the mix, chroma based representations are often used for alignment. In [9] the authors maintain that "chroma vectors drawn from representations using a logarithmic frequency scale are the most efficient features, and lead to a good precision, even with a simple alignment strategy." Here, we not only utilize a chroma based representation obtained from audio analysis but also create one from the score.

3. MID-LEVEL REPRESENTATION

In this work we restrict our method to pieces using the grand staff in which a system consists of the top staff notated with the treble clef and the lower staff with the bass clef. We have been using scanned scores from the International Music Score Library Project (IMSLP). These images are particularly challenging due to the fact that they contain skew within the page, have different print styles, their original resolutions vary and they are quite noisy. Our purpose is to perform some basic image processing operations on the digitized score and extract the relevant sections to arrive at an intermediate representation that would be useful for alignment. As a first step, prior to any image processing a binarization step is performed using Otzu's method [15] which optimizes the foreground/background classification of pixels through an exhaustive threshold search.

3.1 Overall Page Structure

The first step is to identify the overall page structure in terms of systems. A horizontal projection P_p is calculated by summing the pixel values across the page. This projection is generally a quite good representation to identify line positions in scanned scores that are reasonably straight. We assume that the original rotation of the scanned page produces a projection in which the staff lines are identifiable through local peaks. We can optionally perform an automated page rotation to correct for scanning errors using a procedure similar to the one described in the next section for individual systems. The projection P_p is then smoothed with a truncated Gaussian filter with width equal to a single staff. The position of each staff is determined by peak picking and simultaneously, the positions of the top and bottom lines of each system are determined by finding the local peaks of the unsmoothed projection in the vicinity of the peaks of the smoothed projection. Figure 1 shows part of the original score at the top and the projection resulting from that image below. The projection is aligned with the image of the two systems shown at the top. This process results in fairly reliable vertical position estimates of the systems on a single page. This segmentation is performed for all subsequent pages in the score for the piece in question.



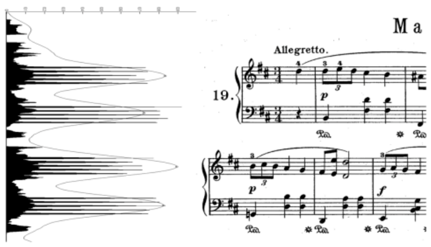


Figure 1. Top: first two systems from a scanned score. Bottom: horizontal projection and Gaussian smoothing of the top figure for locating systems in a page.

3.2 Aligning Systems and Automatic Calibration

We extract systems one by one according to their positions on a page as described above. Each system I_n, runs from C2, two ledger lines below the bottom line in the bass clef, to approximately E6 on the third ledger line above the treble clef. This image is then corrected for optimal rotation by fitting parabolas onto local peaks in the projection. The position of each line is determined by peak picking on the horizontal projection of the extracted system. We observe that the shape of the intensity distribution around each peak is correlated to how well the system is aligned – the narrowest distribution is considered the best rotation for I_n which results in maximally horizontal staff lines. We therefore, fit a least-squares parabola on the points neighboring each peak that exceeds a threshold. Since the parabolas are opening downward we find the rotation angle θ that minimizes the sum of the coefficients of the second degree terms of the parabolas for all 10 peaks. The optimal rotation angle is given by

$$\theta_n = \underset{\theta}{arg\,min}\left(\sum_i \Theta(\Psi(I_n,\theta),i)\right) \qquad (1)$$

where Ψ represents the rotation of I_n by θ and Θ is the coefficient of the second degree term in the parabola equation for local peak i that serves as the relative width estimate in the horizontal projection of the rotated image.

For our purposes the rotation correction for each system turns out to be quite important. We have observed that even systems on the same page can have different rotation and skew values. In order to correctly identify notes, an adaptive reference grid delineating the note ranges is required for each system. In contrast to other approaches, our method does not remove the staff lines because the templates which are taken from actual images already include parts of those lines.

3.3 Compressing the Grand Staff

The process of finding the optimal rotation for each system also allows us to more accurately identify the positions of the lines. Next, we separate each system into two images by cutting it in half with a horizontal line that lies between the lowest line in the treble clef (E4) and the highest line in the bass clef (A3). We then merge the upper and lower halves of each system by multiplying (ORing – with pixel intensity values between 0 and 1) the content such that the positions of C4 in each part coincide. Figure 2 shows the compressed image for the first system given in Figure 1. Note that, for example, the note D4 that originally appears on the lower staff now has the correct position with respect to the upper staff. The reference grid which contains positions for the note boundaries is calculated from the distance between the top and bottom staff lines. Figure 3 shows the grid on top of a fragment of the rotated image. The regions between lines of the grid represent the C major diatonic set regardless of any accidentals that are in use.

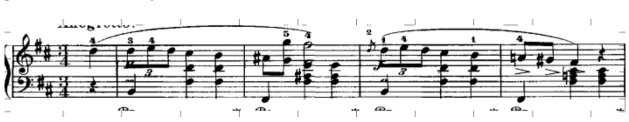


Figure 2. Compressed image of first system in Figure 1.

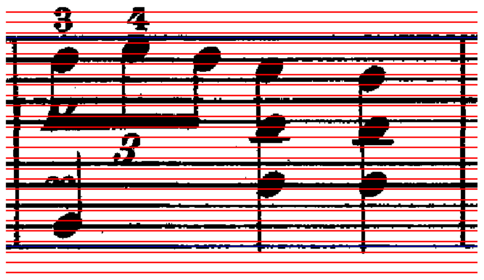


Figure 3. The reference grid for note boundaries superimposed on the optimally rotated image. The space between each pair of lines corresponds to a diatonic note.

3.4 Note Identification

The process of note identification follows a template matching approach. Three templates are constructed: one for a filled-in note head positioned between lines, one for a filled-in note head on a line and one for an empty note head. The templates are slightly larger than the oval note head and include a small portion of the surrounding lines. Their registration point is at the center of the symbol and ideally should fall either on a line or midway between two lines. The only symbols of interest are the note heads and other symbols such as stems, accidentals, rests, clefs, beams etc. are not considered. Notes are found by convolving the optimally rotated system image $\Psi(I_n,\theta_n)$ separately with each of the templates T_l. The original templates are scaled according to the line spacing of the system under consideration. The two images are represented with bipolar encoding (±1) for the convolution. Local peaks indicate matches between a template and a system. We then obtain the set of all recognized notes by the union of notes recognized in all systems in the piece.

$$Q = \bigcup_{n,l} \Gamma\left[\Psi(I_n, \theta_n) * T_l\right] \tag{2}$$

Here * is the 2D convolution operator and Γ[ ] is the function for finding local peaks. Since the note recognition is done without regard to key signature or any preceding accidentals, only the notes corresponding to white keys are found. This process results in a set of recognized notes Q with 2-tuple elements $q_v=(n_v, o_v)$ each with a note index n_v that corresponds to the bins of the chromagram and an onset frame (column) number o_v. An example of the output is shown in Figure 4.

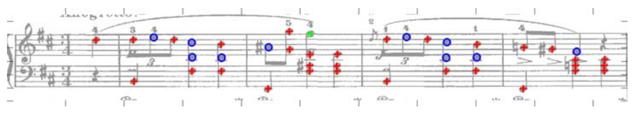


Figure 4. Recognized notes from the image in Figure 2. The original has been lightened and '+' indicates a note head on a line, 'o' between lines and 'x' an unfilled note head.

3.5 Chroma Representation from the Score

Before we define a distance function to establish a relationship between the audio and printed score representations we would like to find the most compatible frame based features that could be practically calculated from each form of the music. On the audio side we calculate an open 'audio chromagram,' A, using constant Q spectral analysis, that is not folded into one octave. The bins represent logarithmically spaced frequency ranges that are each a semitone wide and calculated with respect to a reference of A4=440Hz. We then proceed to construct a similar feature using the notes recognized from the score to form the 'score chromagram,' S. Each recognized note is placed into the chromagram in the bin representing the note and at the corresponding frame. In addition to the fundamental frequency component, the note's harmonics are also added with amplitude $1/h$, where h is the harmonic number and $h=1..H$. All components incur a fixed exponential decay to account for the passage of time, i.e. to not have the same values for the duration of the note and give more weight to the onset. The note model for a given note q_v is represented by a sequence of k-element chroma vectors

$$R^j(q_v) = (r_0^j, r_1^j, r_2^j, \ldots r_{k-1}^j)^T \tag{3}$$

$$r^j_{\varphi(q_v,h)} = \frac{1}{h} e^{-c(j-o_v)}, \quad j \ge o_v \tag{4}$$

where c is the decay rate and o_v is the frame on which the note starts. The function $\varphi(q_v,h)$ is the index of the bin in the chromagram that corresponds to note q_v and harmonic h. The resultant score chromagram is given by the summation of the note events calculated for all recognized notes:

$$S^j = E \circ \sum_{\substack{\forall q_v \in Q, \\ h=1..H}} R^j(q_v) \tag{5}$$

After the summation of note spectra, a spectral weighting is applied to the score chromagram to match its long-term spectral shape with that of the audio. The weighting E is calculated from the audio chromagram by simply averaging it across time and dividing by the maximum element. The operator ∘ denotes the elementwise multiplication of the vector E with each column of the summation that holds the unweighted score chromagram. Figure 5 shows the score chromagram for the notes of Figure 4 and the audio chromagram for the same fragment of music.

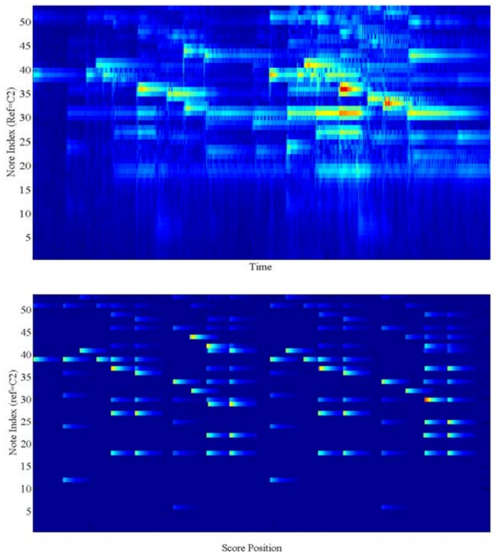


Figure 5. Top: audio chromagram. Bottom: score chromagram obtained from recognized notes as shown in Figure 4.

4. LOCAL DISTANCE AND ALIGNMENT

The defined system would have worked if the piece being analyzed was in C major and a standard distance such as a Euclidean or a cosine distance was being used. However, due to the limitation of the front-end and its notational system which is based on diatonic pitch spacing, these standard distance measures would become progressively meaningless as the keys pick up more accidentals. We therefore define a tolerant distance function between two k-element chroma vectors S (score) at frame i and A (audio) at frame j:

$$S^i = (s_0^i, s_1^i, s_2^i, \ldots s_{k-1}^i)^T, \quad A^j = (a_0^j, a_1^j, a_2^j, \ldots a_{k-1}^j)^T$$

$$b_{i,j} = \sum_{p=1}^{k-2} \frac{max\left(s_p^i a_{p-1}^j, \; s_p^i a_p^j, \; s_p^i a_{p+1}^j\right)}{\left\|S^i\right\| . \left\|A^j\right\|} \tag{6}$$

$$d_{i,j} = 1 - b_{i,j} / b_{max} \tag{7}$$

where b_{max} represents the maximum value of $b_{i,j}$.

The alignment of the score chromagram and the audio chromagram is performed using DTW. The following step size condition constrains the slope of the warping path

$$D_{i,j} = min\left(D_{i-1,j-1}, D_{i-2,j-1}, D_{i-1,j-2}\right) + d_{i,j} \qquad (8)$$

Note that vertical and horizontal moves are not allowed. This ensures that the two sequences move forward at either the same frame rate or twice the other, and also that a single frame in one sequence does not map to multiple frames in the other. This preserves the monotonicity condition of the DTW for our purpose.

5. EVALUATION

We have evaluated the proposed method in various ways. Primarily the evaluation has concentrated on the accuracy of the alignment on the printed score. For this we needed the audio as well as the printed score to be annotated. The scores were taken from IMSLP's Petrucci Library which is a web site that has scanned scores for which the copyright has expired. The audio annotations for the Chopin Mazurkas were taken from The Mazurka Project (http://www.mazurka.org.uk) in which Craig Sapp collected beat-level onsets for different performances of the same piece.

We calculate the alignment accuracy with respect to two frames of reference. The first is the score where the alignment error is reported as a percentage of the staff width. The times of all beats in the audio (given by the ground truth) are mapped to score positions using the warping path produced by the DTW algorithm. The error is calculated by taking the average of the absolute differences between these numbers and the beat locations in the score given by the ground truth. The second frame of reference is the audio where the alignment error is found in seconds. The two measures are similar in nature and are not meant to provide different viewpoints, rather, they give a good sense of the average and maximum errors in the two modalities of experience: visually following the printed score while listening to the performance.

The following parameters were used for all scores and performances in the evaluation. The audio analysis was done with 50 percent overlapped windows of duration 50 milliseconds. Each column in the score chromagram represents a group of pixels in the input image. The number of pixels in a group is calculated separately for each audio file in order to make the number of score frames comparable to the audio frames. The decay rate, *c*, was determined empirically to be on the order of one beat as seen in Figure 5 but will vary from score to score depending on the density of the typesetting. Four harmonics (H) were used for the note model. The range of the note recognition was restricted to the range C2 to E6 and any notes beyond this range were ignored. The scores were scanned at 300 pixels per inch.

Table 1 shows the list of piece/score edition/performer combinations tested. The second column lists the alignment results with respect to the audio. The average absolute error and maximum error figures are given. The alignment error with respect to the score is given in the third column. The error is in pixel real distances on the digitized image. It shows the horizontal distance between the ground truth and result of the alignment as a percentage of the width of the score. It is reported as a percentage to make it independent of image resolution, however, by the same token, it could be affected by the number of measures that the publisher chose to fit in a single line. The same edition has been tested with different performers as well as different pieces from the same editor. In our tests a number of scores with heavy fonts and poor quality images did not produce acceptable alignments mainly due to the errors in the front-end. We observed that these were primarily grouped around certain publishers and that the template matching could be made more adaptive in future work to cater to even wider stylistic variations.

Piece/ Edition	**Av (Max) Err. Audio (seconds)**	**Av (Max) Err. % score width**	**Performer**
Mazurka 30-2 Mikuli	0.24 (1.78)	3.49 (28.16)	Mohovich
Mazurka 30-2 Mikuli	0.34 (2.07)	4.07 (19.13)	Fou
Mazurka 30-2 Mikuli	0.13 (0.84)	2.40 (13.53)	Ashkenazy
Mazurka 30-2 Klindworth	0.13 (1.27)	1.53 (11.05)	Mohovich
Mazurka 30-2 Klindworth	0.21 (2.51)	1.87 (14.31)	Fou
Mazurka 30-2 Klindworth	0.11 (0.89)	1.65 (14.19)	Ashkenazy
Mazurka 30-2 Scholtz	0.18 (1.50)	2.39 (18.41)	Mohovich
Mazurka 30-2 Scholtz	0.31 (2.17)	3.46 (19.69)	Fou
Mazurka 30-2 Scholtz	0.12 (1.14)	1.93 (17.04)	Ashkenazy
Mazurka 63-3 Mikuli	0.16 (1.97)	1.37 (10.85)	Ashkenazy
Mazurka 63-3 Joseffy	0.29 (3.41)	2.17 (23.65)	Ashkenazy
Mazurka 63-3 Kullak	0.29 (2.29)	1.99 (14.77)	Ashkenazy
Mazurka 67-1 Joseffy	0.17 (1.65)	2.22 (17.89)	Chiu
Mazurka 67-1 Klindworth	0.21 (1.70)	2.65 (18.51)	Chiu
Mazurka 68-3 Joseffy	0.36 (1.84)	4.49 (30.09)	Chiu

Table 1. Alignment errors for a number of Chopin Mazurkas by different performers and various editions of printed scores.

Results of the evaluation show that the method is able to align real-world printed scores to expressive audio performances. We have evaluated the method at the beat level to explore the possibility of more precise alignment. It can be seen from the table that the average time accuracy is quite good. We have implemented a test application that displays the score position as the music is playing based on the alignment. The tracking can be comfortably followed by eye and the application allows the viewer to see the errors as the performance unfolds. The average error figures on the score side are also good. However, the maximum errors appear to be somewhat high. The reason for this seems to be the fact that when the last beat in a system is carried over to the next system (or a beat is aligned early from the next system) the calculated error includes the distance of the margins in between the two adjacent systems. Therefore, we do not think that these figures are as drastic as they look but appear as a delayed response while following.

Approximate matching offers many advantages to the problem at hand. With the relatively simple mid-level feature and the complexity of the recognition problem with the given less-than-ideal historical scores, recognition errors are frequent. However, the method allows for graceful recovery due to two reasons. One is the tolerant distance measure which inherently absorbs pitch errors. The other is the step condition of the DTW algorithm that prevents one sequence from stalling for extended periods. This allows for catch-up after a sequence of misdetections, rests or page segmentation errors. While selecting the templates and their detection thresholds a balance was struck between false positives and false negatives.

6. CONCLUSIONS

We have presented a mid-level score representation for aligning printed music to audio. The mid-level representation allows us to bypass sophisticated OMR techniques used for recognition and semantic analysis. The method allows for alignment through use of approximate pattern matching between the compatible features obtained from audio and score representations and therefore performs alignment within a framework in which symbolic recognition accuracy is not the primary concern. At this stage of the ongoing project, the model has been evaluated on piano music and a number of scanned scores at the beat level and the results are encouraging.

Future work will concentrate on adding sectioning and support for repeats, timbre learning from audio mixtures for more accurate note modeling, adding duration and dynamics into the note model, and catering to clef changes in the sheet. An extension of the proposed method to scores that employ systems other than the grand staff is of interest and would enable the method's application to symphonic as well as ensemble music.

7. REFERENCES

[1] Cardoso, J. S., Capela, A., Rebelo, A., Guedes, C., and Costa, J. P., "Staff Detection with Stable Paths," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 31(6), 1134–1139, 2009.

[2] Choudhury, G. S., T. DiLauro, M. Droettboom, I. Fujinaga, B. Harrington, and K. MacMillan, "Optical Music Recognition System within a Large-Scale Digitization Project," *Proc. 1st International Society for Music Information Retrieval Conference* (ISMIR), 2000.

[3] Cont, A., "A Coupled Duration-Focused Architecture for Real-Time Music-to-score Alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6), 974–987, 2010.

[4] Duan, Z., and Pardo, B., "Aligning Semi-Improvised Music with its Lead Sheet," *Proc. 12th International Society for Music Information Retrieval Conference* (ISMIR), Miami, 2011.

[5] Fremerey C., Clausen, M., Ewert S., and Muller, M., "Sheet Music-Audio Identification," *Proc. 10th International Society for Music Information Retrieval Conference* (ISMIR), Kobe, 2009.

[6] Fremerey, C., Damm, D., Muller, M., Kurth, F., and Clausen, M., "Handling Scanned Sheet Music and Audio Recordings in Digital Music Libraries," *Proc. International Conference on Acoustics NAG/DAGA*, 2009.

[7] Hu, N., Dannenberg, R. B., and Tzanetakis G., "Polyphonic Audio Matching and Alignment for Music Retrieval," *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (WASPAA), New Palz, New York, 2003.

[8] Joder, C., Essid, S., and Richard, G., "An Improved Hierarchical Approach for Music-to-symbolic Score Alignment," *Proc. 11th International Society for Music Information Retrieval Conference* (ISMIR), Utrecht, 2010.

[9] Joder, C., Essid, S., and Richard, G., "A Comparative Study of Tonal Acoustic Features for a Symbolic Level Music-To-Score Alignment," *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP), Dallas, TX, US, March 2010.

[10] Joder, C., Essid, S., and Richard, G.,"A Conditional Random Field Framework for Robust and Scalable Audio-To-Score Matching," *IEEE Transactions on Audio, Speech and Language Processing,* 19(8), 2385 - 2397, November, 2011.

[11] Kurth, F., Muller, M., Fremerey, C., Chang, Y., and Clausen, M., "Automated Synchronization of Scanned Sheet Music with Audio Recordings," *Proc. 8th International Conference on Music Information Retrieval* (ISMIR), Vienna, 2007.

[12] Muller, M., Mattes, H., and Kurth, F., "An Efficient Multiscale Approach to Audio Synchronization," *Proceedings of the 7th International Conference on Music Information Retrieval* (ISMIR), Victoria, 2006.

[13] Niedermayer, B., and Widmer, G., "A Multi-pass Algorithm for Accurate Audio-to-score Alignment," *Proc. 11th International Society for Music Information Retrieval Conference* (ISMIR), Utrecht, 2010.

[14] Orio, N., and Schwarz D., "Alignment of Monophonic and Polyphonic Music to a Score," *Proc. International Computer Music Conference* (ICMC), Havana, Cuba, 2001.

[15] Otsu, N., "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 9, No. 1, 62-66, 1979.

[16] Raphael, C., "Aligning Music Audio with Symbolic Scores Using a Hybrid Graphical Model," *Machine Learning*, Vol. 65 (2-3), 389–409, 2006.

[17] Raphael, C., and Wang, J., "New Approaches to Optical Music Recognition," *Proc. 12th International Society for Music Information Retrieval Conference* (ISMIR), Miami, 2011.

[18] Rebelo, A., Fujinaga, I., Paszkiewicz, F., Marcal, A. R. S., Guedes, C., and Cardoso, J. S., "Optical Music Recognition: State-of-the-art and Open Issues," *International Journal of Multimedia Information Retrieval* (IJMIR), 2012.

REAL-TIME ONLINE SINGING VOICE SEPARATION FROM MONAURAL RECORDINGS USING ROBUST LOW-RANK MODELING

Pablo Sprechmann
University of Minnesota
sprec009@umn.edu

Alex Bronstein
Tel Aviv University
bron@eng.tau.ac.il

Guillermo Sapiro
University of Minnesota
guille@umn.edu

ABSTRACT

Separating the leading vocals from the musical accompaniment is a challenging task that appears naturally in several music processing applications. Robust principal component analysis (RPCA) has been recently employed to this problem producing very successful results. The method decomposes the signal into a low-rank component corresponding to the accompaniment with its repetitive structure, and a sparse component corresponding to the voice with its quasi-harmonic structure. In this paper we first introduce a non-negative variant of RPCA, termed as robust low-rank non-negative matrix factorization (RNMF). This new framework better suits audio applications. We then propose two efficient feed-forward architectures that approximate the RPCA and RNMF with low latency and a fraction of the complexity of the original optimization method. These approximants allow incorporating elements of unsupervised, semi- and fully-supervised learning into the RPCA and RNMF frameworks. Our basic implementation shows several orders of magnitude speedup compared to the exact solvers with no performance degradation, and allows online and faster-than-real-time processing. Evaluation on the MIR-1K dataset demonstrates state-of-the-art performance.

1. INTRODUCTION

The leading voice in musical pieces carries valuable information about the song. A system capable of separating the singing voice from the music accompaniment can be used to facilitate a number of applications such as music information retrieval, singer identification, or lyric recognition.

Separating the leading singing voice from the musical background from a monaural recording is very challenging. Existing approaches can be classified according to the level of supervision that they require. Supervised approaches tend to have a model for either the musical background, the singing voice, or both, and in general map the mixture signals onto a feature space where the separation is performed, e.g. [4, 11, 15, 19]. A common drawback of these methods is the need to identify the vocal segments beforehand, typically using features such as the Mel-Frequency Cepstrum Coefficients (MFCC). Unsupervised approaches make basic fundamental assumptions requiring no prior training or particular features. For example, in [13] the authors tackle the separation by extracting the repeating background (music) from the non-repeating foreground (voice). Most relevant for our work is the method proposed in [9]. The authors model the repetitive structure of the accompaniment with a low-rank linear model, while the singing voice is regarded as sparse and non-repetitive. The separation is performed using *robust* PCA (RPCA) [3], producing state-of-the-art results. Common drawbacks of unsupervised approaches include the requirement to observe the whole audio track to perform the separation and the fact that, unlike supervised models, the obtained sources might not follow known characteristics of the signals.

In this paper, we consider the promising results presented in [9] as a starting point. We first develop an extension of RPCA in which the low rank model is represented as a *non-negative* linear combination of *non-negative* basis vectors. This is done following recent results connecting non-convex optimization with nuclear norm optimization [17, 18] (further references are given in Section 2). As with standard non-negative matrix factorization (NMF) methods, this new model is more appropriate to represent audio signals, being applied to the magnitude of the spectrum. The use of robust NMF (RNMF) is not restricted to this application and the usage in combination with divergences in lieu of Euclidean distances is straightforward. The proposed framework can also be seen as an extension of the robustification of NMF introduced in [22]; not only does our model consider a sparse variable accounting for outliers (singing voice), but it also adds a regularization term that minimizes the rank of the linear model.

In Section 3 we show that the RPCA and RNMF frameworks induce an architecture of multi-layer feed-forward networks designed to approximate the output of the exact optimization algorithms at a fraction of their computational cost and with no decrease in performance in our various experiments. Moreover, this new framework allows to incorporate unsupervised, semi- and fully-supervised learning into RPCA and RNMF. In this way, we aim at taking the advantages of the unsupervised methods while minimizing their drawbacks via realistic learning. When combined with learning as here proposed, the obtained networks produce over 1 dB improvement in the signal-to-distortion ratio when compared to the optimization-based RPCA (extensive experimental results are presented in Section 4), and, after the offline learning, are computable online and faster than real time without the need to observe the whole audio file.

These proposed networks are closely related to the ones introduced in [6], used to produce meaningful audio features for music style and gender classification [7]. These approaches are examples of recent successful efforts in the machine learning community to produce fast trainable (auto-)encoders of sparse features of visual and audio signals (see [5, 16] and references therein). While the work in this paper comes from these ideas, it presents a fundamental difference in the sense that the proposed networks do not compute features, but perform the full separation of the singing voice from the musical accompaniment.

2. LOW-RANK SPARSE MODELS

2.1 Robust PCA

Principal component analysis (PCA) is the most widely used statistical technique for dimensionality reduction. Its performance is, however, highly sensitive to the presence of samples not following the assumed model (subspace); even a single outlier in the data matrix can render the estimation of the low rank component arbitrarily far from the true model. In [3, 21], a very elegant remedy was developed for this shortcoming, in which the low rank matrix is determined as the minimizer of a convex program. The basic idea is to decompose the data matrix $\mathbf{X}$ as $\mathbf{X} = \mathbf{L} + \mathbf{O} \in \mathbb{R}^{m \times n}$, where $\mathbf{L}$ is a low rank matrix and $\mathbf{O}$ an error matrix with a sparse number of non-zero coefficients with arbitrarily large magnitude. RPCA can be solved by minimizing the convex program

$$\min_{\mathbf{L},\mathbf{O}} \|\mathbf{L}\|_* + \lambda \|\mathbf{O}\|_1 \quad \text{s.t.} \quad \mathbf{X} = \mathbf{L} + \mathbf{O}, \qquad (1)$$

where $\|\cdot\|_*$ denotes the matrix nuclear norm, defined as the sum of the singular values (the convex surrogate of the rank), and λ is a positive scalar parameter controlling the sparsity of the outliers. Several efficient optimization algorithms have been proposed for solving (1) as, for example, the augmented Lagrangian approach presented in [12].

When the observations are noisy, the equality constraint in (1) no longer holds. The RPCA model can be reformulated as

$$\min_{\mathbf{L},\mathbf{O}} \|\mathbf{L}\|_* + \lambda \|\mathbf{O}\|_1 \ \text{ s.t. } \|\mathbf{X} - \mathbf{L} - \mathbf{O}\|_F^2 \leq \epsilon, \quad (2)$$

with $\|\cdot\|_{\mathrm{F}}$ denoting the Frobenius norm, and ϵ a parameter controlling the approximation error [21].

2.2 Robust PCA via non-convex factorization

In this paper, we tackle the RPCA problem by solving the unconstrained optimization problem

$$\min_{\mathbf{L},\mathbf{O}} \frac{1}{2} \|\mathbf{X} - \mathbf{L} - \mathbf{O}\|_F^2 + \lambda_* \|\mathbf{L}\|_* + \lambda \|\mathbf{O}\|_1 . \quad (3)$$

This formulation is equivalent to (2) in the sense that for every $\epsilon > 0$ one can find a $\lambda_* > 0$ such that both problems admit the same solution. The unconstrained formulation can be efficiently optimized via proximal methods as in [3].

In [17] it was shown that the nuclear norm of a matrix can be reformulated as a penalty over all possible factorizations,

$$\|\mathbf{L}\|_* = \min_{\mathbf{U},\mathbf{S}} \frac{1}{2} \|\mathbf{U}\|_F^2 + \frac{1}{2} \|\mathbf{S}\|_F^2 \ \text{ s.t.} \, \mathbf{US} = \mathbf{L}, \quad (4)$$

with the minimum achieved via Singular Value Decomposition (SVD) [14]. In (3), neither the rank of $\mathbf{L}$ nor the level of sparsity in $\mathbf{O}$ are assumed known *a priori*. However, in common applications, it is reasonable to have a rough upper bound, rank$(\mathbf{L}) \leq q$. Combining this with (4), we reformulate (3) as the minimization

$$\begin{aligned}\min_{\mathbf{U},\mathbf{S},\mathbf{O}} \quad & \tfrac{1}{2} \|\mathbf{X} - \mathbf{US} - \mathbf{O}\|_F^2 + \\ & \tfrac{\lambda_*}{2} (\|\mathbf{U}\|_F^2 + \|\mathbf{S}\|_F^2) + \lambda \|\mathbf{O}\|_1 \qquad (5)\end{aligned}$$

over $\mathbf{U} \in \mathbb{R}^{m \times q}$, $\mathbf{S} \in \mathbb{R}^{q \times n}$, and $\mathbf{O} \in \mathbb{R}^{m \times n}$. This decomposition reveals interesting structure hidden in the problem. The low rank component can now be thought of as an under-complete dictionary $\mathbf{U}$, with q atoms, multiplied by a matrix $\mathbf{S}$ containing the corresponding coefficients for each data vector in $\mathbf{X}$. This interpretation brings the RPCA problem close to that of matrix factorization and sparse coding.

This new factorized formulation drastically reduces the number of optimization variables from $2nm$ to $nm + q(n+m)$. While problem (5) is no longer convex, it can be shown that any of its stationary points satisfying $||\mathbf{X} - \mathbf{US} - \mathbf{O}||_2^2 \leq \lambda_*$, is an optimal solution of (5) [14]. Thus, the problem can be solved using alternating minimization or block coordinate schemes, without the risk of remaining stuck in a local minimum. This redounds in a significant speed-up in the optimization [18].

2.3 Robust NMF

In many applications, such as spectrogram decompositions, it desirable to find non-negative factorizations. This is in the heart of the non-negative matrix factorization paradigm . We now extend (5) to consider the low rank and the outlier terms to be non-negative,

$$\min_{\mathbf{U}\geq\mathbf{0},\mathbf{S}\geq\mathbf{0},\mathbf{O}\geq\mathbf{0}} \quad \tfrac{1}{2}\left\|\mathbf{X} - \mathbf{US} - \mathbf{O}\right\|_F^2 + \tfrac{\lambda_*}{2}(\|\mathbf{U}\|_F^2 + \|\mathbf{S}\|_F^2) + \lambda\left\|\mathbf{O}\right\|_1 . \quad (6)$$

This new formulation is no longer equivalent to (3). In fact, applying (4) directly to the matrix $\mathbf{US}$, we obtain $\hat{\mathbf{U}}\hat{\mathbf{S}}$ with the factors $\hat{\mathbf{S}}$ and $\hat{\mathbf{U}}$ not being necessarily non-negative. Adding the non-negativity constraint produces the inequality

$$||\mathbf{US}||_* \leq \frac{1}{2}\min_{\hat{\mathbf{S}}\geq\mathbf{0},\hat{\mathbf{U}}\geq\mathbf{0}} ||\hat{\mathbf{U}}||_{\mathrm{F}}^2 + ||\hat{\mathbf{S}}||_{\mathrm{F}}^2. \qquad (7)$$

Thus, the sum of the Frobenius norms of the non-negative matrices $\mathbf{S}$ and $\mathbf{U}$ regularizes an upper bound of the nuclear norm of their product.

Standard NMF is obtained as a particular case by setting to zero both λ_* and λ, while the robust version of NMF introduced in [22] is obtained when only λ_* is selected as zero. In this paper we use RNMF as stated in (6), however its extension to more general fitting terms such as β-divergences is straightforward. Problem (6) can be optimized using multiplicative algorithms, commonly used in the NMF context.

2.4 Robust non-negative projections

Let us now assume to be given a low dimensional model, $\mathbf{U} \in \mathbb{R}^{m\times q}$, learned from some data $\mathbf{X} \approx \mathbf{US} + \mathbf{O} \in \mathbb{R}^{m\times n}$. A new input vector $\mathbf{x}$ drawn from the same distribution as $\mathbf{X}$ can be decomposed into $\mathbf{x} = \mathbf{Us} + \mathbf{n} + \mathbf{o}$, where $\mathbf{Us}$ represents the low dimensional component, $\mathbf{n}$ is a small perturbation, and $\mathbf{o}$ is a sparse outlier vector. It can be obtained via

$$\min_{\mathbf{s}\geq\mathbf{0},\mathbf{o}\geq\mathbf{0}} \quad \tfrac{1}{2}\left\|\mathbf{x} - \mathbf{Us} - \mathbf{o}\right\|_2^2 + \tfrac{\lambda_*}{2}\left\|\mathbf{s}\right\|_2^2 + \lambda\left\|\mathbf{o}\right\|_1 ,(8)$$

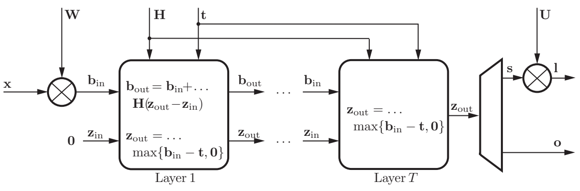

W
H
t
U
x
$\mathbf{b}_{\text{in}}$
$\mathbf{b}_{\text{out}} = \mathbf{b}_{\text{in}} + \ldots$ $\mathbf{H}(\mathbf{z}_{\text{out}} - \mathbf{z}_{\text{in}})$
$\mathbf{b}_{\text{out}}$
0
$\mathbf{z}_{\text{in}}$
$\mathbf{z}_{\text{out}} = \ldots$ $\max\{\mathbf{b}_{\text{in}} - \mathbf{t}, \mathbf{0}\}$
$\mathbf{z}_{\text{out}}$
Layer 1
Layer T
s
l
o

Figure 1. RNMF encoder architecture with T layers.

a convex problem similar to the one of standard sparse coding. The solution can be obtained via proximal methods [1], which split the objective function (8) into a smooth part (the first two terms), and a non-differentiable part (the ℓ_1 norm of the outliers vector). Proximal methods iterate between a gradient descent on the smooth function and an application of the proximal operator (which assumes a closed form of one-sided soft-thresholding), as detailed in Algorithm 1. This algorithm is conceptually very similar to the popular iterative shrinkage and thresholding algorithm (ISTA) [2]. We do not use this algorithm as an explicit tool, but rather as a motivation of the architecture of a feed-forward network capable of accurately performing the separation in real time, as discussed next.

```
input : Data $\mathbf{x}$, dictionary $\mathbf{U}$.
output: Nonnegative coefficient vector $\mathbf{s}$ and
        nonnegative outlier vector $\mathbf{o}$.
Define
$\mathbf{H} = \mathbf{I} - \frac{1}{\alpha}\begin{pmatrix} \mathbf{U}^{\mathrm{T}}\mathbf{U} + \lambda_*\mathbf{I} & \mathbf{U}^{\mathrm{T}} \\ \mathbf{U} & (1+\lambda_*)\mathbf{I} \end{pmatrix}$,
$\mathbf{W} = \frac{1}{\alpha}\begin{pmatrix} \mathbf{U}^{\mathrm{T}} \\ \mathbf{I} \end{pmatrix}$, and $\mathbf{t} = \frac{\lambda}{\alpha}\begin{pmatrix} \mathbf{0} \\ \mathbf{1} \end{pmatrix}$.
Initialize $\mathbf{z} = \mathbf{0}$, $\mathbf{b} = \mathbf{Wx}$.
repeat
    $\mathbf{y} = \max\{\mathbf{b} - \mathbf{t}, 0\}$
    $\mathbf{b} = \mathbf{b} + \mathbf{H}(\mathbf{y} - \mathbf{z})$
    $\mathbf{z} = \mathbf{y}$
until until convergence ;
Output $(\mathbf{o}, \mathbf{s}) = \mathbf{z}$.
```

Algorithm 1: RNMF given the dictionary $\mathbf{U}$.

3. FAST ROBUST SPARSE MODELING

To avoid the computational complexity inherent to exact sparse coding algorithms, it has been recently proposed to learn non-linear regressors capable of producing good approximations in a fixed amount of time [6,10]. We follow these ideas to obtain encoders capable of efficiently approximating the solution of RPCA and RNMF. [1] We first discuss the general framework and then describe specific uses.

[1] Due to space constraints, we show details only for RNMF; RPCA can be obtained by removing the non-negativity constraints and modifying the proximal operator.

We aim at constructing a parametric regressor $\mathbf{z} = (\mathbf{o}, \mathbf{s}) = \mathbf{h}(\mathbf{x}, \mathbf{\Theta})$, with some set of parameters, collectively denoted as $\mathbf{\Theta}$, capable of accurately performing the singing voice separation for a given training sample $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Here, each $\mathbf{x}_i$ represents the magnitude spectrum of a mixture of voice and music; training samples may come from many different singers and songs.

As in [6], we design an architecture for the encoders based on an exact optimization algorithm, in this case Algorithm 1. We propose a multi-layer artificial neural networks where each layer implements a single iteration of the algorithm, as depicted in Figure 1. The parameters of the network are the matrices $\mathbf{W}$ and $\mathbf{H}$ and the thresholds $\mathbf{t}$. [2] These encoder architectures are continuous and almost everywhere $\mathcal{C}^1$ with respect to the parameters, allowing the use of (sub)gradient descent methods for training.

We train the encoders by minimizing over $\mathcal{X}$ functions of the form

$$\mathcal{L}(\mathbf{\Theta}) = \frac{1}{|\mathcal{X}|} \sum_{x_i \in \mathcal{X}} L(\mathbf{\Theta}, \mathbf{x}_i), \tag{9}$$

where $L(\mathbf{\Theta}, \mathbf{x}_i)$ is a function that measures the quality of the code $\mathbf{z}_i = \mathbf{h}(\mathbf{x}_i, \mathbf{\Theta})$. Specifically, we iteratively select a random subset of $\mathcal{X}$ and then update the network parameters as $\mathbf{\Theta} \leftarrow \mathbf{\Theta} - \mu \frac{\partial \mathcal{L}(\mathbf{\Theta})}{\partial \mathbf{\Theta}}$, where μ is a decaying step, repeating the process until convergence. The decoder is just a linear operator given by a dictionary $\mathbf{U}$, see Figure 1.

Once trained, the parameters $\mathbf{\Theta}$ and the dictionary $\mathbf{U}$ are fixed, and the network is used to sequentially process new data. The latency of both the RPCA and RNMF networks (referred henceforth as *NN RPCA* and *NN RNMF*, respectively) is of the order of a single STFT frame (hundreds of milliseconds), while the exact algorithms require the entire signal to be observed.

3.1 Training regimes

Training of the proposed RPCA and RNMF encoders is possible under different regimes. We refer as *supervised* to the setting where the training set consists of the mixed signal $\mathbf{x}_i = \mathbf{o}_i^* + \mathbf{l}_i^*$, and the synchronized ground truth voice and accompaniment signals $\mathbf{o}_i^*$ and $\mathbf{l}_i^*$ (each vector corresponding to the magnitude spectrogram). In that case, we set $L(\mathbf{\Theta}, \mathbf{x}_i) = ||\mathbf{U}\mathbf{s}_i - \mathbf{l}_i^*||_2^2 + ||\mathbf{o}_i - \mathbf{o}_i^*||_2^2$, with $(\mathbf{o}_i, \mathbf{s}_i) = \mathbf{h}(\mathbf{x}_i, \mathbf{\Theta})$. For *NN RPCA*, the dictionary $\mathbf{U}$ is established using SVD applied to the clean accompaniment samples, $\mathbf{l}_i^*$, while for *NN RNMF*, the non-negative dictionary $\mathbf{U}$ is constructed running the multiplicative RNMF algorithm on the training data.

[2] In the network, extra flexibility is obtained by learning different thresholds t_i for each component.

Table 1. Performance on the recovered vocal track on MIR-1K.

Method	GNSDR	GSNR	GSAR	GSIR
Ideal freq. mask	13.48	5.46	13.65	31.22
ADMoM RPCA [9]	5.00	2.38	6.68	13.76
Proximal RPCA	5.48	3.29	7.02	13.91
NN RPCA Untrained	5.30	2.66	6.80	13.00
NN RPCA Unsupervised	5.62	2.87	6.90	14.02
NN RPCA Supervised	6.38	3.18	7.22	16.47
NN RPCA Dict. update	6.42	3.19	7.23	16.57
Multiplicative RNMF	5.60	3.39	6.94	14.67
NN RNMF Untrained	1.62	0.00	5.85	5.13
NN RNMF Unsupervised	5.00	2.66	6.63	11.89
NN NMF Supervised	6.36	3.37	7.10	16.96
NN RNMF Dict. update	**6.55**	**3.55**	**7.24**	**17.65**

We refer as *semi-supervised* to the setting in which isolated samples of voice and background are available, but are not synchronized (the $\mathbf{x}_i$ are now either the voice or the accompaniment). The training of the network is performed in the same way as the supervised case, but setting to zero the missing source.

Finally, in the *unsupervised* setting we only have access to mixtures as training data and the objective $L(\mathbf{\Theta}, \mathbf{x}_i) = \frac{1}{2}\|\mathbf{x}_i - \mathbf{U}\mathbf{s}_i - \mathbf{o}_i\|_2^2 + \frac{\lambda_*}{2}\|\mathbf{s}_i\|_2^2 + \lambda\|\mathbf{o}_i\|_1$ is used to directly minimize the cost in (6).

Dictionary adaptation. The performance of both the RPCA and RNMF networks can be further improved if the dictionary $\mathbf{U}$ (decoder) is updated during the training. In the unsupervised setting, for *NN RPCA*, $\mathbf{U}$ is updated via gradient descent as before, while in *NN RNMF* via the standard multiplicative update,

$$\mathbf{U} \leftarrow \mathbf{U} \odot \frac{\mathbf{Y}\mathbf{S}^{\mathrm{T}}}{\mathbf{U}(\mathbf{S}\mathbf{S}^{\mathrm{T}} + \lambda_* \mathbf{I})}, \tag{10}$$

where $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the input matrix, $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_n)$ is the matrix of the corresponding codes, $\mathbf{Y} = (\mathbf{x}_1 - \mathbf{o}_1, \ldots, \mathbf{x}_n - \mathbf{o}_n)$, and $\odot$ and the fraction denote, respectively, element-wise multiplication and division. This update minimizes the objective in (6) for fixed $\mathbf{O}$ and $\mathbf{S}$, and is guaranteed to preserve the non-negativity of $\mathbf{U}$. Analogously, in the semi- and fully-supervised scenarios, $\mathbf{U}$ can be updated by minimizing the corresponding $\mathcal{L}(\mathbf{\Theta})$ using the ground-truth music accompaniment. Again using gradient descent and multiplicative updates for RPCA and RNMF respectively.

4. EXPERIMENTAL RESULTS

Dataset. We evaluate the separation performance of the proposed methods on the MIR-1K dataset [8], containing 1000 16 kHz clips extracted from 110 Chinese karaoke songs performed by 19 amateur singers (11 males and 8 females). Each clip duration ranges from 4 to 13 seconds, totaling about 133 minutes. We reserved about 23 minutes of audio sang by one male

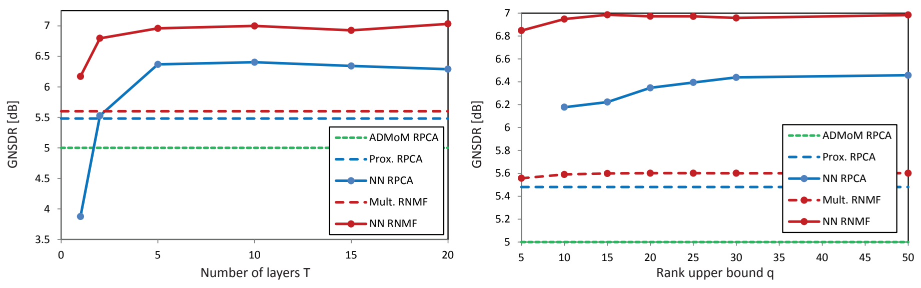


Figure 2. Performance of the supervised *NN RPCA* and *NN RNMF* on the MIR-1K dataset for different number of layers T (left, q fixed to 20), and values of the rank bound q (right, T fixed to 10). GNSDR of the recovered vocal track is used as the comparison criterion. For reference, the performance of exact RPCA and RNMF is given.

and one female singers (*abjones* and *amy*) for the purpose of training; the remaining 110 minutes of 17 singers were used for testing. The voice and the music tracks were mixed linearly with equal energy.

Evaluation. As the evaluation criteria, we used the BSS-EVAL metrics [20], which calculate the *source-to-distortion ratio* (SDR), [3] the *source-to-artifacts ratio* (SAR), and the *source-to-interference ratio* (SIR). As in [9], we computed the global normalized SDR,

$$\mathrm{GNSDR} = \sum_{i=1}^{N} \delta_i(\mathrm{SDR}(\hat{s}, s) - \mathrm{SDR}(x, s)),$$

where $\hat{s}$ and s are the corresponding original and estimated voice signal, x is the mixture, δ_i is the relative duration of each of the N testing pieces. Prefix "G" indicates average sample performance,e.g. GSAR. We also computed the *signal-to-noise ratio* (SNR).

Comparison of separation methods. We evaluated the proposed *NN RPCA* and *NN RNMF* using the different training settings discussed in Section 3.1. In all our examples (except when explicitly mentioned), we used $T = 10$ layers and $q = 20$. We compare these result against three exact solvers: *ADMoM RPCA* solving (1) with $\lambda = 1/\sqrt{n}$ (as suggested in [9]) via the alternating direction method of multipliers [12], for which the code from [9] was used; *Proximal RPCA* solving (3) using the proximal method from [3], with $\lambda = \sqrt{2n}\sigma$ and $\lambda_* = \sqrt{2}\sigma$ with $\sigma = 0.3$ set following [3]; and *Multiplicative RNMF* solving (6) using the standard multiplicative algorithm.

In all experiments, the spectrogram of each mixture was computed using a window size of 1024 and a step size of 256 samples (at 16 KHz sampling rate). Training was performed using 1000 safe-guarded gradient descent iterations on a random subset of 10.000 spectral frames for training and the same amount of distinct frames for cross-validation.

[3] In this work the SDR is computed using the latest release of the BSS-EVAL code. The reported values are higher (equally for all algorithms) than the ones reported in [9], since they used the older release of that package.

Table 1 summarizes the performance of the compared methods. The best performance is achieved by the *NN RNMF* with trained dictionary. The use of the proximal RPCA algorithm allowing for inexact reconstruction of the data (thus accounting for unstructured noise) gives almost 0.5 dB improvement over [9]. The use of unsupervised training was more successful in the *NN RPCA*; however, both *NN RPCA* and *NN RNMF* outperform ADMoM RPCA.

The complexity of the proposed systems is significantly lower to the one of exact algorithms: our unoptimized Matlab code that uses GPU acceleration is capable of computing the networks about 70 faster than real time, while a preliminary implementation on iPhone 4S is online and $6 - 7$ times faster than real time (after offline training).

Parameter selection. We also evaluated the performance of the supervised RPCA and RNMF networks as a function of the two principal parameters: the number of layers T and the rank bound q, see Figure 2.

Supervised learning has a dramatic effect on the performance of the networks. With just two layers, the RPCA network already outperforms the exact RPCA algorithms; as a reference, an untrained network, with the parameters $\mathbf{W}, \mathbf{H}$, and $\mathbf{t}$ set according to Algorithm 1, requires over 15 layers to approach this performance. This phenomenon is even more pronounced in the case of RNMF. The influence of the number of layers quickly saturates; slight oscillations in the GNDSR are due to the randomization used at training.

In contrast, the effect of q is less dramatic. The networks outperform the exact algorithms already for $q = 5$ and the performance saturates for $q \geq 30$. This is radically different from the behavior of standard NMF approaches, in which setting the number of columns in the non-negative factor $\mathbf{U}$ significantly affects the performance. In fact, RNMF with $\lambda_* = 0$ as [22] yields 5.60 dB GNSDR for $q = 1$, which drops to 2.88 dB for $q = 3$ and to -2.5 dB for $q = 10$.

Supervised training settings. We evaluated the influence of the different training regimes on the perfor-

5

Table 2. Performance of NN RNMF on the vocal trak of *Sunrise* song. Audio files are available for download here

Method	NSDR	SNR	SAR	SIR
Ideal freq. mask	14.98	5.84	18.46	39.40
ADMoM RPCA [9]	1.61	2.99	11.13	6.60
Supervised (*MIR-1K*)	7.16	4.86	14.21	13.25
Supervised (*We are in love*)	7.85	5.47	15.35	13.59
Supervised (*Sunrise*)	10.93	5.67	16.16	19.20
Semi-supervised (*We are in love*)	7.35	4.69	11.39	20.01
Semi-supervised (*Sunrise*)	8.46	5.11	12.20	23.97

mance of the networks on Shannon Hurley's song *Sunrise*, available from `archive.org`. The song was resampled at 16 kHz and voice was artificially mixed with the guitar accompaniment with equal energies. Three distinct datasets were used for training the nets: two singers from MIR-1K used in the previous experiments; another Shannon Hurley's song *We are in love*; and the same *Sunrise*, song on which the testing was performed (given only for comparison). Supervised and semi-supervised regimes were used.

Table 2 summarizes the obtained results. RNMF networks trained using mixtures from MIR-1K outperform [9] by nearly 5.5 dB GNSDR; training on more singer-specific data (*We are in love* song) improves this result by about 0.7 dB. ; finally, training on a mixture from the same song yields over 3.5 dB improvement. We conclude that training the networks on unrelated singers and accompaniments already achieves very high performance. Semi-supervised training on the *We are in love* song yields a minor improvement over MIR-1K, and cedes 0.5 dB to the fully-supervised training. We conclude that in the absence of synchronized voice and music tracks for supervised training, semi-supervised training still produces comparable results.

5. CONCLUSION

Marrying ideas from convex optimization with multilayer neural networks, we have developed efficient architectures for real-time online single-channel separation of singing voice from musical accompaniment. Our approach achieves state-of-the-art results on the MIR-1K datasets with orders of magnitude improvement in runtime and latency. In future work, we are going to extend this framework to denoising and simultaneous separation and speaker identification.

6. REFERENCES

[1] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Convex optimization with sparsity-inducing norms. In *Optimization for Machine Learning*. MIT Press, 2011.

[2] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2:183–202, March 2009.

[3] E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3), May 2011.

[4] J.-L. Durrieu, B. David, and G. Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *J. Sel. Topics Signal Processing*, 5(6):1180–1191, 2011.

[5] I. Goodfellow, Q. Le, A. Saxe, H. Lee, and A. Y. Ng. Measuring invariances in deep networks. In *NIPS*, pages 646–654. 2009.

[6] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *ICML*, pages 399–406, 2010.

[7] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun. Unsupervised learning of sparse features for scalable audio classification. In *ISMIR*, 2011.

[8] C.L. Hsu and J.S.R. Jang. On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 18(2):310–319, 2010.

[9] P.-S. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *ICASSP*, 2012.

[10] K. Kavukcuoglu, M.A. Ranzato, and Y. LeCun. Fast inference in sparse coding algorithms with applications to object recognition. *arXiv:1010.3467*, 2010.

[11] Y. Li and D. L. Wang. Separation of singing voice from music accompaniment for monaural recordings. *IEEE Trans. on Audio, Speech & Lang. Proc.*, 15(4):1475–1487, 2007.

[12] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *Arxiv preprint arXiv:1009.5055*, 2010.

[13] A. Liutkus, Z. Rai, R. Badeau, B. Pardo, and G. Richard. Adaptive filtering for music/voice separation exploiting the repeating musical structure. In *ICASSP*, 2012.

[14] G. Mateos and G. B. Giannakis. Robust PCA as bilinear decomposition with outlier-sparsity regularization. *arXiv.org:1111.1788*, 2011.

[15] A. Ozerov, P. Philippe, F. Bimbot, and R. Gribonval. Adaptation of bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *IEEE Trans. on Audio, Speech & Lang. Proc.*, 15(5):1564–1578, 2007.

[16] M. A. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*, 2007.

[17] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.*, 52(3):471–501, 2010.

[18] B. Recht and C. Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. 2011.

[19] S. Vembu and S. Baumann. Separation of vocals from polyphonic audio recordings. In *ISMIR*, pages 337–344, 2005.

[20] E. Vincent, R. Gribonval, and C. Févotte. Performance measurement in blind audio source separation. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 14(4):1462–1469, 2006.

[21] H. Xu, C. Caramanis, and S. Sanghavi. Robust PCA via outlier pursuit. In *NIPS*, pages 2496–2504. 2010.

[22] L. Zhang, Z. Chen, M. Zheng, and X. He. Robust non-negative matrix factorization. *Frontiers of Electrical and Electronic Engineering in China*, 6:192–200, 2011.

FOLKSONOMY-BASED TAG RECOMMENDATION FOR ONLINE AUDIO CLIP SHARING

Frederic Font[1], Joan Serrà[2] and Xavier Serra[1]
[1]Music Technology Gorup, Universitat Pompeu Fabra, Barcelona, Spain
[2]Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Barcelona, Spain
frederic.font@upf.edu, jserra@iiia.csic.es, xavier.serra@upf.edu

ABSTRACT

Collaborative tagging has emerged as an efficient way to semantically describe online resources shared by a community of users. However, tag descriptions present some drawbacks such as tag scarcity or concept inconsistencies. In these situations, tag recommendation strategies can help users in adding meaningful tags to the resources being described. Freesound is an online audio clip sharing site that uses collaborative tagging to describe a collection of more than 140,000 sound samples. In this paper we propose four algorithm variants for tag recommendation based on tag co-occurrence in the Freesound folksonomy. On the basis of removing a number of tags that have to be later predicted by the algorithms, we find that using ranks instead of raw tag similarities produces statistically significant improvements. Moreover, we show how specific strategies for selecting the appropriate number of tags to be recommended can significantly improve algorithms' performance. These two aspects provide insight into some of the most basic components of tag recommendation systems, and we plan to exploit them in future real-world deployments.

1. INTRODUCTION

Online platforms where people share user generated content have stressed the need for efficient methods to describe and retrieve such content. Freesound [1] is an online audio clip sharing site which clearly reflects this need. It contains more than two million users and 140,000 user-contributed sound samples covering a wide variety of sounds (from field recordings and sound effects to drum loops and instrument samples), which have to be well described to allow proper retrieval.

In recent years, collaborative tagging has emerged as an efficient way to describe online resources shared by a community of users. In collaborative tagging systems, users describe information items by annotating them with a number of "free-form" semantically-meaningful textual labels, called tags, that act as keywords and that can be later used for retrieval purposes. A collection of tags together with their associations to content resources is commonly known as a *folksonomy*.

In the majority of collaborative tagging systems, including Freesound, users are not constrained by any particular number of tags to assign, nor by the use of any specific vocabulary where to pick the tags from. Therefore, descriptions can be done at many different levels of detail and accuracy. Description inconsistencies can then arise due to the ambiguity of tag meanings, tag scarcity, the use of personal naming conventions, typographical errors, or even the use of different languages [2].

One strategy for trying to overcome some of these problems, and thus obtain more comprehensive and consistent descriptions, has been the use of tag recommendation systems to help users in the tagging process. These systems analyze the first (usually few) tags that users introduce when describing a particular item, and quickly suggest new tags that can also be meaningful or relevant for the item being described. The same algorithms for tag recommendation can be used, in an off-line mode, to extend the descriptions of information items by analyzing their tags and automatically adding new ones (what is normally called tag propagation).

In this paper we present and evaluate four variants of an algorithm for tag recommendation in Freesound. Our recommendation is based on tag semantic similarity derived from tag co-occurrence in the Freesound folksonomy. A novel aspect of the algorithm is a step focused on automatically selecting the number of tags to recommend given a sorted list of candidate tags. Tag propagation methods found in related work (see below) do not perform this step, and usually evaluate algorithms at different values of N recommended tags. We compare our algorithm with simpler versions which either always recommend a fixed number of tags, or only recommend tags that are repeated in the list of candidates.

The rest of the paper is organized as follows. In Sec. 2 we review the related work and in Sec. 3 we briefly describe the Freesound folksonomy. Sec. 4 explains the proposed algorithm for tag recommendation. Secs. 5 and 6 describe the evaluation methodology and present the obtained results. In Sec. 7 we conclude the paper with a discussion about our findings and future work.

2. RELATED WORK

Collaborative tagging has been widely researched in the last few years. Some studies focus on a general description of the dynamics of collaborative tagging and user behavior when tagging [2–5]. Other studies have looked at the motivations that users have at the moment of tagging, proposing then automatic tag classification methods to organize types of tags according to these motivations [6, 7]. Most of the work done in the analysis of collaborative tagging systems takes as case studies well-known sites such as Delicious (bookmark sharing), CiteULike (scientific reference sharing) and Flickr (photo sharing).

A variety of methods have been proposed for tag propagation and tag recommendation, especially for the case of image annotation. In [5] and [8], content analysis of images is used to obtain similar images and then propagate or recommend tags from these images to the source. Instead of using content analysis, Sigurbjörnsson and Zwol [9] propose a method for image tag recommendation based on tag similarities derived from a folksonomy. Their approach is similar to the one we describe in this paper, though they use different strategies for sorting candidate tags and do not perform a final selection of the number of tags to recommend. In [10] and [11], more complex strategies for tag recommendation based on folksonomies are described and evaluated with data from Delicious, BibSonony and Last.fm (using hierarchical tag structures [10] and the *FolkRank* ranking algorithm [11]). Again, none of these approaches performs any selection of the number of tags to recommend.

In the field of sound and music, most of the work related with tag propagation or recommendation is not based on folksonomy analysis, but on the extraction of content-based audio features that can later be used to annotate songs with labels or tags (which is more commonly known as semantic annotation). Sordo [12] describes a method based on audio content similarity for propagating tags (related with style and mood) between acoustically similar songs. Martínez et al. [13] use a similar idea for automatically propagate tags in scarcely annotated samples of Freesound. In [14] and [15], a different approach for automatic annotation of music and sound effects is described, which is based on using machine learning techniques to learn mappings between tags and audio features. Due to the content-based nature of these strategies, they are not directly comparable to the approach we propose in this paper.

3. FREESOUND FOLKSONOMY

In Freesound, users can upload sound samples and then describe them with as many tags as they feel appropriate [1]. For building the folksonomy we use in our experiments, we considered user annotations between April 2005 and September 2011. The folksonomy includes a total of 785,466 annotations that assign 30,985 unique tags (not necessarily semantically unique, but with different string representations) to 118,620 sounds. As opposite to other well studied collaborative tagging systems such as Delicious or CiteULike, Freesound has what is called a *narrow folksonomy* [16], meaning that sound annotations are shared among all users and therefore one single tag can only be assigned once to a particular sound (e.g. the tag `field-recording` cannot be added twice to the same sound).

[1] Since a recent software upgrade, Freesound requires a minimum of three tags to annotate a sound. However, the data we analyze is prior to the introduction of this requirement.

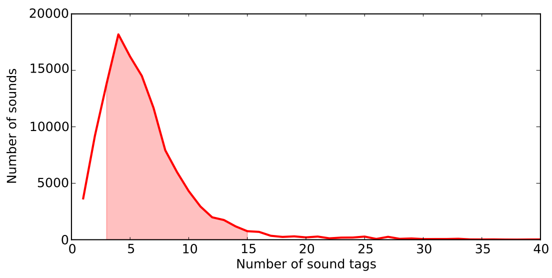


Figure 1: Distribution of sounds per number of tags. The global average of tags per sound is 6.16 and the standard deviation is 6.23.

Fig. 1 shows the distribution of the number of tags per sound in Freesound. We are particularly interested in recommending tags for the sounds that fall in the range of $[3, 15]$ tags (shadowed zone in Fig. 1), which are more than 80% of the total. The reason for focusing on these sounds is that the algorithm variants we present take as input the tags that have already been assigned to a sound. We consider 3 tags as enough input information for our algorithms to provide good recommendations. For sounds with less tags, content-based strategies such as the ones outlined in Sec. 2 are probably more suitable. On the other hand, sounds with more than 15 tags are, in general, enough well described.

Among the total number of 30,985 unique tags present in the folksonomy, we have applied a threshold to take only into consideration the tags that have been used at least 10 times, i.e. the tags that appear on at least 10 different sounds. By this we assume that tags that have been used less than 10 times are irrelevant for our purposes. In addition, by discarding less frequent tags, we reduce the computational complexity of the calculations described in Sec. 4.1. After applying this threshold, we are left with 6,232 unique tags, representing 20% of the total. Nonetheless, 93% of all annotations associate one of these 6,232 unique tags with a sound, thus we still take into account the vast majority of the original information.

4. PROPOSED ALGORITHM

The tag recommendation algorithm described in this paper consists of the three steps depicted in the diagram of Fig. 3. Variants are obtained by combining the different strategies proposed for the second and third steps. Feeding the algorithm variants with a number of $inputTags$, they output a set of $recommendedTags$. In the following subsections we describe each one of the depicted steps.

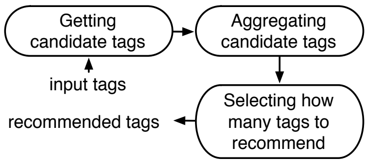

Getting candidate tags
Aggregating candidate tags
input tags
Selecting how many tags to recommend
recommended tags

Figure 3: Block diagram of the described tag recommendation algorithm.

4.1 Getting candidate tags

The first step in the recommendation processes is getting a number of candidate tags according to the set of $inputTags$. For this purpose we build a tag similarity matrix based on tag co-occurrences in sound descriptions, following the Actor-Concept-Instance model proposed by Mika [17]. This tag similarity matrix gathers information from the whole folksonomy and only needs to be computed once.

The Actor-Concept-Instance model proposes to represent a folksonomy F as a tripartite hypergraph $H(F) = \langle V, E \rangle$, where vertices are given by three finite sets of objects, $V = U \cup T \cup R$ (U, T, and R denoting users, tags, and resources, respectively), and each edge represents a tag-resource association done by a user $E = \{\{u, t, r\} | (u, t, r) \in F\}$. We unfold this tripartite hypergraph into the bipartite graph TR, which reflects only the associations between tags and resources (sounds in our case). We can represent the bipartite graph TR as a matrix $D = \{d_{ij}\}$, where $d_{ij} = 1$ if tag t_i has been used to label resource r_j (otherwise $d_{ij} = 0$). We can then define the matrix $S = DD'$, which corresponds to a one-mode network connecting tags on the basis of shared resources. Elements s_{ij} of S indicate the number of sounds in which tags t_i and t_j appear together. Therefore, the diagonal of S represents the total number of different sounds labeled with a tag $t_{i=j}$. We then normalize S by dividing each element by $\sqrt{s_{ii}}\sqrt{s_{jj}}$. In this manner, we obtain the cosine similarity measures between tags t_i and t_j (the cosine similarity between the i-th and the j-th rows of D). Preliminary experiments using other distances such as Jaccard reported worse results. Furthermore, cosine similarity has been widely used in the literature and has been shown to be effective as a semantic relatedness measure in folksonomies [18]. Fig. 2 shows a graph visualization of an excerpt of S.

Having calculated the tag similarity matrix S, we iterate over $inputTags$ and get, for each element i, a set of candidates $C_{inputTag_i}$. For that we select the N most similar tags of $inputTag_i$ (i.e. the N most similar graph neighbors). We keep these similarity values for further processing in the following steps. In all our experiments we use $N = 100$. Hence, for instance, if our algorithm is feeded with three input tags, it will get a maximum of 300 candidate tags (provided that all three input tags have at least 100 neighbors). In preliminary experiments we observed that using values of $N > 100$ did not alter our recommendation results.

4.2 Aggregating candidate tags

The next step of our algorithm is to aggregate and sort the obtained candidate tags into a single list C_{sorted}. For this purpose we propose two different strategies which are now described.

4.2.1 Similarity-based strategy

In the first strategy we construct C_{sorted} by aggregating all sets of candidate tags $C_{inputTag_i}$ into a single list C_{raw}, and then ordering them by their similarity values (taken from the previous steps). As we do not want to recommend tags that are already part of $inputTags$, we remove any occurrences of these tags in C_{raw}. If there are repeated candidate tags in C_{raw}, their similarity values are added as a way of promoting these tags that appear in $C_{inputTag_i}$ of more than one input tag i. We finally normalize the similarity values by dividing them by the number of $inputTags$ (thus maintaining values in the original range).

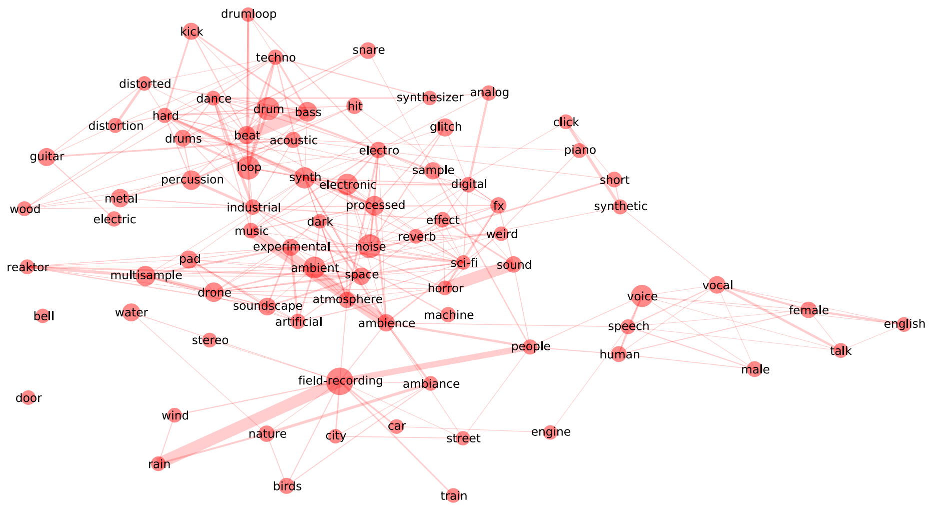


Figure 2: Graph visualization of the tag similarity matrix S. Edge widths represent the similarity between two tags. Node size is a logarithmic function of the absolute tag frequency. For the sake of clarity, only edges above a certain threshold and tags above a certain level of absolute frequency are shown.

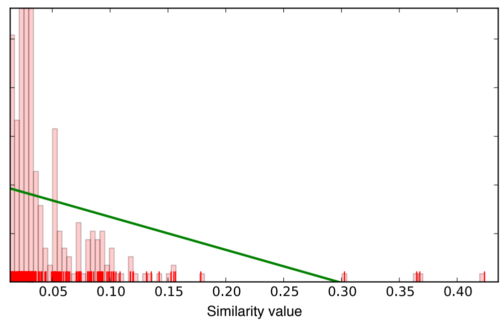


Figure 4: Example of the linear regression strategy for selecting how many tags to recommend. The straight line shows the linear regression of the histogram. Recommended tags are those placed at the right of the point where the linear regression crosses the y-axis.

4.2.2 Rank-based strategy

The second strategy is based on assigning a rank value to each candidate tag. For this purpose, we sort each set of $C_{inputTag_i}$ and then assign rank values as:

$$rank(neighbor_n) = N - (n - 1),$$

where n is the position of the neighbor in $C_{inputTag_i}$ (thus n ranges from 1 to N). This way, the closest tag to every input tag will be assigned with a rank value of N. We then perform the aggregation as we would do in the similarity-based strategy, but using the assigned rank values as similarities.

4.3 Selecting how many tags to recommend

Once we have computed C_{sorted} (either using similarity-based or rank-based strategies), we select how many of these tags should be outputted as $recommendedTags$. Our approach is based on the hypothesis that given the distribution of similarity or rank values in C_{sorted}, it will be possible to determine a threshold that separates a set of meaningful tags from the other candidates. That is to say, that "good" tags for recommendation will appear as an isolated group from the rest of the distribution. In order to automatically determine this threshold, we again propose two different strategies.

4.3.1 Linear regression strategy

The first strategy consists in calculating the least-squares linear regression of the histogram of C_{sorted}. The threshold is set to the point where the linear regression crosses the y-axis. Fig. 4 shows an example of using this strategy with a histogram of tag similarity values. In that case threshold is set at 0.29. Therefore, all candidate tags with a similarity value higher than 0.29 would be outputted as $recommendedTags$.

4.3.2 Statistical test strategy

The second strategy has two steps. First, we estimate the probability density function (PDF) of C_{sorted}. For that purpose, we use a kernel density estimator [19]. Second, we iteratively take consecutive samples of the PDF (starting from the side where the candidates with highest rank or similarity lay) and perform a statistical test for normality. For that purpose we use the Anderson-Darling test [20]. The threshold is set at the point of the PDF where the test fails for the first time (i.e.the probability of having an independent Gaussian distribution is not statistically significant). The idea behind this process is that there will be a set of good tags for the recommendation that will exhibit a normal, independent distribution separated from the rest of candidate tags. The statistical test fails when it detects departures from normality, and according to our hypothesis this happens when non-meaningful candidate tags start affecting the PDF. Fig. 5 shows an example of applying this strategy using rank values for C_{sorted}. All tags under the shaded zone in the right will be recommended.

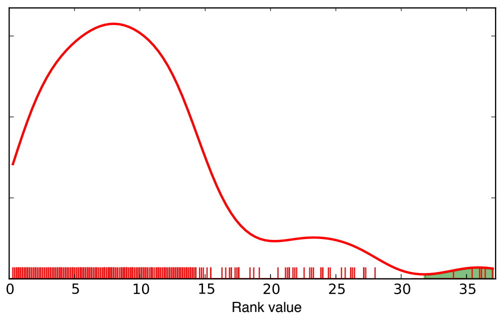


Figure 5: Example of the statistical test strategy for selecting how many tags to recommend. The curve represents the estimated PDF of C_{sorted}. Markers on the x-axis show the actual positions of candidate tags. Recommended tags are those under the shaded zone in the right.

5. EVALUATION METHODOLOGY

In order to compare and evaluate the efficacy of the proposed variants we followed a systematic approach based on removing a number of tags from the Freesound sound descriptions and then trying to predict them. The advantage of this approach is that it allows us to quickly evaluate different tag recommendation methods without the need of human input. The main drawback is that tags that could be subjectively considered as good recommendations for a particular sound description but that are not present in the set of removed tags, will not count as positive results (see Sec. 7).

We performed a 10-fold cross validation following the methodology described in [21]. For each fold, we build a tag similarity matrix using the subset of the folksonomy corresponding to the training set of sounds. Then, for each one of the sounds in the evaluation set, we remove a random number of their tags ($removedTags$) and run tag recommendation methods using the tag similarity matrix derived from the training set. We compute standard precision, recall and f-measure for each evaluated sound according to:

Method name	Aggregation step	Selection step
Proposed algorithm variants		
RankST	Rank-based	Statistical test
SimST	Similarity-based	Statistical test
RankLR	Rank-based	Linear regression
SimLR	Similarity-based	Linear regression
Basic methods		
RankFIX@K	Rank-based	Fixed number ($K \in [1, 10]$)
SimFIX@K	Similarity-based	Fixed number ($K \in [1, 10]$)
Repeated@R	Repeated tags in C_{raw} ($R \in [2, 6]$)	
Random baselines		
Random (for every method)	Random selection of tags from C_{raw}, with the same length as $recommendedTags$	

Table 1: Evaluated tag recommendation methods.

$$precision = \frac{|recommendedTags \cap removedTags|}{|recommendedTags|},$$

$$recall = \frac{|recommendedTags \cap removedTags|}{|removedTags|}, \text{ and}$$

$$f_{measure} = 2 \cdot \frac{precision \cdot recall}{precision + recall}.$$

Table 1 shows the tag recommendation methods that we compare. The first group of methods (*Proposed algorithm variants*) are the four possible combinations of aggregation and selection strategies described in Secs. 4.2 and4.3. *Basic methods* correspond to more basic tag recommendation methods that we used for comparison. On the one hand, we compare with two simpler versions of our proposed algorithm (RankFIX@K and SimFIX@K) which skip the last step of the recommendation process and always recommend a fixed number of K tags. We run these methods for values of K ranging from 1 to 10. On the other hand, we compare with an even simpler method (Repeated@R), which only recommends tags that appear more than R times in C_{raw} (independently of any rank or similarity values). We run these methods for values of R ranging from 2 to 6. Finally, we also compute a random baseline for each one of the previous methods by replacing the set of $recommendedTags$ with a random selection (of the same length) taken from C_{raw}. We choose as the general random baseline the one that gets the highest f-measure.

6. RESULTS

Table 2 shows the results of our evaluation as described in the previous section. The first group of results (under the label *with input tags range filter*) has been obtained by limiting the number of input tags we used to feed our algorithms to the range of $[3, 15]$, thus avoiding scarcely tagged sounds. The second group of results does not apply any restriction to the number of input tags (provided that there is at least one input tag).

A first observation is that using the input tags range filter produces an average increase in f-measure of 0.150 among our proposed algorithm variants (statistically significant using pairwise Kruskal-Wallis test, p≈0). Other methods present an average increase of 0.074 (p≈0). This means that, as expected, our algorithm works better in the range of $[3, 15]$ input tags. This is due to the notable increase in recall when using the input tags range filter (bigger than the increase in precision), suggesting that when feeded with at least three tags, our algorithm is able to select more relevant candidates. That supports the idea that for sounds with less tags, content-based approaches for tag recommendation would probably be more appropriate.

Algorithm	Precision	Rrecall	F-measure
With input tags range filter (83,010 sounds)			
RankST	0.443	0.537	0.432
RankLR	0.394	0.564	0.419
RankFIX@2	0.395	0.466	0.391
SimLR	0.348	0.397	0.325
SimST	0.381	0.333	0.317
RankFIX@5	0.233	0.614	0.308
SimFIX@2	0.303	0.344	0.294
SimFIX@5	0.181	0.467	0.237
Repeated@3	0.177	0.679	0.236
RankFIX@10	0.136	0.696	0.212
SimFIX@10	0.111	0.566	0.173
Random	0.006	0.033	0.010
Without input tags range filter (118,620 sounds)			
RankST	0.317	0.290	0.258
RankFIX@2	0.310	0.246	0.244
RankFIX@5	0.214	0.366	0.238
RankLR	0.236	0.301	0.221
SimLR	0.271	0.223	0.212
SimST	0.294	0.195	0.202
SimFIX@2	0.256	0.195	0.196
SimFIX@5	0.176	0.294	0.193
RankFIX@10	0.142	0.447	0.192
SimFIX@10	0.120	0.371	0.161
Repeated@3	0.095	0.262	0.110
Random	0.020	0.054	0.026

Table 2: Average of precision, recall and f-measure results for the evaluated tag recommendation methods. For the sake of readability, we only show some representative results of FIX and Repeated methods using $K = 2, 5, 10$ and $R = 3$. Methods are ordered by f-measure.

We can also see that all methods using the rank-based strategy for aggregating candidate tags always report higher f-measure than their similarity-based counterparts. Among the group with the input tags range filter, the average increase is of 0.104 (p≈0), while in the group without the filter the increase is of 0.033 (p≈0). That difference between both groups suggests that rank-based strategy works better when aggregating longer sets of candidates.

Regarding the selection step of our algorithm, both using the statistical test (ST) or the linear regression (LR) strategy significantly improves the performance with respect to the basic methods. When using the input tags filter, we observe an average increase in f-measure of 0.114 and 0.111 for the ST and LR methods, respectively (p≈0). Without using the filter the average increase is less important, of 0.045 and 0.036 for ST and LR, respectively (p≈0). It is surprising that methods recommending a fixed number of two tags (FIX@2) perform quite close to their counterparts using ST or LR strategies (and even in some cases scoring higher when not using the input tags range filer). This might be due to the fact that the average number of removed tags among all the experiments is 2.5. There-

Sound id	Input tags	Removed tags	Recommended tags	F-measure
8780	analog, glitch, warped	lofi	noise, electronic	0.0
124021	newspaper, reading, paper, page, news	read	magazine	0.0
38006	hit, glass, oneshot	percussion	singlehit, singlebeat, single, tap, hits, house, **percussion**, place, thuds, drum, plock	0.17
54374	spring, nightingale, nature, bird	field-recording, birdsong, binaural	birds, **field-recording**, forest, **birdsong**	0.5
78282	metal, medium-loud, interaction	impact	**impact**, wood	0.67

Table 3: Example of tag recommendations using the method RankST. Corresponding sounds can be listened at the following url: `http://www.freesound.org/search?q=`[Sound id].

fore, precision errors are minimized when recommending that amount of tags. Moreover, the good performance of FIX@2 reflects the effectiveness of the aggregation strategies, that successfully promote the most relevant tags on the first positions. On the other hand, ST and LR strategies perform generally better while at the same time recommending more tags (average of 3.16 and 4.6 respectively). This suggests that the selection step is able to choose, for each sound, the appropriate number of tags to recommend. Overall, the method that reports the highest f-measure is RankST (both with and without the input tags filter), and all our proposed algorithm variants perform much better than the random baseline.

7. CONCLUSION AND FUTURE WORK

In this paper we have described and evaluated four algorithm variants for tag recommendation based on the Freesound folksonomy. We have found that using a ranking instead of raw tag similarity values for sorting a list of candidate tags produces significantly better recommendations. The most novel aspect of the described algorithm is a step focused in automatically selecting the number of tags to recommend from a sorted list of candidate tags. The two strategies proposed for this step (statistical test and linear regression) have proved to be effective and statistically significantly increase the performance of the algorithm.

Although the systematic evaluation we have conducted allowed us to compare the different tag recommendation methods using a lot of sounds, the results in terms of f-measure are probably much worse than what a user-based evaluation could have reported. To exemplify this observation, Table 3 shows a few examples of tag recommendations performed using the RankST method (the one with the highest f-measure). We have marked in bold the tags that are considered good recommendations under our evaluation framework. Notice that many of the recommended tags which are not in italics could also be judged as meaningful recommendations if we listen to the sounds. In future work we would like to perform some user-based evaluation. Additionally, we plan to further improve our tag recommendation algorithm by introducing more tag-specific information such as characterizations of tag relevance, semantic category or usage context. Finally, we also plan to include our tag recommendation system in future deployments of Freesound.

8. ACKNOWLEDGEMENTS

This work is partially supported under BES-2010-037309 FPI grant from the Spanish Ministry of Science and Innovation for the TIN2009- 14247-C02-01 DRIMS project. JS acknowledges 2009-SGR-1434 from Generalitat de Catalunya, JAEDOC069/2010 from Consejo Superior de Investigaciones Científicas, TIN2009-13692-C03-01 from the Spanish Government, and EU Feder Funds.

9. REFERENCES

[1] V. Akkermans et al.:"Freesound 2: An Improved Platform for Sharing Audio Clips," *Late-braking demo abstract of the Int. Soc. for Music Information Retrieval Conf.*, 2011.

[2] H. Halpin et al.: "The dynamics and semantics of collaborative tagging," *Proc. of the 1st Semantic Authoring and Annotation Workshop*, 1-21, 2006.

[3] S. A. Golder and B. A. Huberman: "Usage patterns of collaborative tagging systems," *Journal of Information Science*, 32(2), 198-208, 2011.

[4] C. Marlow et al.: "HT06, Tagging Paper, Taxonomy, Flickr, Academic Article, ToRead," *Proc. of the 17th Conf. on Hypertext and Hypermedia*, 31-40, 2006.

[5] U. Farooq et al.: "Evaluating Tagging Behavior in Social Bookmarking Systems: Metrics and design heuristics," *Human-Computer Interaction*, 1, 351-360, 2007.

[6] K. Bischoff et al.:"Can all tags be used for search?," *Proc. of the 17th ACM Conf. on Information and Knowledge Management*, 32(2), 193202. ACM, 2008.

[7] I. Cantador et al.: "Categorising social tags to improve folksonomy-based recommendations," *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(1), 1-15, 2011.

[8] I. Ivanov et al.:"Object-based tag propagation for semi-automatic annotation of images," *Proc. of the Int. Conf. on Multimedia Information Retrieval*, 497-506, 2010.

[9] B. Sigurbjörnsson and R. Van Zwol: "Flickr tag recommendation based on collective knowledge," *Proc. of the 17th Int. Conf. on World Wide Web*, 327-336, 2008.

[10] P. De Meo et al.: "Exploitation of semantic relationships and hierarchical data structures to support a user in his annotation and browsing activities in folksonomies," *Information Systems Journal*, 34(6), 511-535, 2009.

[11] R. Jäschke et al.: "Tag Recommendations in Folksonomies," *Knowledge Discovery in Databases PKDD*, 34(6), 506-514, 2009.

[12] M. Sordo: "Semantic Annotation of Music Collections: A Computational Approach," *PhD thesis*, Universitat Pompeu Fabra, 2012.

[13] E. Martínez et al.: "Extending the folksonomies of freesound.org using content-based audio analysis," *Proc. of the Sound and Music Computing Conf.*, 23-25, 2009.

[14] D. Turnbull et al.: "Semantic Annotation and Retrieval of Music and Sound Effects," *IEEE Transactions On Audio Speech And Language Processing*, 16, 467-476 2008.

[15] L. Barrington et al.: "Audio Information Retrieval using Semantic Similarity," *IEEE Int. Conf. on In Acoustics, Speech and Signal Processing*, 16, 725-728, 2007.

[16] T. Vander Wal: "Explaining and showing broad and narrow folksonomies," http://www.personalinfocloud.com/ 2005/02/explaining_and_.html, 2005.

[17] P. Mika: "Ontologies are Us: A Unified Model of Social Networks and Semantics," *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1), 5-15, 2007.

[18] C. Cattuto et al.: "Semantic Analysis of Tag Similarity Measures in Collaborative Tagging Systems," *Data Engineering*, 805, 5-11, 2008.

[19] B.W. Silverman: "Density Estimation for Statistics and Data Analysis," *Applied Statistics*, 37(1), 1986.

[20] F. W. Scholz and M. A. Stephens: "K-Sample Anderson-Darling Tests," *Journal of the American Statistical Association*, 82, 918-924, 1987.

[21] S. L. Salzberg: "On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach ," *Data Mining and Knowledge Discovery*, 1(3), 317-328, 1997.

INFINITE COMPOSITE AUTOREGRESSIVE MODELS FOR MUSIC SIGNAL 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 novel probabilistic models that can be used to estimate multiple fundamental frequencies (F0s) from polyphonic audio signals. These models are nonparametric Bayesian extensions of nonnegative matrix factorization (NMF) based on the source-filter paradigm, and in them an amplitude or power spectrogram is decomposed as the product of two kinds of spectral atoms (sources and filters) and time-varying gains of source-filter pairs. In this study we model musical instruments as autoregressive systems that combine two types of sources—periodic signals (comb-shaped densities) and white noise (flat density)—with all-pole filters representing resonance characteristics. One of the main problems with such *composite autoregressive models* (CARMs) is that the numbers of sources and filters should be given in advance. To solve this problem, we propose nonparametric Bayesian models based on gamma processes and efficient variational and multiplicative learning algorithms. These *infinite* CARMs (iCARMs) can discover appropriate numbers of sources and filters in a data-driven manner. We report the experimental results of multipitch analysis on the MAPS piano database.

1. INTRODUCTION

Multiple fundamental frequency estimation (a.k.a. *multipitch analysis*) is the basis of various kinds of music content analysis. Recently, nonnegative matrix factorization (NMF) has gained a lot of popularity [1–13]. The standard NMF approximates an amplitude or power spectrogram (nonnegative matrix) as the product of two nonnegative matrices, one of which is a compact set of spectral bases and the other of which is a set of the corresponding time-varying gains [15, 16]. Such low-rank approximation is well justified by the fact that each musical piece consists of only limited kinds of sounds that repeatedly appear. In addition, a practical advantage of NMF is that the bases and gains can be alternately optimized by using efficient iterative algorithms called *multiplicative update* (MU) rules. The standard NMF, however, has three fundamental limitations:

1. The spectral bases are time-invariant, and only their gains vary over time. A large number of *independent* bases are needed to fully represent the timbral variations of instrument spectra (e.g., envelopes) even if these spectra share the same fundamental frequency (F0). Such an unconstrained increase of model complexity is likely to result in optimization algorithms easily getting stuck in bad local optima.
2. A post-processing step for estimating the F0s from individual bases is required because the F0s are not parameterized for representing the spectral bases. If the shapes of spectral bases are unconstrained, the resulting bases often deviate from natural harmonic spectra. This makes F0 estimation difficult and we need to judge the existence of an F0.
3. The number of bases (model complexity) should be carefully specified in advance because it has a strong impact on the decomposition results. Note that this limitation is closely related to the first. A naive solution is to exhaustively test all possible complexities and find an optimal value, but such *model selection* is often computationally impractical.

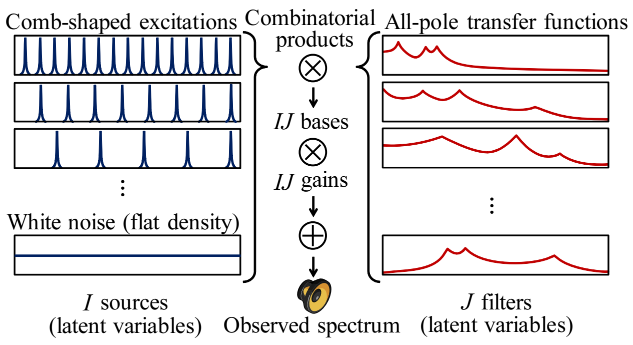

Comb-shaped excitations
Combinatorial products
All-pole transfer functions
IJ bases
IJ gains
White noise (flat density)
I sources (latent variables)
Observed spectrum
J filters (latent variables)

Figure 1. Overview of composite autoregressive models: The combinatorial products of I sources and J filters yield IJ spectral bases, which are activated according to the corresponding time-varying gains at each frame. We take the infinite limit as both I and J diverge to infinity.

As noted above, unconstrained NMF is too flexible for a set of musically-meaningful bases to be induced automatically. Although these limitations have partially been dealt with in previous studies [1–13], no study has overcome all of them simultaneously in a principled manner.

In this paper we propose *infinite composite autoregressive models* (iCARMs) (Fig. 1) developed for fusing the following techniques into a unified Bayesian framework:

1. **Source-filter factorization** (inspired by [1])
We further factorize the spectral bases as the combinatorial products of sources and all-pole filters. This

idea originates in the autoregressive (AR) modeling of speech signals: various vowels can be generated by changing the shape of the vocal tract (filter) while keeping the same F0 (source). This factorization enables us to represent a wide variety of instrumental sounds in terms of two separate aspects (timbre and F0) with reasonable complexity.

2. **Harmonicity modeling** (inspired by [7] and [9])
We represent each source as a comb-shaped function that uses an F0 parameter for representing equally-spanned harmonic partials of the same weight. Since such sources are multiplied by acoustically-inspired AR filters, the relative weights of partials of the bases are constrained to take realistic values as natural harmonic sounds. In addition, we can directly optimize the values of the F0s jointly with decomposition.
As proposed in [7, 14], we additionally introduce a special source representing white noise (flat density). This enables us to deal with percussive and transient sounds having widely distributed spectra. Their timbres (envelopes) are characterized by AR filters.

3. **Bayesian nonparametrics** (inspired by [12])
We build nonparametric Bayesian models that can automatically adjust the numbers of sources and filters needed to factorize a given spectrogram. Rather than these numbers being specified, the infinite limit of the conventional source-filter NMF [1] is taken as the numbers of sources and filters diverge to infinity. We perform sparse learning by introducing infinite-dimensional priors in such a way that only limited numbers of sources and filters are actually activated.

To optimize the iCARMs, we propose a new class of iterative algorithms that integrates a variational Bayesian (VB) technique with standard MU rules [8, 9].

The rest of this paper is organized as follows: Section 2 discusses the positioning of this study. Section 3 presents the iCARMs. Section 4 describes the evaluation. Section 5 concludes the paper with a mention of future work.

2. RELATED WORK

This section introduces two machine-learning (ML) techniques, i.e., NMF and Bayesian nonparametrics.

2.1 Nonnegative Matrix Factorization

NMF is a powerful tool for sparse decomposition of nonnegative matrix data [15]. It was first used for representing face images as linear combinations of a compact set of basis images corresponding to "local parts" such as eyes and noses. Such parts-based sparse representation is spontaneously induced by the nonnegativity constraint that allows only summation of basis images. Therefore, NMF fits naturally into audio spectrogram decomposition because the energy of harmonic sounds is concentrated at the discrete frequencies of harmonic partials.

2.1.1 Optimization Criteria

To perform NMF, we need some criterion for evaluating the "goodness-of-fit" of reconstructed data (linear combinations of spectral bases) to observed data (a spectrogram).

Method	Divergence	Sources	#	Filters	#
Kameoka [1]	IS	-	I	AR	J
Badeau [2]	IS	H	I	MA	J
Durrieu [3]	IS	(H)	I	-	J
Virtanen [4]	KL	-	I	-	J
Carabias-Orti [5]	KL	H	I	-	J
Heittola [6]	KL	-	I^*	-	J
Yasuraoka [7]	KL	H + N	I^*	AR	J
Hennequin [8]	Beta (0.5)	-	I	ARMA	J^*
Proposed	KL or IS	H + N	∞	AR	∞

(H: harmonic sources, N: noise source, -: others, $*$: varying over time)

Table 1. Several variants of source-filter NMF

As shown in Table 1, for example, Kullback-Leibler (KL) [15] and Itakura-Saito (IS) [16] divergences have been used intensively. Some studies used beta divergence [17], which includes KL and IS divergences as special cases.

In the context of audio modeling, although IS-NMF is justified in theory (see [16] and Section 3.2.1), KL-NMF often yields better results in a maximum likelihood estimation setting. One main reason is that the nonconvexity of IS divergence makes it difficult for gradient-descent-type optimization algorithms to find global optima. Note that no comparative tests have been conducted under a Bayesian estimation setting. In this paper we formulate two kinds of iCARMs, i.e., KL-iCARM and IS-iCARM.

2.1.2 Source-Filter Factorization

One extension is obtained with the source-filter paradigm, as listed in Table1. Kameoka and Kashino [1], for example, originally proposed the idea of the composite autoregressive model (CARM) using fixed numbers of unconstrained sources and autoregressive (AR) filters (all-pole transfer functions). Although similar models were devised by some researchers [3–5], filters were not acoustically constrained. Badeau *et al.* [2] used moving-average (MA) filters (all-zero transfer functions) with harmonic sources.

Some NMF variants allow sources or filters to vary over time to richly capture temporal variations of spectral bases at the cost of increasing complexity. Heittola *et al.* [6] and Yasuraoka and Okuno [7] used time-varying sources while a fixed number of filters was shared over time. Hennequin *et al.* [8], on the other hand, used time-varying ARMA filters that could be estimated by efficient MU rules.

2.1.3 Harmonicity Modeling

Another extension is based on harmonicity constraints on spectral bases or sources. For example, Vincent *et al.* [10] and Bertin *et al.* [11] assumed each basis as a weighted sum of narrowband template spectra consisting of a few adjacent harmonic partials. In the source-filter paradigm, Badeau *et al.* [2] represented each source as a binary vector whose elements are determined by independent Bernoulli trials, where particular elements corresponding to harmonic partials are more likely to take the value of 1. Yasuraoka and Okuno [7] and Hennequin *et al.* [9] represented each source as a parametric function based on a (weighted) sum of atomic functions (e.g., Gaussian functions) corresponding to harmonic partials. Carabias-Orti *et al.* [5] proposed to further factorize a set of partials' weights as a weighted sum of several patterns. Efficient MU rules for estimating the parameters of the function were proposed in [5, 9].

A key feature of [7] is to consider an additional source having a flat density. This idea was inspired by the speech production mechanism. Excitation signals produced by vocal cords are roughly categorized into periodic signals (harmonic comb-shaped spectra) and white noise (flat spectra). These signals are then articulated by the vocal tract whose resonance characteristics can be represented by AR filters. This assumption is widely accepted as reasonable to some extent for music signal modeling. In this study we model this generative process in a Bayesian framework.

2.2 Bayesian Nonparametrics

Another emerging ML technique is Bayesian nonparametrics [18], which is a generalization of the classical Bayesian technique. In the typical Bayesian framework, we put prior distributions on unknown random variables of interests and then, given observed data, estimate a posterior distribution over those variables. However, this framework cannot be used for determining model complexities (the numbers of sources and filters in this study) because these complexities are simply treated as hyperparameters. We thus have to use an expensive grid search for *combinatorial* model selection. Bayesian nonparametrics enables us to treat model complexities as random variables and estimate their optimal values jointly with posterior computation.

Bayesian modeling is being used in music signal analysis, and Bayesian extensions of NMF [19] have been used with great success for audio decomposition (source separation). An especially important breakthrough was recently made by Hoffman *et al.* [12]. They proposed a nonparametric Bayesian extension called the gamma-process NMF (GaP-NMF) that in theory allows an observed spectrogram to contain an infinite number of bases. A limited effective number of bases can be obtained by using an efficient variational inference algorithm. This extension is the basis of a more elaborate model that can consider infinite kinds of temporal variations of each basis [13].

3. PROPOSED MODELS

This section presents new nonparametric Bayesian models called infinite composite autoregressive models (iCARMs). The essential concept of these models is inspired by a composite autoregressive model (CARM) [1] that decomposes a *power* spectrogram into *fixed* numbers of sources and AR filters by using IS divergence as an optimization criterion. We formulate another CARM that decomposes an *amplitude* spectrogram by using KL divergence. To enforce harmonicity we explicitly represent each source—except for a single source that has a flat spectral density (white noise)—as a parametric comb-shaped function as proposed in [7]. Finally, both KL-CARM and IS-CARM are extended to in theory contain *infinite* numbers of sources and filters by using gamma processes as suggested in [12].

3.1 Overall Framework

We first define mathematical symbols as shown in Table 2. Let $\boldsymbol{X}$ be an $M \times N$ complex-valued spectrogram, where M is the number of frequency bins and N is the number of frames. Let I be the number of sources and J be the number of filters, which are assumed to go to infinity. Let the lower-case letters m, n, i, and j indicate the indices.

M	Number of frequency bins
N	Number of frames
I	Number of sources (diverges to infinity)
J	Number of filters (diverges to infinity)
X_{mn}	Amplitude (power) at m-th bin and n-th frame
Y_{mn}	Reconstructed value at m-th bin and n-th frame
θ_i	Global gain of i-th source
ϕ_j	Global gain of j-th filter
W_{im}	Amplitude (power) of i-th source at m-th bin
A_{jm}	Gain of j-th filter at m-th bin
H_{ijn}	Gain of i-th source & j-th filter pair at n-th frame

Table 2. Definition of mathematical symbols

In this paper we aim to factorize a nonnegative representation of $\boldsymbol{X}$ (amplitude or power spectrogram) into three kinds of "factors" $\boldsymbol{W}$, $\boldsymbol{A}$, and $\boldsymbol{H}$ as follows:

$$|X_{mn}| \text{ or } |X_{mn}|^2 \approx \sum_{i,j}^{I,J\to\infty} \theta_i \phi_j W_{im} A_{jm} H_{ijn} \tag{1}$$

where W_{im}, A_{jm}, and H_{ijn} respectively indicate the amplitude (power) of the i-th source at the m-th bin, the gain of the j-th filter at the m-th bin, and the gain of the i-th source and j-th filter pair at the n-th frame. In addition, two kinds of variables, θ_i and ϕ_j, are introduced to respectively indicate the *global* gain of the i-th source and the *global* gain of j-th filter over all N frames. Even when I and J diverge to infinity, finite numbers of the elements of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are expected to be substantially greater than zero while all other elements are negligibly small. This makes it possible for the "effective" numbers of sources and filters, I^+ and J^+, to be estimated in a data-driven manner.

Our goal is, given the spectrogram $\boldsymbol{X}$, to compute a posterior distribution $p(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}|\boldsymbol{X}; \boldsymbol{W}, \boldsymbol{A})$ over random variables and estimate parameters that represent $\boldsymbol{W}$ and $\boldsymbol{A}$. We will discuss concrete forms of priors $p(\boldsymbol{\theta})$, $p(\boldsymbol{\phi})$, $p(\boldsymbol{H})$, likelihood $p(\boldsymbol{X}|\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}; \boldsymbol{W}, \boldsymbol{A})$, and parametric functions of $\boldsymbol{W}$ and $\boldsymbol{A}$ according to KL or IS divergence.

3.2 Mathematical Formulation

We explain the different formulations of iCARMs based on KL and IS divergences.

3.2.1 Observation Likelihoods for $\boldsymbol{X}$

We use KL or IS divergence as an optimization criterion. Let Y_{mn} be $\sum_{ij} Y_{mn}^{ij}$, where $Y_{mn}^{ij} = \theta_i \phi_j W_{im} A_{jm} H_{ijn}$. We aim to optimize Y_{mn} such that the KL or IS divergence between X_{mn} and Y_{mn} is minimized, as shown in Eq.(1). This is known to be equivalent to maximum likelihood estimation of a Poisson or exponential distribution having Y_{mn} as its parameter, given an observation X_{mn} [16]. We here introduce a complex-valued latent variable X_{mn}^{ij} that indicates the contribution of the i-th source and j-th filter pair in X_{mn} such that $X_{mn} = \sum_{ij} X_{mn}^{ij}$ is satisfied.

The KL-iCARM is based on an amplitude-additivity assumption; i.e., $|X_{mn}| = \sum_{ij} |X_{mn}^{ij}|$. This is obviously incorrect but is useful in practice. If $|X_{mn}^{ij}| \sim \text{Poisson}(Y_{mn}^{ij})$, the reproductive property of the Poisson distribution leads to $|X_{mn}| \sim \text{Poisson}(\sum_{ij} Y_{mn}^{ij})$, which means

$$|X_{mn}| \sim \text{Poisson}\left(Y_{mn}\right) \tag{2}$$

The IS-iCARM is based on a complex-domain additivity assumption (see Section 3.2.5). If $X_{mn}^{ij} \sim \mathcal{N}_c(0, Y_{mn}^{ij})$, the reproductive property of the complex Gaussian leads to $X_{mn} \sim \mathcal{N}_c(0, \sum_{ij} Y_{mn}^{ij})$. This assumption, however, may be violated when the sources are *not* stationary Gaussian noise (see Section 3.2.4). We nonetheless assume

$$|X_{mn}|^2 \sim \text{Exponential}\,(Y_{mn}) \tag{3}$$

3.2.2 Gamma Process Priors on $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$

We put gamma process (GaP) priors on infinite-dimensional vectors $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$. More specifically, we introduce independent gamma priors on elements of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ as follows:

$$\theta_i \sim \text{Gamma}\left(\frac{\alpha}{I}, \alpha\right), \quad \phi_j \sim \text{Gamma}\left(\frac{\gamma}{J}, \gamma\right) \tag{4}$$

As the truncation level I diverges to infinity, the vector $\boldsymbol{\theta}$ approximates an infinite sequence drawn from a GaP with shape parameter α. It is proven that the *effective* number of elements, I^+, such that $\theta_i > \epsilon$ for some number $\epsilon > 0$ is almost surely finite. If we set I to be sufficiently larger than α, we can expect that only a few of the I elements of $\boldsymbol{\theta}$ will be substantially greater than zero. This condensation property enables sparse learning in an infinite space. The same reasoning can be applied to the GaP on $\boldsymbol{\phi}$.

3.2.3 Gamma Chain Priors on $\boldsymbol{H}$

To impose smooth transitions on $\boldsymbol{H}$, we put a gamma chain prior [20] on a temporal sequence of gains of each source-filter pair. More specifically, $\boldsymbol{H}_{ij}$ is modeled as follows:

$$\begin{aligned} H_{ij1} &\sim \text{Gamma}\,(\beta, \beta/d) \\ G_{ijn} &\sim \text{Gamma}\,(\beta, \beta H_{ijn-1}) \\ H_{ijn} &\sim \text{Gamma}\,(\beta, \beta G_{ijn}) \end{aligned} \tag{5}$$

where β is a hyperparameter that controls the strength of the priors (degree of smoothness) and G_{ijn} is an auxiliary variable that imposes a positive correlation between temporally adjacent gains H_{ijn-1} and H_{ijn} ($\mathbb{E}_{\text{prior}}[G_{ijn}] = H_{ijn-1}^{-1}$ and $\mathbb{E}_{\text{prior}}[H_{ijn}] = G_{ijn}^{-1}$). Marginalizing G_{ijn} out, we obtain a positively correlated Markovian transition kernel as $p(H_{ijn}|H_{ijn-1}) = \frac{\Gamma(2\beta)}{2\Gamma(\beta)} \frac{(H_{ijn-1}H_{ijn})^\beta}{(H_{ijn-1}+H_{ijn})^{2\beta}} H_{ijn}^{-1}$.

3.2.4 Comb-shaped Functions for $\boldsymbol{W}$

We represent each harmonic source $\boldsymbol{W}_i$ as a comb-shaped function that is the sum of H Gaussian functions, where H is the number of harmonic partials. Specifically,

$$W_{im} = \sum_{h=1}^{H} \exp\left(-\frac{(m - h\mu_i)^2}{2\sigma^2}\right) \tag{6}$$

where μ_i indicates F0[1] and σ indicates an energy diffusion around the frequencies of partials. Note that only the last source is reserved as white noise, i.e., $W_{Im} = 1$.

3.2.5 All-pole Transfer Functions for $\boldsymbol{A}$

We assume each basis signal $\boldsymbol{x}^{ij} \equiv \{x_t^{ij}\}_{t=1}^{2M}$ in a frame to be represented as a P-order AR process as follows:

$$x_t^{ij} = -\sum_{p=1}^{P} a_p^j x_{t-p}^{ij} + s_t^i \tag{7}$$

[1] When the value of F0 is given by $\tilde{\mu}_i$ [Hz], $\mu_i = \tilde{\mu}_i/(r/2M)$ [bins], where r is a sampling rate and $2M$ is a window size of frequency analysis.

where $\boldsymbol{s}^i \equiv \{s_t^i\}_{t=1}^{2M}$ is a signal of the i-th source and $\boldsymbol{a}_j \equiv \{a_0^j, \cdots, a_p^j\}^T$ is a coefficient vector of the j-th filter ($a_0^j = 1$). Let $\boldsymbol{w}^i \equiv \{w_t^i\}_{t=1}^{2M}$ be the autocorrelation of $\boldsymbol{s}^i$ and $\{W_{im}\}_{m=1}^{2M}$ be a complex (amplitude) spectrum density obtained by discrete Fourier transform (DFT) of $\boldsymbol{w}^i$. Let $\{X_m^{ij}\}_{m=1}^{2M}$ be a complex spectrum density obtained by DFT of $\boldsymbol{x}^{ij}$. *If the source signal $\boldsymbol{s}^i$ is a stationary Gaussian noise*, each X_m^{ij} is independently distributed as a complex Gaussian $\mathcal{N}_c(0, \Sigma_m^{ij})$, where $\Sigma_m^{ij} = W_{im}A_{jm}$ and

$$A_{jm} = \frac{1}{\left|\sum_{p=0}^{P} a_p^j e^{-2\pi \frac{m}{2M} pi}\right|^2} = \frac{1}{\boldsymbol{a}_j^T \boldsymbol{U}_m \boldsymbol{a}_j} \tag{8}$$

$\boldsymbol{U}_m$ is a $(P+1) \times (P+1)$ Toeplitz matrix with $[\boldsymbol{U}_m]_{pq} = \cos(2\pi \frac{m}{2M}(p-q))$. This means that $|X_m^{ij}|^2$ is distributed as an exponential distribution having $W_{im}A_{jm}$ as its scale parameter. In other words, maximum likelihood estimation of $\boldsymbol{a}_j$ for $\boldsymbol{x}^{ij}$ is equivalent to minimizing the IS divergence between $\{|X_m^{ij}|^2\}_{m=1}^M$ and $\{W_{im}A_{jm}\}_{m=1}^M$ [2].

In the iCARMs based on KL and IS divergences, the above discussion leads to the following formulations:

$$A_{jm}^{\text{KL}} = \sqrt{\frac{1}{\boldsymbol{a}_j^T \boldsymbol{U}_m \boldsymbol{a}_j}} \quad \text{or} \quad A_{jm}^{\text{IS}} = \frac{1}{\boldsymbol{a}_j^T \boldsymbol{U}_m \boldsymbol{a}_j} \tag{9}$$

A reason for taking the "root" in the KL-iCARM is that we assume an "amplitude" spectrogram as observed data.

3.3 Variational and Multiplicative Optimization

The posterior over random variables $p(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}|\boldsymbol{X}; \boldsymbol{W}, \boldsymbol{A})$ and $\boldsymbol{W}$ and $\boldsymbol{A}$ (parameters $\boldsymbol{\mu}$, σ, and $\boldsymbol{a}$) are determined such that the log-evidence $\log p(\boldsymbol{X}; \boldsymbol{W}, \boldsymbol{A})$ is maximized. Since this cannot be analytically computed, we use an approximate method called variational Bayes (VB), which restricts the posterior to a factorized form given by

$$q(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}) = \prod_i q(\theta_i) \prod_j q(\phi_j) \prod_{ijn} q(H_{ijn}) \tag{10}$$

and iteratively updates this form by monotonically increasing a *lower bound*[3] of the log-evidence, $\mathcal{L}$, given by

$$\begin{aligned} \log p(\boldsymbol{X}; \boldsymbol{W}, \boldsymbol{A}) &\geq \mathbb{E}[\log p(\boldsymbol{X}|\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}; \boldsymbol{W}, \boldsymbol{A})] \\ &+ \mathbb{E}[\log p(\boldsymbol{\theta})] + \mathbb{E}[\log p(\boldsymbol{\phi})] + \mathbb{E}[\log p(\boldsymbol{H})] \\ &- \mathbb{E}[\log q(\boldsymbol{\theta})] - \mathbb{E}[\log q(\boldsymbol{\phi})] - \mathbb{E}[\log q(\boldsymbol{H})] \equiv \mathcal{L} \end{aligned} \tag{11}$$

The iterative update rules are

$$\begin{aligned} q(\boldsymbol{\theta}) &\propto \exp(\mathbb{E}_{q(\boldsymbol{\phi},\boldsymbol{H})}[\log p(\boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}; \boldsymbol{W}, \boldsymbol{A})]) \\ q(\boldsymbol{\phi}) &\propto \exp(\mathbb{E}_{q(\boldsymbol{\theta},\boldsymbol{H})}[\log p(\boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}; \boldsymbol{W}, \boldsymbol{A})]) \\ q(\boldsymbol{H}) &\propto \exp(\mathbb{E}_{q(\boldsymbol{\theta},\boldsymbol{\phi})}[\log p(\boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{H}; \boldsymbol{W}, \boldsymbol{A})]) \end{aligned} \tag{12}$$

To optimize $\boldsymbol{W}$ and $\boldsymbol{A}$ ($\boldsymbol{\mu}$, σ, and $\boldsymbol{a}$), we use multiplicative update (MU) rules inspired by [8,9]. A general rule is obtained from the partial derivative of a "cost" function, e.g., $-\mathcal{L}$. For example, if we can write the derivative as the difference of two positive terms, i.e., $\frac{-\partial \mathcal{L}}{\partial \mu_i} = G_{\mu_i} - F_{\mu_i}$, an update rule for μ_i is given by $\mu_i \leftarrow \mu_i \times \frac{F_{\mu_i}}{G_{\mu_i}}$. Note that μ_i becomes constant if the derivative is zero, and is updated in the opposite direction of the derivative. We omit detailed derivations and only describe update rules below.

[2] In linear predictive coding (LPC), the source signal $\boldsymbol{s}^i$ is generally limited to white noise ($W_{im} = 1$). This is a conventional assumption.

[3] More specifically, a further lower bound of $\mathcal{L}$ should be computed.

3.3.1 Variational Updates for KL-iCARM

The variational posterior of each random variable is set to be the same family as its prior distribution as follows:

$$q(\theta_i) = \text{Gamma}(a_i^\theta, b_i^\theta), \quad q(\phi_j) = \text{Gamma}(a_j^\phi, b_j^\phi)$$
$$q(H_{ijn}) = \text{Gamma}(a_{ijn}^H, b_{ijn}^H) \tag{13}$$

The variational parameters are given by

$$\begin{aligned}
a_i^\theta &= \frac{\alpha}{I} + \textstyle\sum_{mnj} |X_{mn}|\lambda_{mnij} \\
b_i^\theta &= \alpha + \textstyle\sum_{mnj} \mathbb{E}[\phi_j W_{im} A_{jm} H_{ijn}] \\
a_j^\phi &= \frac{\gamma}{J} + \textstyle\sum_{mni} |X_{mn}|\lambda_{mnij} \\
b_j^\phi &= \gamma + \textstyle\sum_{mni} \mathbb{E}[\theta_i W_{im} A_{jm} H_{ijn}] \\
a_{ijn}^H &= 2\beta + \textstyle\sum_m |X_{mn}|\lambda_{mnij} \qquad (14) \\
b_{ijn}^H &= \beta\mathbb{E}[G_{ijn}+G_{ijn+1}] + \textstyle\sum_m \mathbb{E}[\theta_i\phi_j W_{im} A_{jm}]
\end{aligned}$$

where λ_{mnij} is an auxiliary variable given by

$$\lambda_{mnij} \propto \exp(\mathbb{E}[\log(\theta_i\phi_j W_{im} A_{jm} H_{ijn})]) \tag{15}$$

3.3.2 Variational Updates for IS-iCARM

As proposed in [12], the variational posterior of each variable is given by a generalized inverse-Gaussian (GIG) distribution (see the Appendix) as follows:

$$q(\theta_i) = \text{GIG}(a_i^\theta, b_i^\theta, c_i^\theta), \quad q(\phi_j) = \text{GIG}(a_j^\phi, b_j^\phi, c_j^\phi)$$
$$q(H_{ijn}) = \text{GIG}(a_{ijn}^H, b_{ijn}^H, c_{ijn}^H) \tag{16}$$

The variational parameters are given by

$$\begin{aligned}
a_i^\theta &= \frac{\alpha}{I}, \quad b_i^\theta = \alpha + \textstyle\sum_{mnj} \frac{\mathbb{E}[\phi_j W_{im} A_{jm} H_{ijn}]}{\xi_{mn}} \\
c_i^\theta &= \textstyle\sum_{mnj} |X_{mn}|^2 \eta_{mnij}^2 \mathbb{E}\big[\frac{1}{\phi_j W_{im} A_{jm} H_{ijn}}\big] \\
a_j^\phi &= \frac{\gamma}{J}, \quad b_j^\phi = \gamma + \textstyle\sum_{mni} \frac{\mathbb{E}[\theta_i W_{im} A_{jm} H_{ijn}]}{\xi_{mn}} \\
c_j^\phi &= \textstyle\sum_{mni} |X_{mn}|^2 \eta_{mnij}^2 \mathbb{E}\big[\frac{1}{\theta_i W_{im} A_{jm} H_{ijn}}\big] \\
a_{ijn}^H &= 2\beta, \ c_{ijn}^H = \textstyle\sum_m |X_{mn}|^2 \eta_{mnij}^2 \mathbb{E}\big[\frac{1}{\theta_i\phi_j W_{im} A_{jm}}\big] \\
b_{ijn}^H &= \beta\mathbb{E}[G_{ijn}+G_{ijn+1}] + \textstyle\sum_m \frac{\mathbb{E}[\theta_i\phi_j W_{im} A_{jm}]}{\xi_{mn}} \qquad (17)
\end{aligned}$$

where η_{mnij} and ξ_{mn} are auxiliary variables given by

$$\begin{aligned}
\eta_{mnij} &\propto \mathbb{E}[\tfrac{1}{\theta_i\phi_j W_{im} A_{jm} H_{ijn}}]^{-1} \text{ s.t.} \textstyle\sum_{ij} \eta_{mnij} = 1 \\
\xi_{mn} &= \textstyle\sum_{ij} \mathbb{E}[\theta_i\phi_j W_{im} A_{jm} H_{ijn}] \qquad (18)
\end{aligned}$$

3.3.3 Multiplicative Updates for KL- and IS-iCARMs

The MU rules for $\boldsymbol{\mu}$, σ, and $\boldsymbol{a}$ are given by $\mu_i \leftarrow G_{\mu_i}^{-1} F_{\mu_i}\mu_i$, $\sigma^2 \leftarrow G_{\sigma^2}^{-1} F_{\sigma^2}\sigma^2$, and $\boldsymbol{a}_j \leftarrow \boldsymbol{G}_{\boldsymbol{a}_j}^{-1} \boldsymbol{F}_{\boldsymbol{a}_j} \boldsymbol{a}_j$, where

$$\begin{aligned}
F_{\mu_i} &= \textstyle\sum_{mnjh} h(mV_{mnij}^F + h\mu_i V_{mnij}^G)\exp\big(-\frac{(m-h\mu_i)^2}{2\sigma^2}\big) \\
G_{\mu_i} &= \textstyle\sum_{mnjh} h(mV_{mnij}^G + h\mu_i V_{mnij}^F)\exp\big(-\frac{(m-h\mu_i)^2}{2\sigma^2}\big) \\
F_{\sigma^2} &= \textstyle\sum_{mnijh} V_{mnij}^F (m-h\mu_i)^2 \exp\big(-\frac{(m-h\mu_i)^2}{2\sigma^2}\big) \\
G_{\sigma^2} &= \textstyle\sum_{mnijh} V_{mnij}^G (m-h\mu_i)^2 \exp\big(-\frac{(m-h\mu_i)^2}{2\sigma^2}\big) \\
\boldsymbol{F}_{\boldsymbol{a}_j}^{\text{KL}} &= \textstyle\sum_{mni} \theta_i\phi_j W_{im} H_{ijn} A_{jm}^3 \boldsymbol{U}_m \\
\boldsymbol{G}_{\boldsymbol{a}_j}^{\text{KL}} &= \textstyle\sum_{mni} |X_{mn}|\lambda_{mnij} A_{jm}^2 \boldsymbol{U}_m \\
\boldsymbol{F}_{\boldsymbol{a}_j}^{\text{IS}} &= \textstyle\sum_{mni} \frac{\mathbb{E}[\theta_i\phi_j W_{im} H_{ijn}]}{\xi_{mn}} A_{jm}^2 \boldsymbol{U}_m \\
\boldsymbol{G}_{\boldsymbol{a}_j}^{\text{IS}} &= \textstyle\sum_{mni} |X_{mn}|^2 \eta_{mnij}^2 \mathbb{E}\big[\frac{1}{\theta_i\phi_j W_{im} H_{ijn}}\big] \boldsymbol{U}_m \qquad (19)
\end{aligned}$$

V_{mnij}^F and V_{mnij}^G are given by $V_{mnij}^F = \mathbb{E}[\theta_i\phi_j A_{jm} H_{ijn}]$ and $V_{mnij}^G = |X_{mn}|\lambda_{mnij} W_{im}^{-1}$ in the KL-iCARM. On the other hand, $V_{mnij}^F = \frac{\mathbb{E}[\theta_i\phi_j A_{jm} H_{ijn}]}{\xi_{mn}}$ and $V_{mnij}^G = |X_{mn}|^2 \eta_{mnij}^2 \mathbb{E}\big[\frac{1}{\theta_i\phi_j W_{im}^2 A_{jm} H_{ijn}}\big]$ in the IS-iCARM.

4. EVALUATION

We report comparative experiments that were conducted to evaluate the performance of the iCARMs based on KL and IS divergences as multipitch analyzers.

4.1 Experimental Conditions

We used thirty pieces of "ENSTDkCl" subset included in the MAPS piano database [21]. We truncated each piece to 30 s as in [5,11] and converted the original CD-quality signals into monaural signals sampled at 16 [kHz]. The spectrograms were obtained with short-time Fourier transform (STFT) with a window size of 2048 samples and a shifting interval of 10 [ms], i.e., $M = 1024$ and $N = 3000$. The amplitude or power spectrogram of each piece was scaled such that $\frac{1}{MN}\sum_{mn}|X_{mn}| = 1$ or $\max_{mn}|X_{mn}|^2 = 1$. The hyperparameters were specified as $I = 88{+}1$, $J = 10$, $\alpha{=}1$, $\beta = \gamma = 0.1$, $H = 20$, $P = 4$, and $d{=}\mathbb{E}_{\text{emp}}[|X_{mn}|]$ or $\mathbb{E}_{\text{emp}}[|X_{mn}|^2]$. Although $J = 10$ was too small to accurately approximate the GaP, it was sufficiently large in our experiments because the audio signals contain only piano sounds. We initialized $\{\mu_i\}_{i=1}^{88}$ as the frequencies corresponding to the 88 keys of the standard piano. The other parameters were initialized randomly.

Multiple F0s were detected at each frame in a thresholding process. If the gain of the i-th source, $\sum_j \theta_i\phi_j H_{ijn}$, was larger than the threshold, we judged that the n-th frame includes an F0 indicated by μ_i. The threshold was globally determined such that the frame-level precision and recall rates were balanced to yield the best average F-measure.

4.2 Experimental Results

We first tested our models on toy data obtained by extracting the first 4.9 s (490 frames) of the piece "alb_se2," which contains five different F0s and a polyphony level that increases one by one up to five (D4, +C#4, +C4, +A3, +F#3). As shown in Fig. 2, the KL-iCARM could successfully discover the correct number of sources (five harmonic sources + one white-noise source) in a data-driven manner. In addition, we could separate $\boldsymbol{X}$ into harmonic and noise components by computing $\mathbb{E}[Y_{mn}^i] = \sum_j \mathbb{E}[\theta_i\phi_j W_{im} A_{jm} H_{ijn}]$, which represents the component of the i-th source at the m-th bin and n-th frame.

As shown in Fig. 3, on the other hand, the IS-iCARM overestimated the numbers of sources and filters and made many octave errors (half-F0 errors). One reason is that IS divergence permits a reconstructed power to exceed an observed power with a smaller penalty. It is therefore difficult to reduce false alarms of harmonic partials.

We then used the 30 pieces for evaluation. The KL- and IS-iCARMs achieved the frame-level F-measures of 48.4% and 35.1% respectively. Although these preliminary results were not really impressive compared with the state-of-the-art results [5,11], we consider our framework to be promising because of its elegant nature of sparse learning over an infinite space. A main limitation of the KL-iCARM is that we still need to resort a thresholding process for temporal gains although limited numbers of sources and filters can be obtained by using GaPs. One solution would be to introduce *binary* latent variables that indicate note existences.

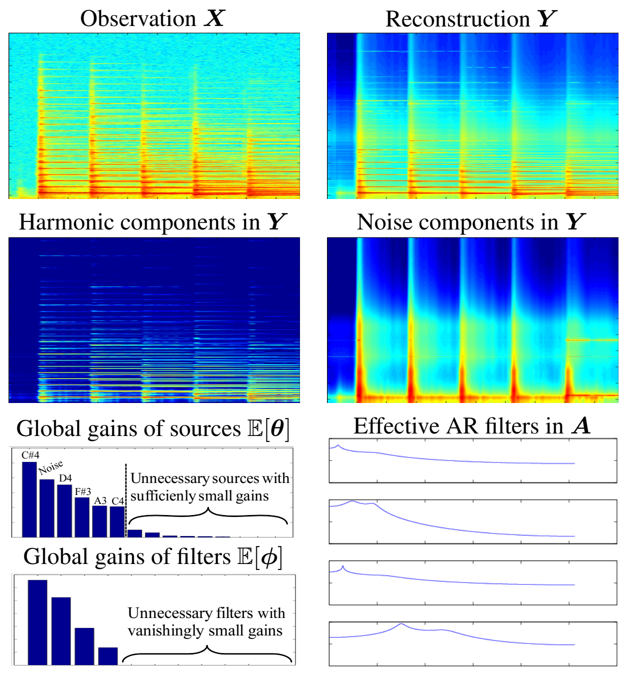


Figure 2. Decomposition results obtained by KL-iCARM

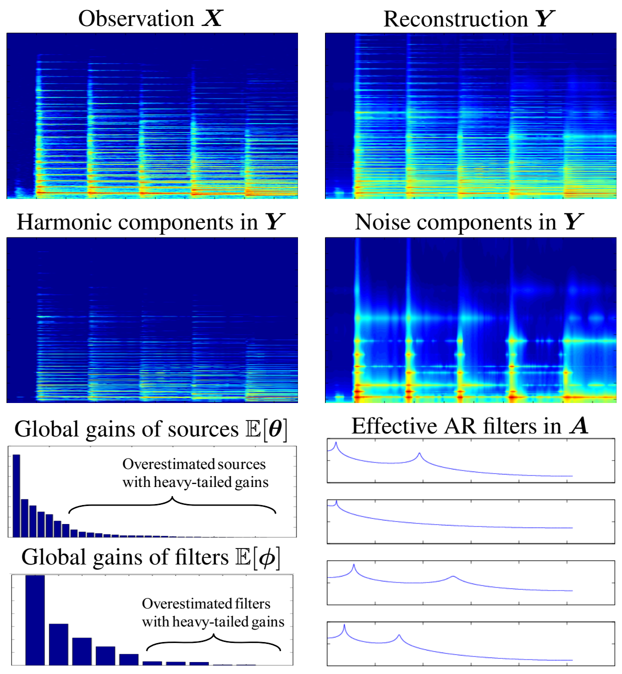


Figure 3. Decomposition results obtained by IS-iCARM

5. CONCLUSION

We presented nonparametric Bayesian models called infinite composite autoregressive models (iCARMs) that decompose an observed spectrogram into three kinds of factors, i.e., sources, filters, and time-varying gains of source-filter pairs. The experimental results showed that appropriate numbers of sources and filters can be discovered in a data-driven manner by using gamma processes for sparse learning. To improve the accuracy of multipitch analysis, we are considering the use of log-frequency spectrograms obtained by constant-Q or wavelet transform. We also plan to use these models for "timbre-based" source separation by distinguishing different resonance characteristics of instrument and vocal sounds by AR filters.

Acknowledgment: This study was supported by JSPS KAKENHI 23700184 and JST OngaCREST project.

6. REFERENCES

[1] H. Kameoka and K. Kashino. Composite autoregressive system for sparse source-filter representation of speech. *ICASSP*, pp. 2477–2480, 2009.

[2] R. Badeau, V. Emiya, and B. David. Expectation-maximization algorithm for multi-pitch estimation and separation of overlapping harmonic spectra. *ICASSP*, pp. 3073–3076, 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 Trans. on ASLP*, 18(3):564–575, 2010.

[4] T. Virtanen and A. Klapuri. Analysis of polyphonic audio using source-filter model and non-negative matrix factorization. *NIPS Ws. on Adv. in Mod. for Acoust. Proc.*, 2009.

[5] J. J. Carabias-Orti, T. Virtanen, P. Vera-Candeas, N. Ruiz-Reyes, and F. J. Cañadas-Quesada. Musical instrument sound multi-excitation model for non-negative spectrogram factorization. *IEEE J. of Sel. Top. in Sig. Proc.*, 5(6):1144–1158, 2011.

[6] T. Heittola, A. Klapuri, and T. Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. *ISMIR*, pp. 327–332, 2009.

[7] N. Yasuraoka and H. G. Okuno. Musical audio signal modeling for joint estimation of harmonic, inharmonic, and timbral structure and its application to source sepatation. *SIG Technical Reports*, volume 2012-MUS-94, pp. 1–8, 2012.

[8] R. Hennequin, R. Badeau, and B. David. NMF with time-frequency activations to model nonstationary audio events. *IEEE Trans. on ASLP*, 19(4):744–753, 2011.

[9] R. Hennequin, R. Badeau, and B. David. Time-dependent parametric and harmonic templates in non-negative matrix factorization. *DAFx*, pp. 1–8, 2010.

[10] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Trans. on ASLP*, 18(3):528–537, 2010.

[11] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in Bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Trans. on ASLP*, 18(3):538–549, 2010.

[12] M. Hoffman, D. Blei, and P. Cook. Bayesian nonparametric matrix factorization for recorded music. *ICML*, 2010.

[13] M. Nakano *et al.* Bayesian nonparametric spectrogram modeling based on infinite factorial infinite hidden Markov model. *WASPAA*, pp. 325–328, 2011.

[14] B. Niedermayer. Improving accuracy of polyphonic music-to-score alignment. *ISMIR*, pp. 585–590, 2009.

[15] D. Lee and H. Seung. Algorithms for non-negative matrix factorization. *NIPS*, pp. 556–562, 2000.

[16] C. Févotte, N. Bertin, and J.-L. Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis. *NECO*, 21(3):793–830, 2009.

[17] C. Févotte and J. Idier. Algorithms for nonnegative matrix factorization with the beta-divergence. *NECO*, 23(9):2421–2456, 2011.

[18] P. Orbanz and Y. W. Teh. Bayesian nonparametric models. *Encyclopedia of Machine Learning*. Springer, 2010.

[19] A. T. Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009:Article ID 785152, 2009.

[20] A. T. Cemgil and O. Dikmen. Conjugate gamma Markov random fields for modelling nonstationary sources. *ICA*, 2007.

[21] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Trans. on ASLP*, 18(6):1643–1654, 2010.

A. PROBABILITY DISTRIBUTIONS

$$\mathrm{Gamma}(x|a,b) = \frac{b^a}{\Gamma(a)} x^{a-1} e^{-bx} \quad \mathrm{Exponential}(x|\lambda) = \frac{1}{\lambda} e^{-\frac{x}{\lambda}}$$

$$\mathrm{Poisson}(x|\lambda) = \frac{\lambda^x}{x!} e^{-\lambda} \quad \mathrm{GIG}(x|a,b,c) = \frac{(b/c)^{\frac{a}{2}} x^{a-1}}{2\mathcal{K}_a(2\sqrt{bc})} e^{-(bx+\frac{c}{x})}$$

PROFESSIONALLY-PRODUCED MUSIC SEPARATION GUIDED BY COVERS

Timothée Gerber, Martin Dutasta, Laurent Girin
Grenoble-INP, GIPSA-lab
firstname.lastname@gipsa-lab.grenoble-inp.fr

Cédric Févotte
TELECOM ParisTech, CNRS LTCI
cedric.fevotte@telecom-paristech.fr

ABSTRACT

This paper addresses the problem of demixing professionally produced music, i.e., recovering the musical source signals that compose a (2-channel stereo) commercial mix signal. Inspired by previous studies using MIDI synthesized or hummed signals as external references, we propose to use the multitrack signals of a cover interpretation to guide the separation process with a relevant initialization. This process is carried out within the framework of the multichannel convolutive NMF model and associated EM/MU estimation algorithms. Although subject to the limitations of the convolutive assumption, our experiments confirm the potential of using multitrack cover signals for source separation of commercial music.

1. INTRODUCTION

In this paper, we address the problem of source separation within the framework of professionally-produced (2-channel stereo) music signals. This task consists of recovering the individual signals produced by the different instruments and voices that compose the mix signal. This would offer new perspectives for music active listening, editing and post-production from usual stereo formats (e.g., 5.1 upmixing), whereas those features are currently roughly limited to multitrack formats, in which a very limited number of original commercial songs are distributed.

Demixing professionally produced music (PPM) is particularly difficult for several reasons [11, 12, 17]. Firstly, the mix signals are generally underdetermined, i.e., there are more sources than mix channels. Secondly, some sources do not follow the point source assumption that is often implicit in the (convolutive) source separation models of the signal processing literature. Also, some sources can be panned in the same direction, convolved with large reverberation, or processed with artificial audio effects that are more or less easy to take into account in a separation framework. PPM separation is thus an ill-posed problem and separation methods have evolved from blind to *informed* source separation (ISS), i.e., methods that exploit some "grounded" additional information on the source/mix signals and mix process. For example, the methods in [1, 4, 5, 8, 20] exploit the musical score of the instrument to extract sources, either directly or through MIDI signal synthesis. In user-guided approaches, the listener can assist the separation process in different ways, e.g., by humming the source to be extracted [16], or by providing information on the sources direction [19] or temporal activity [12]. An extreme form of ISS can be found in [6, 9, 10, 14, 15] and in the Spatial Audio Object Coding (SAOC) technology recently standardized by MPEG [3]: here, the source signals themselves are used for separation, which makes sense only in a coder-decoder configuration.

In the present paper, we remain in the usual configuration where the original multitrack signals are not available, although we keep the latter spirit of using *source signals* to help the demixing process: we propose to use *cover multitrack signals* for this task. This idea is settled on several facts. Firstly, a cover song can be quite different from the original for the sake of artistic challenge. But very interestingly, for some applications/markets a cover song is on the contrary intended to be as close as possible to the original song: instruments composition and color, song structure (chorus, verses, solos), and artists interpretation (including the voices) are then closely fitted to the original source signals, hence having a potential for source separation of original mixes. Remarkably, it happens that multitracks of such "mimic" covers are relatively easy to find on the market for a large set of famous pop songs. In fact, they are much easier to obtain than original multitracks. This is because the music industry is very reluctant to release original works while it authorizes the licensed production of mimic multitracks on a large scale. In the present study, we will use such multitracks provided by iKlax Media which is a partner of the DReaM project.[1] iKlax Media produces software solutions for music active listening and has licensed the exploitation of a very large set of cover multitracks of popular songs. Therefore, this work involves a sizeable artistic and commercial stake. Note that similar material can be obtained from several other companies.

We set the cover-informed source separation principle within the currently very popular framework of separation methods based on a local time-frequency (TF) complex Gaussian model combined with a non-negative matrix factorization (NMF) model for the source variances [7, 11, 13].

[1] This research is partly funded by the French National Research Agency (ANR) – Grant CONTINT 09-CORD-006.

Iterative NMF algorithms for source modeling and separation have shown to be very sensitive to initialization. We turn this weakness into strength within the following two-step process in the same spirit as the work carried out on signals synthesized from MIDI scores in, e.g., [8] or by humming in [16]. First, source-wise NMF modeling is applied on the cover multitrack, and the result is assumed to be a suitable initialization of the NMF parameters of the original sources (that were used to produce the commercial mix signal). Starting from those initial values, the NMF process is then refined by applying to the mix the convolutive multichannel NMF model of [11]. This latter model provides both refined estimation of the source-within-mix (aka source images) NMF parameters and source separation using Wiener filters built from those parameters.

The paper is organized as follows. In Sections 2 and 3, we respectively present the models and method employed. In Sections 4 and 5, we present the experiments we conducted to assess the proposed method, and in Section 6, we address some general perspectives.

2. FRAMEWORK: THE CONVOLUTIVE MULTICHANNEL NMF MODEL

2.1 Mixing Model

Following the framework of [11], the PPM multichannel mix signal $x(t)$ is modeled as a convolutive noisy mixture of J source signals $s_j(t)$. Using the short-time Fourier transform (STFT), the mix signal is approximated in the TF domain as:

$$\mathbf{x}_{fn} = \mathbf{A}_f \mathbf{s}_{fn} + \mathbf{b}_{fn}, \tag{1}$$

where $\mathbf{x}_{fn} = [x_{1,fn}, \ldots, x_{I,fn}]^T$ is the vector of complex-valued STFT coefficients of the mix signal, $\mathbf{s}_{fn} = [s_{1,fn}, \ldots, s_{J,fn}]^T$ is the vector of complex-valued STFT coefficients of the sources, $\mathbf{b}_{fn} = [b_{1,fn}, \ldots, b_{I,fn}]^T$ is a zero-mean Gaussian residual noise, $\mathbf{A}_f = [\mathbf{a}_{1,f}, \ldots, \mathbf{a}_{J,f}]$ is the frequency-dependent mixing matrix of size $I \times J$ ($\mathbf{a}_{j,f}$ is the mixing vector for source j), $f \in [0, F-1]$ is the frequency bin index and $n \in [0, N-1]$ is the time frame index. This approach implies standard narrowband assumption (i.e., the time-domain mixing filters are shorter than the STFT window size).

2.2 Source model

Each source $s_{j,fn}$ is modeled as the sum of K_j latent components $c_{k,fn}$, $k \in \mathcal{K}_j$, i.e.,

$$s_{j,fn} = \sum_{k \in \mathcal{K}_j} c_{k,fn}, \tag{2}$$

where $\{\mathcal{K}_j\}_j$ is a non-trivial partition of $\{1, \ldots, K\}$, $K \geq J$ (K_j is thus the cardinal of $\mathcal{K}_j$). Each component $c_{k,fn}$ is assumed to follow a zero-mean proper complex Gaussian distribution of variance $w_{fk}h_{kn}$, where $w_{fk}, h_{kn} \in \mathbb{R}^+$, i.e., $c_{k,fn} \sim \mathcal{N}_c(0, w_{fk}h_{kn})$. The components are assumed to be mutually independent and individually independent across frequency and time, so that we have:

$$s_{j,fn} \sim \mathcal{N}_c(0, \sum_{k \in \mathcal{K}_j} w_{fk}h_{kn}). \tag{3}$$

This source model corresponds to the popular non-negative matrix factorization (NMF) model as applied to the source power spectrogram $|\mathbf{S}_j|^2 = \{|s_{j,fn}|^2\}_{fn}$:

$$|\mathbf{S}_j|^2 \simeq \mathbf{W}_j \mathbf{H}_j, \tag{4}$$

with non-negative matrices $\mathbf{W}_j = \{w_{fk}\}_{f,k \in \mathcal{K}_j}$ of size $F \times K_j$ and $\mathbf{H}_j = \{h_{kn}\}_{k \in \mathcal{K}_j, n}$ of size $K_j \times N$. The columns of $\mathbf{W}_j$ are generally referred to as *spectral pattern vectors*, and the rows of $\mathbf{H}_j$ are referred to as *temporal activation vectors*. NMF is largely used in audio source separation since it appropriately models a large range of musical sounds by providing harmonic patterns as well as non-harmonic ones (e.g., subband noise).

2.3 Parameter estimation and source separation

In the source modeling context, the NMF parameters of a given source signal can be obtained from the observation of its power spectrogram using Expectation-Maximization (EM) iterative algorithms [7]. In [11], this has been generalized to the joint estimation of the J sets of NMF source parameters and $I \times J \times F$ mixing filters parameters from the observation of the mix signal power spectrogram. More precisely, two algorithms were proposed in [11]. An EM algorithm consists of maximizing the exact joint likelihood of the multichannel data, whereas a multiplicative updates (MU) algorithm, maximizes the sum of individual channel log-likelihood. If the former better exploits the inter-channel dependencies and gives better separation results, [2] the latter has a lower computation cost. Those algorithms will not be described in the present paper, the reader is referred to [11] for technical details.

Once all the parameters are estimated, the source signals (or their spatial images $\mathbf{y}_{j,fn} = \mathbf{a}_{j,f} s_{j,fn}$) are estimated using spatial Wiener filtering of the mix signal:

$$\hat{\mathbf{s}}_{fn} = \mathbf{\Sigma}_{\mathbf{s},fn} \mathbf{A}_f^H \mathbf{\Sigma}_{\mathbf{x},fn}^{-1} \mathbf{x}_{fn}, \tag{5}$$

where $\mathbf{\Sigma}_{\mathbf{s},fn}$ is the (estimated) covariance matrix of the source signals, and $\mathbf{\Sigma}_{\mathbf{x},fn} = \mathbf{A}_f \mathbf{\Sigma}_{\mathbf{s},fn} \mathbf{A}_f^H + \mathbf{\Sigma}_{\mathbf{b},f}$ is the (estimated) covariance matrix of the mix signal.

3. PROPOSED COVER-INFORMED SEPARATION TECHNIQUE

3.1 Cover-based initialization

It is well-known that NMF decomposition algorithms are highly dependent on the initialization. In fact, the NMF model does not guarantee the convergence to a global minimum but only to a local minimum of the cost function, making a suitable initialization crucial for the separation performance. In the present study, we have at our disposal

[2] When point source and convolutive mixing assumptions are verified.

the 2-channel stereo multitrack cover of each song to separate, and the basic principle is to use the cover source tracks to provide relevant initialization for the joint multichannel decomposition. Therefore, the NMF algorithms mentioned in Section 2 are applied on PPM within the following configuration. A first multichannel NMF decomposition is run on each stereo source of the cover multitrack (with random initialization). Thus, we obtain a modeled version of each cover source signal in the form of three matrices per source: $\mathbf{W}_j^{cover}$, $\mathbf{H}_j^{cover}$ and $\mathbf{A}_j^{cover} = \{a_{ij,f}^{cover}\}_{i\in[1,2],f}$. The results are ordered according to:

$$\mathbf{W}_{init}^{mix} = [\mathbf{W}_1^{cover} \ldots \mathbf{W}_J^{cover}] \tag{6}$$

$$\mathbf{H}_{init}^{mix} = \begin{bmatrix} \mathbf{H}_1^{cover} \\ \vdots \\ \mathbf{H}_J^{cover} \end{bmatrix} \tag{7}$$

$$\mathbf{A}_{init}^{mix} = [\mathbf{A}_1^{cover} \ldots \mathbf{A}_J^{cover}] \tag{8}$$

Then, (6), (7), and (8) are used as an initialization for a second convolutive stereo NMF decomposition run on the mix signal as in [11]. During this second phase, the spectral pattern vectors and time activation vectors learned from the cover source tracks are expected to evolve to match the ones corresponding to the signals used to produce the commercial mix, while the resulting mixing vectors are expected to fairly model the mix process.

3.2 Pre-processing: time alignment of the cover tracks

One main difference between two versions of the same music piece is often the temporal misalignment due to both tempo variation (global misalignment) and musical interpretation (local misalignments). In a general manner, time misalignment can corrupt the separation performances if the spectral pattern vectors used for initialization are not aligned with the spectral patterns of the sources within the mix. In the present framework, this problem is expected to be limited by the intrinsic automatic matching of temporal activity vectors within the multichannel NMF decomposition algorithm. However, the better the initial alignment, the better the initialization process and thus expected final result. Therefore, we limit this problem by resynchronizing the cover tracks with the mix signal, in the same spirit as the MIDI score-to-audio alignment of [5] or the Dynamic Time Warping (DTW) applied on synthesized signals in [8]. In the present study, this task is performed at quarter-note accuracy using the Beat Detective tool from the professional audio editing software Avid ProTools®. This step allows minimizing synchronization error down to less than a few TF frames, which is in most cases below the synchronization error limit of 200 ms observed in [5]. In-depth study of desynchronization on source separation is kept for future works.

3.3 Exploiting the temporal structure of source signals

In order to further improve the results, we follow a user-guided approach as in [12]. The coefficients of matrix $\mathbf{H}$ are zeroed when the source is not active in the mix, exploiting audio markers of silence zones in the cover source tracks. As there still may be some residual misalignment between the commercial song and the cover after the pre-processing, we relax these constraints to 3 frames before and after the active zone. When using the MU algorithm, the zeroed coefficients remain at zero. When using the EM algorithm, the update rules do not allow the coefficients of $\mathbf{H}$ to be strictly null, hence, we set these coefficients to the *eps* value in our Matlab® implementation. Observations confirm that these coefficients remain small throughout all the decomposition.

3.4 Summarizing the novelty of the proposed study

While our process is similar in spirit to several existing studies, e.g., [5,8,16], our contribution to the field involves:

- the use of cover multitrack signals instead of hummed or MIDI-synthesis source signals. Our cover signals are expected to provide a more faithful image of the original source signals in the PPM context.
- a stereo NMF framework instead of a mono one. The multichannel framework is expected to exploit spatial information in the demixing process (as far as the convolutive model is a fair approximation of the mixing process). It provides optimal spatial Wiener filters for the separation, as opposed to the {estimated magnitude + mix phase} resynthesis of [8] or the (monochannel) soft masks of [16].
- a synchronization pre-process relying on tempo and musical interpretation instead of, e.g., frame-wise DTW. This is completed with the exploitation of the sources temporal activity for the initialization of $\mathbf{H}$.

4. EXPERIMENTS

4.1 Data and experimental settings

Assessing the performances of source separation on true professionally-produced music data is challenging since the original multitrack signals are necessary to perform objective evaluation but they are seldom available. Therefore, we considered the following data and methodology. The proposed separation algorithm was applied on a series of 4 well-known pop-music songs for which we have the stereo commercial mix signal and two different stereo multitrack covers (see Table 2). The first multitrack cover C1 was provided by iKlax Media, and the second one C2 has been downloaded from the commercial website of another company. We present two testing configurations:

- **Setting 1**: This setting is used to derive objective measures (see below). C1 is considered as the "original multitrack", and used to make a stereo remix of the song which is used as the target mix to be separated. This remix has been processed by a qualified sound engineer with a 10-year background in music

Tracks duration	30 s
Number of channels	I=2
Sampling Rate	32 kHz
STFT frame size	2048
STFT overlap	50 %
Number of iterations	500
Number of NMF components	12 or 50

Table 1: Experimental settings

production, using Avid ProTools®.[3] C2 is considered as the cover version and is used to separate the target mix made with C1.

- **Setting 2**: The original commercial mix is separated using C1 as the cover. This setting is used for subjective evaluation in real-world configuration.

The covers are usually composed of 8 tracks which are quite faithful to the commercial song content as explained in the introduction. For simplicity we merged the tracks to obtain 4 to 6 source signals.[4] All signals are resampled at 32kHz, since source separation above 16kHz has very poor influence on the quality of separated signals and this enables to reduce computations. The experiments are carried out on 30s excerpts of each song.

It is difficult to evaluate the proposed method in reference to existing source separation methods since the cover information is very specific. However, in order to have a reference, we also applied the algorithm with a partial initialization: the spectral patterns $\mathbf{W}$ are here initialized with the cover spectral patterns, whereas the time activation vectors $\mathbf{H}$ are randomly initialized (vs. NMF initialization in the full cover-informed configuration). This enables to i) separate the contribution of cover temporal information, and ii) simulate a configuration where a dictionary of spectral bases is provided by an external database of instruments and voices. This was performed for both EM and MU algorithms. The main technical experimental parameters are summarized in Table 1.

4.2 Separation measures

To assess the separation performances in Setting 1, we computed the signal-to-distortion ratio (SDR), signal-to-interference ratio (SIR), signal-to-artifact ratio (SAR) and source image-to-spatial distortion ratio (ISR) defined in [18]. We also calculated the input SIR (SIR_{in}) defined as the ratio between the power of the considered source and the power of all the other sources in the mix to be separated. We consider this criterion because all sources do not contribute to the mix with the same power. Hence, a source with high SIR_{in} is easier to extract than a source with a low SIR_{in}, and SIR_{in} is used to characterize this difficulty.

[3] The source images are here the processed version of C1 just before final summation, hence we do not consider post-summation (non-linear) processing. The consideration of such processing in ISS, as in, e.g., [17], is part of our current efforts.

[4] The gathering was made according to coherent musical sense and panning, e.g., grouping two electric guitars with the same panning in a single track. It is necessary to have the same number of tracks between an original version and its cover. Furthermore, original and cover sources should share approximately the same spatial position (e.g., a cover version of a left panned instrument should not be right panned!)

Title	Tracks	Track names
I Will Survive	6	Bass, Brass, Drums, ElecGuitar, Strings, Vocal.
Pride and Joy	4	Bass, Drums, ElecGuitar, Vocal.
Rocket Man	6	Bass, Choirs, Drums, Others, Piano, Vocal.
Walk this Way	5	Bass, Drums, ElecGuitar1, ElecGuitar2, Vocal.

Table 2: Experimental dataset

Method	SDR	ISR	SIR	SAR
EM W_{init}	0,04	3,51	-1,96	4,82
EM Cover-based	2.45	6.58	4.00	5.38
EM Improvement	2,41	3,08	5,97	0,56
MU W_{init}	-0,98	3,58	-1,14	3,40
MU Cover-based	1.38	6.83	5.04	2.95
MU Improvement	2,36	3,24	6,18	-0,45

Table 3: Average source separation performance for 4 PPM mixtures of 4 to 6 sources (dB).

5. RESULTS

5.1 Objective evaluation

Let us first consider the results obtained with Setting 1. The results averaged across all sources and songs are provided in Table 3. The maximal average separation performance is obtained with the EM cover-informed algorithm with SDR = 2.45dB and SIR = 4.00dB. This corresponds to a source enhancement of $\text{SDR} - \text{SIR}_{\text{in}}$ = 10.05dB and $\text{SIR} - \text{SIR}_{\text{in}}$ = 11.60dB, with the average global SIR_{in} being equal to -7.60dB. These results show that the overall process leads to fairly good source reconstruction and rejection of competing sources. Figure 1a illustrates the separation performances in terms of the difference $\text{SDR} - \text{SIR}_{\text{in}}$ for the song "I will survive". The separation is very satisfying for tracks with sparse temporal activity such as Brass. The Strings track, for which the point source assumption is less relevant, obtains correct results, but tends to spread over other sources images such as Bass. Finally, when cover tracks musically differ from their original sources, the separation performance decreases. This is illustrated with the Electric Guitar (EGtr) and Bass tracks, which do not fully match the original interpretation.

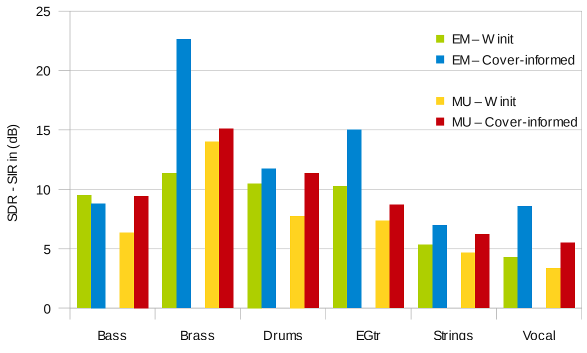
Let us now discuss the cover informed EM and MU methods in relation to the initialization of spectral bases only, referred to as W_{init}. The cover-based EM algorithm provides a notable average SDR improvement of 2.41dB over EM with W_{init} initialization, and a quite large improvement in terms of SIR (+5.97dB), hence a much better interference rejection. The cover-based MU algorithm also outperforms the MU W_{init} configuration to the same extent (e.g., +2.36dB SDR and +6.18dB SIR improvement). This reveals the ability of the method to exploit not only spectral but also temporal information provided by covers.

Note that both cover-based and W_{init} EM methods outperform the corresponding MU methods in terms of SDR. However, it is difficult to claim for clear-cut EM's better use of the inter-channel mutual information, since EM is slightly lower than MU for SIR (approx. 4dB vs. 5dB for the cover-informed method). In fact, the multichannel framework can take advantage of both spectral and spatial information for source extraction, but this depends on the source properties and mixing configuration. In the song "Walk this way", which detailed results are given in Figure 1b, all sources but the Electric Guitar 1 (Egtr1) are panned at the center of the stereo mixture. Thus, the $\text{SDR} - \text{SIR}_{\text{in}}$ obtained for Egtr1 reaches 20.32dB, as the algorithm relies strongly on spatial information to improve the separation. On the other hand, the estimated Vocal track in "I will survive" is well separated (+8.57dB $\text{SDR} - \text{SIR}_{\text{in}}$ for the cover-informed EM) despite being centered and coincident to other tracks such as Bass, Drums and Electric Guitar (EGtr). In this case, the proposed multichannel NMF framework seems to allow separation of spatially coincident sources with distinct spectral patterns. Depending on the song, some sources obtain better SDR results with the MU algorithm. For example, in "Walk this way", the $\text{SDR} - \text{SIR}_{\text{in}}$ for the Drums track increased from 6.59dB with the EM method to 9.74dB with the MU method. As pointed out in [11], the point source assumption certainly does not hold in this case. The different elements of the drums are distributed between both stereo channels and the source image cannot be modeled efficiently as a convolution of a single point source. By discarding a large part of the inter-channel information, the MU algorithm gives better results in this case. Preliminary tests using a monochannel NMF version of the entire algorithm (monochannel separation using monochannel initialization, as in, e.g., [8, 16]), even show slightly better results for the Drums track, confirming the irrelevancy of the point source convolutive model in this case.

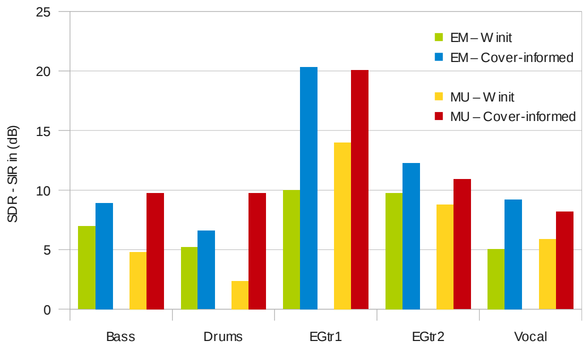
Finally, it can be mentioned that the number of NMF components per source K_j does not influence significantly the SDR and SIR values, although we perceive a slight improvement during subjective evaluation for $K_j = 50$.[5]

5.2 Discussion

Informal listening tests on the excerpts from Setting 2 confirm the previous results and show the potential of cover-informed methods for commercial mix signal separation.[6] Our method gives encouraging results on PPM when point



(a) I Will Survive



(b) Walk This Way

Figure 1: Separation results

source and convolutive assumptions are respected. For instance, the vocals are in most cases suitably separated, with only long reverberation interferences. As expected, the quality of the mix separation relies on the quality and faithfulness of the cover. A good point is that when original and cover interpretations are well matched, the separated signal sounds closer to the original than to the cover, revealing the ability of the adapted Wiener filters to well preserve the original information.

Comparative experiments with spectral basis initialization only (W_{init}) confirm the importance of the temporal information provided by covers, Although this has not been tested formally, the cover-to-mix alignment of Section 3.2 was shown by informal tests to also contribute to good separation performances.

6. CONCLUSION

The results obtained by plugging the cover-informed source separation concept in the framework of [11] show that both spectral and temporal information provided by cover signals can be exploited for source separation. This study indicates the interest (and necessity) of using high-quality covers. In this case, the separation process may better take into consideration the music production subtleties, compared to MIDI- or hummed-informed techniques.

Part of the results show the limitations of the convolutive mixing model in the case of PPM. This is the case for sources that cannot be modeled efficiently as a point source convolved on each channel with a linear filter, such as large instruments (e.g., drums and piano). Also, some

[5] Assessing the optimal number of components for each source is a challenging problem left for future work.

[6] Examples of original and separated signals are available at http://www.gipsa-lab.grenoble-inp.fr/∼laurent.girin/demo/ismir2012.html.

tracks such as vocals make use of reverberation times much higher than our analysis frame. As a result, most of the vocals reverberation is not properly separated. The present study and model also do not consider the possible nonlinear processes applied during the mixing process.

Therefore, further research directions include the use of more general models for both sources and spatial processing. For instance, we plan to test the full-rank spatial covariance model of [2], within the very recently proposed general framework of [13] which also enables more specific source modeling, still in the NMF framework (e.g., source-filter models). Within such general model, sources actually composed of several instruments (e.g., drums) may be spectrally and spatially decomposed more efficiently and thus better separated.

7. REFERENCES

[1] S. Dubnov. Optimal filtering of an instrument sound in a mixed recording using harmonic model and score alignment. In *Int. Computer Music Conf. (ICMC)*, Miami, FL, 2004.

[2] N. Q. K. Duong, E. Vincent, and R. Gribonval. Underdetermined reverberant audio source separation using a full-rank spatial covariance model. *IEEE Trans. on Audio, Speech, and Language Proc.*, 18(7):1830–1840, 2010.

[3] J. Engdegård, C. Falch, O. Hellmuth, J. Herre, J. Hilpert, A. Hölzer, J. Koppens, H. Mundt, H. Oh, H. Purnhagen, B. Resch, L. Terentiev, M. Valero, and L. Villemoes. MPEG spatial audio object coding—the ISO/MPEG standard for efficient coding of interactive audio scenes. In *129th Audio Engineering Society Convention*, San Francisco, CA, 2010.

[4] S. Ewert and M. Müller. Score-informed voice separation for piano recordings. In *Proc. of the 12th Int. Society for Music Information Retrieval Conf. (ISMIR)*, Miami, USA, 2011.

[5] S. Ewert and M. Müller. Using score-informed constraints for NMF-based source separation. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Proc. (ICASSP)*, Kyoto, Japan, 2012.

[6] C. Faller, A. Favrot, Y-W Jung, and H-O Oh. Enhancing stereo audio with remix capability. In *Proc. of the 129th Audio Engineering Society Convention*, 2010.

[7] C. Févotte, N. Bertin, and J.-L. Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence. With application to music analysis. *Neural Computation*, 21(3):793–830, 2009.

[8] J. Ganseman, P. Scheunders, G. Mysore, and J. Abel. Source separation by score synthesis. In *Proc. of the Int. Computer Music Conf. (ICMC)*, New-York, 2010.

[9] S. Gorlow and S. Marchand. Informed source separation: Underdetermined source signal recovery from an instantaneous stereo mixture. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, 2011.

[10] A. Liutkus, J. Pinel, R. Badeau, L. Girin, and G. Richard. Informed source separation through spectrogram coding and data embedding. *Signal Processing*, 92(8):1937–1949, 2012.

[11] A. Ozerov and C. Févotte. Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation. *IEEE Trans. on Audio, Speech, and Language Proc.*, 18(3):550–563, 2010.

[12] A. Ozerov, C. Févotte, R. Blouet, and J.-L. Durrieu. Multichannel nonnegative tensor factorization with structured constraints for user-guided audio source separation. In *Proc. of the Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP)*, Prague, Czech Republic, 2011.

[13] A. Ozerov, E. Vincent, and F. Bimbot. A general flexible framework for the handling of prior information in audio source separation. *IEEE Trans. on Audio, Speech and Language Proc.*, 20(4):1118–1133, 2012.

[14] M. Parvaix and L. Girin. Informed source separation of linear instantaneous under-determined audio mixtures by source index embedding. *IEEE Trans. on Audio, Speech, and Language Proc.*, 19(6):1721–1733, 2011.

[15] M. Parvaix, L. Girin, and J.-M. Brossier. A watermarking-based method for informed source separation of audio signals with a single sensor. *IEEE Trans. on Audio, Speech, and Language Proc.*, 18(6):1464–1475, 2010.

[16] P. Smaragdis and G. Mysore. Separation by "humming": User-guided sound extraction from monophonic mixtures. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, 2009.

[17] N. Sturmel, A. Liutkus, J. Pinel, L. Girin, S. Marchand, G. Richard, R. Badeau, and L. Daudet. Linear mixing models for active listening of music productions in realistic studio condition. In *Proc. of the 132th Audio Engineering Society Conv.*, Budapest, Hungary, 2012.

[18] E. Vincent, R. Gribonval, and C. Févotte. Performance measurement in blind audio source separation. *IEEE Trans. on Audio, Speech, and Language Proc.*, 14(4):1462–1469, 2006.

[19] M. Vinyes, J. Bonada, and A. Loscos. Demixing commercial music productions via human-assisted time-frequency masking. In *Proc. of the 120th Audio Engineering Society Convention*, 2006.

[20] J. Woodruff, B. Pardo, and R. B. Dannenberg. Remixing stereo music with score-informed source separation. In *Int. Society for Music Information Retrieval Conference (ISMIR)*, Victoria, Canada, 2006.

BAYESIAN NONNEGATIVE HARMONIC-TEMPORAL FACTORIZATION AND ITS APPLICATION TO MULTIPITCH ANALYSIS

Daichi Sakaue Takuma Otsuka Katsutoshi Itoyama Hiroshi G. Okuno
Graduate School of Informatics, Kyoto University
{dsakaue,ohtsuka,itoyama,okuno}@kuis.kyoto-u.ac.jp

ABSTRACT

Since important musical features are mutually dependent, their relations should be analyzed simultaneously. Their Bayesian analysis is particularly important to reveal their statistical relation. As the first step for a unified music content analyzer, we focus on the harmonic and temporal structures of the wavelet spectrogram obtained from harmonic sounds. In this paper, we present a new Bayesian multipitch analyzer, called Bayesian nonnegative harmonic-temporal factorization (BNHTF). BNHTF models the harmonic and temporal structures separately based on Gaussian mixture model. The input signal is assumed to contain a finite number of harmonic sounds. Each harmonic sound is assumed to emit a large number of sound quanta over the time-log-frequency domain. The observation probability is expressed as the product of two Gaussian mixtures. The number of quanta is calculated in the ϵ-neighborhood of each grid point on the spectrogram. BNHTF integrates latent harmonic allocation (LHA) and nonnegative matrix factorization (NMF) to estimate both the observation probability and the number of quanta. The model is optimized by newly designed deterministic procedures with several approximations for the variational Bayesian inference. Results of experiments on multipitch estimation with 40 musical pieces showed that BNHTF outperforms the conventional method by 0.018 in terms of F-measure on average.

1. INTRODUCTION

Multipitch estimation [5, 7, 10, 19] is one of the most fundamental techniques of music information retrieval (MIR) because the temporal pattern of pitch strongly expresses the content of musical pieces, especially in Western music. It is useful for a wide range of applications, including content-based music search [2], musical instrument identification [9], and chord recognition [15].

One promising technique in multipitch analysis is to assume a probabilistic generative model of musical signals and then perform pattern matching between the model and recorded signals using EM algorithm. Variational Bayesian methods are particularly valuable because they can flexibly model the probabilistic relation between the occurrence pattern of pitch and many important musical aspects, including harmonic structure, musical instrument, musical structure [11], chord, onset, and emotion [8]. Our goal is to formulate a music analyzer that estimates the relation between all such latent variables. At present, latent harmonic allocation (LHA) is the most suitable candidate for further extensions.

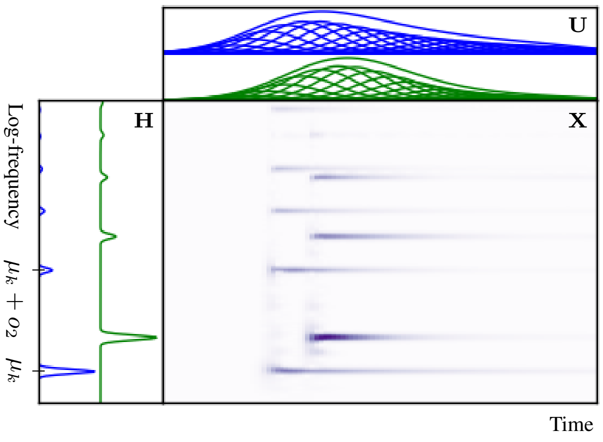


Figure 1. Illustration of Bayesian nonnegative harmonic-temporal factorization. Basis and activation matrices are explicitly modeled using Gaussian mixtures.

The most important features in an observed wavelet spectrogram are the harmonic and temporal structures. These structures are mutually dependent and should therefore be analyzed simultaneously. Conventionally, LHA defines the volume of a sound at a time frame as the relative coefficient to the total volume of that time frame. As a result, there is no explicit variable that describes the actual volume of each sound. This makes it difficult to enhance the model so that it considers the temporal envelope of the sounds.

In this paper, we present a new method that explicitly models the volume of each harmonic sound using a time series of Poisson distributions and its harmonic and temporal structure using mixtures of Gaussians. This is illustrated in Figure 1. Here, the observed spectrogram is interpreted as a histogram of a large number of statistically independent particles. This interpretation corresponds to the integration of LHA and Bayesian NMF [3], because both of them assumes the spectrogram as a histogram. In our model, the number of observations at each time-log-frequency point is calculated in an ϵ-neighborhood of the point. To do this, we integrate the probabilistic density functions (pdfs) of harmonic structures around the f-th frequency bin, with a quite small amount of a band of width ϵ_{F}. For the temporal structure, we introduce a similar assumption using ϵ_{T}. The objective is to fit the generative model of BNHTF to the standard formulation of NMF. The variational posterior distributions are approximately conjugate, when we take the limit $\epsilon_{\mathrm{F}}, \epsilon_{\mathrm{T}} \to 0$. The formulation strategy is similar to those formerly obtained by Ochiai [12, 13]. Ours seem to deliver a more concrete interpretation of the generative process, rather than their construction which depends on the direct use of Dirac delta function.

Our method can also be viewed as a Bayesian NMF-based method with adaptive harmonic basis. This method should be quite valuable for many researchers, because such a model has been searched for long years [1, 14, 18]. As the proposed method is a variation of NMF, our method can easily be extended further by using recent improvements of NMF.

2. CONVENTIONAL METHODS

2.1 Harmonic Clustering

The spectral envelope of harmonic sound has several peaks at its fundamental frequency and the overtone frequencies. This structure, known as harmonic structure, can be approximated by using a mixture of Gaussians. In this paper, we call this approach *harmonic clustering*. Harmonic clustering methods represent the wavelet spectrum of each harmonic sound as a probabilistic density function described by a mixture of Gaussians. PreFEst [5], harmonic temporal clustering (HTC) [7], and LHA [19] are notable examples of harmonic clustering.

The mathematical representation is as follows. Let x be the log-frequency, μ_k be the logarithm of the fundamental frequency of the k-th harmonic sound, and λ_k be the precision of the Gaussian components. Moreover, let M be the number of harmonic partials considered in the model. The relative weight of each harmonic partial is indicated using $\eta_k = [\eta_{k1}, \cdots, \eta_{kM}]$. Following this, the k-th harmonic sound is represented as:

$$p_k(x|\eta_k, \mu_k, \lambda_k) = \sum_{m=1}^{M} \eta_{km} \mathcal{N}(x|\mu_k + o_m, \lambda_k^{-1}), \quad (1)$$

where $\mathcal{N}$ denotes normal distribution and o_m denotes the relative position of the m-th overtone component on the log-frequency axis. To retain the versatility of the model, in most cases, o_m is set so that the model represents completely harmonic sounds. The relationship between the log and linear frequency scales is defined as:

$$f_{\log} = 1200(\log_2 f - \log_2 440 + 4.75). \quad (2)$$

2.2 Latent Harmonic Allocation

Latent harmonic allocation (LHA) is a variational Bayesian method of harmonic clustering that represents each time frame spectrum of the observed spectrogram using a mixture of harmonic sound models. The observed spectrum is interpreted as a histogram of numerous sound quanta that are generated by the observation model. The t-th time frame spectrum is represented as a linear combination of K harmonic sounds with mixing coefficients $\pi_t = [\pi_{t1} \cdots \pi_{tK}]$. The emission probability of a sound quantum is described as:

$$p_t(x|\pi, \eta, \mu, \lambda) = \sum_{k=1}^{K} \pi_{tk} p_k(x|\eta_k, \mu_k, \lambda_k). \quad (3)$$

Let x_{tf} be the value of the wavelet spectrogram at the t-th time frame and the f-th frequency bin. The overall observation probability of the t-th time frame $X_t = [x_{t1}, \cdots, x_{tF}]$ is described as:

$$p_t(X_t|\pi, \eta, \mu, \lambda) = \prod_{f=1}^{F} \left(\sum_{km} \pi_{tk} \eta_{km} \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1}) \right)^{x_{tf}}, \quad (4)$$

where x_f denotes the log-frequency of the f-th frequency bin.

The volume of the k-th sound at the t-th time frame is implicitly determined by π_{tk}, which describes the relative weight of the k-th sound in the time frame. Because the total volume of each time frame differs between the frames, it can be difficult to discuss the temporal envelope of π_{tk}. To solve this problem, we explicitly model the volume using a time series of Poisson distributions, which is similar to the construction of Bayesian NMF.

2.3 Bayesian Nonnegative Matrix Factorization

Nonnegative matrix factorization (NMF) is a promising information retrieval method that factorizes the observed matrix of size $N \times M$ into two matrices of size $N \times K$ and $K \times M$. Naturally, we set $K \ll N$ and M. In music analysis, NMF is often applied to an STFT or a wavelet spectrogram. The two matrices are learned by minimizing a cost function, $D(\mathbf{X}||\mathbf{UH})$, where $\mathbf{X} = \{x_{tf}\}$ denotes the observed spectrogram and $\mathbf{U} = \{u_t^k\}$ and $\mathbf{H} = \{h_f^k\}$ denotes the factorized matrices. The cost function is usually customized for specific objectives [1, 4, 14, 17]. Following this, x_{tf} is approximated as:

$$x_{tf} \approx \sum_{k=1}^{K} h_f^k u_t^k. \quad (5)$$

As in LHA, the index k stands for the specific spectral pattern. This pattern is described by the vector $[h_1^k, \cdots, h_F^k]$. The volume of the k-th basis in the t-th time frame is denoted by u_t^k. $\mathbf{H}$ is often called the basis matrix, and $\mathbf{U}$ is called the activation matrix.

Cemgil [3] proposed a full Bayesian inference of NMF, in which the joint posterior probability of latent variables

$p(S, \mathbf{H}, \mathbf{U}|\mathbf{X})$ is estimated. Here, S denotes the set of K separated spectrograms. The likelihoods and prior distributions are written as:

$$p(x_{tf}|s_{tf}^{[k]}) = \delta(x_{tf} - \sum_{k=1}^{K} s_{tf}^{k}), \tag{6}$$

$$p(s_{tf}^{k}|h_f^k, u_t^k) = \mathcal{P}(s_{tf}^k|h_f^k u_t^k), \tag{7}$$

$$p(h_f^k) = \mathrm{Gam}(h_f^k|a_0, b_0), \tag{8}$$

$$p(u_t^k) = \mathrm{Gam}(u_t^k|a_0, b_0), \tag{9}$$

where s_{tf}^k denotes the observation of k-th sound. Hereafter, a square bracket indicates a set over the index variable, δ denotes delta function, $\mathcal{P}$ denotes Poisson distribution, and Gam denotes Gamma distribution. The prior distributions of h_f^k and u_t^k are the conjugate priors, where a_0, b_0 are the hyperparameters of the distributions. A variational EM algorithm is obtained based on a mean-field approximation, $p(S, \mathbf{H}, \mathbf{U}|\mathbf{X}) \approx q(S)q(\mathbf{H})q(\mathbf{U})$.

The main drawback of NMF is its inability to model spectral and temporal continuity using Gaussian mixtures. Though one can say that template-based approach can model these continuities, in that case, we cannot update the envelope of basis and activation matrices adaptively. Many methods have been proposed to solve this problem [1, 14, 16, 18], but these methods are difficult to extend further because they do not estimate Gaussian mixture densities based on variational Bayes.

3. SIGNAL MODEL

In this section, we describe how to integrate the two promising methods: LHA and NMF. At first, we describe our idea by introducing the spectral continuity. In these methods, the wavelet spectrogram is interpreted as a histogram of sound quanta. Here, the volume of each sound is interpreted as the number of quanta. The number is determined from a Poisson distribution for each time frame. Next, the distribution of sound quanta is determined from a mixture of Gaussians. In the next subsection, we describe a straightforward observation model. This model is not suitable for the estimation, so we introduce several approximations afterward to formulate a VB-EM algorithm.

3.1 Generative Model

The volume of the k-th sound at the t-th time frame is drawn from a Poisson distribution $\mathcal{P}(S_t^k|u_t^k)$, similar to NMF. Next, we draw the number of observations of the m-th harmonic partial of the k-th sound following a multinomial distribution. The relative weight of each component is determined by η_{km}.

$$p(S_t^k) = \mathcal{P}(S_t^k|u_t^k) \tag{10}$$

$$p(S_t^{k[m]}) = \mathcal{M}(S_t^{k[m]}|S_t^k, \eta_{k[m]}) \tag{11}$$

Here, $\mathcal{M}$ denotes multinomial distribution.

Next, we draw a set of observed particles: $X_t^{km} = [x_{t1}^{km}, \cdots, x_{tS_t^{km}}^{km}]$. The value of each particle follows the Gaussian distribution of the corresponding the m-th harmonic partial. The likelihood of the observation is written as:

$$p(x_{tn}^{km}|\mu_k, \lambda_k) = \mathcal{N}(x_{tn}^{km}|\mu_k + o_m, \lambda_k^{-1}), \tag{12}$$

$$p(X_t^{km}|S_t^{km}, \mu_k, \lambda_k) = \prod_{n=1}^{S_t^{km}} \mathcal{N}(x_{tn}^{km}|\mu_k + o_m, \lambda_k^{-1}). \tag{13}$$

To generate the spectrogram of each harmonic partial, we assume the number of particles observed in spectrograms as a histogram of particles that have a value error in the range of $\epsilon_{\mathrm{F}}/2$. This corresponds with our transformation of the continuous probabilistic density functions into discrete probabilistic mass functions.

$$x_{tf}^{km} = \#\{n|x_f - \epsilon_{\mathrm{F}}/2 \le x_{tn}^{km} \le x_f + \epsilon_{\mathrm{F}}/2\} \tag{14}$$

$$x_{t\neg}^{km} = S_t^{km} - \sum_{f=1}^{F} x_{tf}^{km} \tag{15}$$

$$\hat{r}_f^{km} = \int_{x_f - \epsilon_{\mathrm{F}}/2}^{x_f + \epsilon_{\mathrm{F}}/2} \mathcal{N}(x|\mu_k + o_m, \lambda_k^{-1}) dx \tag{16}$$

$$p(x_{t[f]}^{km}, x_{t\neg}^{km}) = \mathcal{M}(x_{t[f]}^{km}, x_{t\neg}^{km}|S_t^{km}, \hat{r}_{[f]}^{km}, \hat{r}_{\neg}^{km}) \tag{17}$$

Here, $\#$ denotes the number of elements in the set, x_{tf}^{km} denotes the number of particles allocated the f-th frequency bin, $x_{t\neg}^{km}$ denotes the number of particles which are not allocated any frequency bin, and $\hat{r}_f^{km}$ denotes the relative weight of each frequency bin. Further, $\hat{r}_f^{km}$ is approximated as:

$$\hat{r}_f^{km} \approx \epsilon_{\mathrm{F}} \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1}). \tag{18}$$

We denote the right hand of the equation r_f^{km}. Finally, the observed spectrogram is obtained as a summation of the all harmonic components.

$$x_{tf} = \sum_{km} x_{tf}^{km} \tag{19}$$

3.2 Approximations

For the above formulations are not appropriate for Bayesian estimation, we introduce the following approximation. The main objective is to marginalize the volume variables S_t^k and S_t^{km}. To do this, we inspect the following characteristics of Poisson distribution.

Poisson distribution gives the probability of n event observations in a unit time when the average occurrence interval is λ^{-1}. Next, we consider to distribute the observations into K classes, following the distribution: $\mathcal{M}(n_{[k]}|n, p_{[k]})\mathcal{P}(n|\lambda)$. The marginal probability of each class of the multinomial distribution follows a binomial distribution, so $p(n_k|n, p_k) = \mathrm{Bin}(n_k|n, p_k)$. Further, we assume that $p(n_k|p_k, \lambda) = \mathcal{P}(n_k|p_k\lambda)$ because the event of the k-th class occurs in an average time interval of $(p_k\lambda)^{-1}$. In the following section, we formulate a VB-EM algorithm based on this observation model.

4. BAYESIAN NONNEGATIVE HARMONIC FACTORIZATION

In this section, we describe the formulation of our model and the update procedures of Bayesian nonnegative harmonic factorization (BNHF). The probabilistic mass function of the intermediate spectrogram of the m-th harmonic partial and the k-th harmonic sound is first described. The spectrogram is generated following the activation u_t^k, which is the relative weight of each harmonic partial η_{km}. The prior distributions are selected to imitate LHA and NMF. This is formulated as follows.

$$\begin{aligned} &p(s_{tf}^{km}|u_t^k, \eta_k, \mu_k, \lambda_k) \\ &\approx \mathcal{P}(s_{tf}^{km}|\epsilon_{\mathrm{F}} u_t^k \eta_{km} \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1})) \end{aligned} \tag{20}$$

$$p(u_t^k) = \mathrm{Gam}(u_t^k|a_0, b_0) \tag{21}$$

$$p(\eta_k) = \mathrm{Dir}(\eta_k|\alpha_m^0) \propto \prod_{m=1}^{M} \eta_{km}^{\alpha_m^0 - 1} \tag{22}$$

$$p(\mu_k, \lambda_k) = \mathcal{N}(\mu_k|m_0, (\beta_0\lambda_k)^{-1})\mathcal{W}(\lambda_k|w_0, \nu_0) \tag{23}$$

Here, $\mathcal{W}$ denotes Wishart distribution and a_0, b_0, α_m^0, m_0, β_0, w_0, and ν_0 are the hyperparameters. The prior distributions are not conjugate, and thus the analytic variational Bayesian inference of the posterior distributions is intractable. Instead, we will follow a limit that $\epsilon_{\mathrm{F}} \to 0$. Under this condition, the posterior distributions are written in an approximately conjugate form. First, we assume the following factorization.

$$q(S, u, \eta, \mu, \lambda) = q(S) \prod_{tk} q(u_t^k) \prod_{k=1}^{K} \{q(\eta_k) q(\mu_k, \lambda_k)\} \tag{24}$$

This is known as mean-field approximation.

4.1 VB-E Step

During the VB-E step, we calculate the temporal estimation of separated source spectrogram s_{tf}^{km}.

$$\begin{aligned} \ln q^*(s_{tf}^{[km]}) &= \ln(X|S) + \mathbb{E}[p(S|u, \eta, \mu, \lambda)] \\ &= \ln \delta(x_{tf} - \sum_{km} s_{tf}^{km}) + \mathbb{E}[\ln u_t^k + \ln \eta_{km} \\ &\quad + \ln \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1})] \end{aligned} \tag{25}$$

Hereafter, all constant variables that do not affect the inference are omitted. The optimal posterior distribution is a multinomial distribution.

$$q(s_{tf}^{[km]}) = \mathcal{M}(s_{tf}^{[km]}|x_{tf}, \gamma_{tf}^{[km]}) \tag{26}$$

$$\ln \tilde{\gamma}_{tf}^{km} = \mathbb{E}[\ln u_t^k + \ln \eta_{km} + \ln \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1})] \tag{27}$$

$$\gamma_{tf}^{km} = \frac{\tilde{\gamma}_{tf}^{km}}{\sum_{k'm'} \tilde{\gamma}_{tf}^{k'm'}} \tag{28}$$

Here, $\mathcal{M}$ denotes multinomial distribution.

4.2 VB-M Step

During the VB-M step, we update the posterior distributions of u_t^k, η_{km}, μ_k, and λ_k. For example, we describe the Bayesian estimation of u_t^k in detail. The logarithm of the optimal posterior distribution is written as:

$$\begin{aligned} \ln q^*(u_t^k) &= \mathbb{E}_S[\ln p(S|u, \eta, \mu, \lambda)] + \ln p(u_t^k) \\ &= \sum_{fm} \mathbb{E}[s_{tf}^{km}] \ln u_t^k + (a_0 - 1) \ln u_t^k \\ &\quad - \sum_{fm} \epsilon_{\mathrm{F}} u_t^k \eta_{km} \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1}). \end{aligned} \tag{29}$$

Taking the limit $\epsilon_{\mathrm{F}} \to 0$, we obtain the following update:

$$q^*(u_t^k) \approx \mathrm{Gam}(u_t^k|a_t^k, b_0), \text{where} \tag{30}$$

$$a_t^k = a_0 + \sum_{fm} \mathbb{E}[s_{tf}^{km}]. \tag{31}$$

The same is true for η_{km}, μ_k, and λ_k: the optimal posterior distributions have the conjugate form when we take the limit $\epsilon_{\mathrm{F}} \to 0$. The approximated posterior distributions are written as:

$$q^*(\eta_k) \approx \mathrm{Dir}(\eta_k|\alpha_k), \tag{32}$$

$$q^*(\mu_k, \lambda_k) \approx \mathcal{N}(\mu_k|m_k, (\beta_k\lambda_k)^{-1})\mathcal{W}(\lambda_k|w_k, \nu_k), \tag{33}$$

where the posterior hyperparameters are written as:

$$\alpha_{km} = \alpha_m^0 + \sum_{tf} \mathbb{E}[s_{tf}^{km}], \tag{34}$$

$$m_k = \frac{m_0\beta_0 + \sum_{tfm} \mathbb{E}[s_{tf}^{km}](x_f - o_m)}{\beta_0 + \sum_{tfm} \mathbb{E}[s_{tf}^{km}]}, \tag{35}$$

$$\beta_k = \beta_0 + \sum_{tfm} \mathbb{E}[s_{tf}^{km}], \tag{36}$$

$$w_k^{-1} = w_0^{-1} + \beta_0 m_0^2 + \sum_{tfm} \mathbb{E}[s_{tf}^{km}](x_f - o_m)^2 - \beta_k m_k^2, \tag{37}$$

$$\nu_k = \nu_0 + \sum_{tf} \mathbb{E}[s_{tf}^{km}]. \tag{38}$$

The update equation of BNHF is quite similar to that of LHA, and these two methods had exactly the same result in our experiment. The difference is that our model explicitly models the volume of each sound, which makes it easier to consider the temporal continuity. This is described in more detail in the next section. We can also formulate the Gibbs sampler by using a similar approximation in which the space complexity is of the order $O(TFKM)$, instead of the $O(TNKM)$ for LHA. The derivations are omitted due to space restrictions.

5. BAYESIAN NONNEGATIVE HARMONIC-TEMPORAL FACTORIZATION

Here, we describe how to introduce temporal continuity to BNHF. The temporal structure is modeled using a mixture of Gaussians arranged at regular intervals. The intensity of

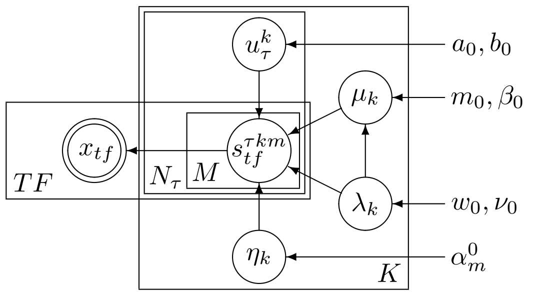


Figure 2. Graphical model of proposed method. Single solid lines indicate latent variables and double solid lines indicate observed variables.

each Gaussian component is decided using a Gamma distribution. Let T be the interval, τ be the index of Gaussian components, λ_{T} be the precision of each Gaussian component, and u_τ^k be the intensity of τ-th Gaussian component of k-th sound. The joint distribution can be written as $p(X, S, u, \eta, \mu, \lambda)$. A graphical model of the method is shown in Figure 2.

The conditional probability of $s_{tf}^{\tau km}$ is:

$$p(s_{tf}^{\tau km}|u, \eta, \mu, \tau) = \mathcal{P}(s_{tf}^{\tau km}|\epsilon_{\mathrm{T}} u_\tau^k \mathcal{N}(y_t|\tau T, \lambda_{\mathrm{T}}^{-1}) \\ \times \epsilon_{\mathrm{F}} \eta_{km} \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1})), \quad (39)$$

where y_t is the temporal position of the t-th time frame. The corresponding optimal posterior distribution is:

$$q(s_{tf}^{[\tau km]}) = \mathcal{M}(s_{tf}^{[\tau km]}|x_{tf}, \gamma_{tf}^{[\tau km]}), \quad (40)$$

$$\gamma_{tf}^{\tau km} \propto \exp(\mathbb{E}[\ln u_\tau^k + \ln \eta_{km} + \\ \ln \mathcal{N}(x_f|\mu_k + o_m, \lambda_k^{-1})] + \ln \mathcal{N}(t|\tau T, \lambda_{\mathrm{T}}^{-1})) \quad (41)$$

Further, the optimal posterior distribution of u_τ^k is approximated as:

$$q(u_\tau^k) \approx \mathrm{Gam}(u_\tau^k|a_\tau^k, b_0), \text{where} \quad (42)$$

$$a_\tau^k = a_0 + \sum_{tfm} \mathbb{E}[s_{tf}^{\tau km}]. \quad (43)$$

6. EVALUATION

In this section, we compare the performance of three multi-pitch estimation methods: LHA, NHF, and NHTF. We then discuss their performance in detail.

6.1 Estimation Target

For the experiment, we used 40 musical pieces from the RWC Music Database [6]. These included five piano solo pieces (RM-J001 to RM-J005), five guitar solo pieces (RM-J006 to RM-J010), ten jazz duo pieces (RM-J011 to RM-J020), ten jazz pieces played with three or more players (RM-J021 to RM-J030), and ten classical chamber pieces (RM-C012 to RM-C021). All the pieces were recorded from MIDI files using a MIDI synthesizer (Yamaha MOTIF-XS.) The drum tracks were muted, and the number of players was counted without including the drum player. The recorded signals were truncated to the first 32 seconds to reduce the large computational time needed for the experiment. They were transformed into wavelet spectrograms using Gabor wavelets with a time resolution of 16 [msec], frequency bins from 30 to 3000 [Hz], and a frequency resolution of 12 [cents]. The ground truths were constructed using the reference MIDI files.

Table 1. Calculated F-measures.

	Non-infomative			Infomative		
Music Type	LHA	BNHF	BNHTF	LHA	BNHF	BNHTF
Piano Solo	0.558	0.558	0.590	0.584	0.584	**0.590**
Guitar Solo	0.684	0.684	0.726	0.728	0.728	**0.740**
Jazz (Duo)	0.524	0.524	0.545	0.552	0.552	**0.556**
Jazz (Trio∼)	0.523	0.523	**0.548**	0.536	0.536	0.541
Chamber	0.481	0.481	0.508	0.503	0.503	**0.512**

6.2 Experimental Settings

Here, we describe the experimental settings. For the estimation, two prior distribution settings were evaluated. In the first one, all priors were set to be non-informative. That is, $a_0, b_0, \alpha_m^0, \beta_0, w_0$, and ν_0 were set to unity and m_0 was set to zero. In the other one, the prior distribution of the harmonic structure was set appropriately. That is, $\alpha_m^0 = 0.6547Nm^{-2}$, where N is the number of total observations. The other hyperparameters were set to be non-informative. The setting of α_m^0 was the same setting as HTC.

As an initialization, the relative weight of the source model in the t-th frame was set to the sum of the amplitudes of the nearest frequency bins of its overtones. The initial overtone weights were set to decay exponentially.

Model orders K and M were set to 73 and 6, respectively, where the number of overtones M is equal to HTC [7]. The EM algorithm was truncated at 200 iterations. The number of iterations was determined experimentally on the basis of estimation accuracy saturation.

After the iterations, the estimated pitches were extracted from the posterior hyperparameters. Let r be the threshold. The effective observation count of the k-th basis in the t-th time frame, N_{tk}, satisfies $N_{tk} \geq r \max_{tk} N_{tk}$, is considered to be sounding. The threshold was optimized experimentally for each piece and the method used to evaluate the potential performance of each model. The model frequencies were allocated the nearest note number. The estimated result and the ground truth were transformed into $T \times 128$ binary maps, and the F-measure was then calculated. Let N be the number of entries in the estimated map, C be the number of entries in the truth map, and R be the number of correct entries in the estimated map. The resultant F-measure is calculated as $F = 2R/(N + C)$. The larger the value of the F-measure, the better the performance.

6.3 Results

The results are shown in Table 1. Our method outperforms the conventional method for all the music type and both prior settings. The highest performance was attained with

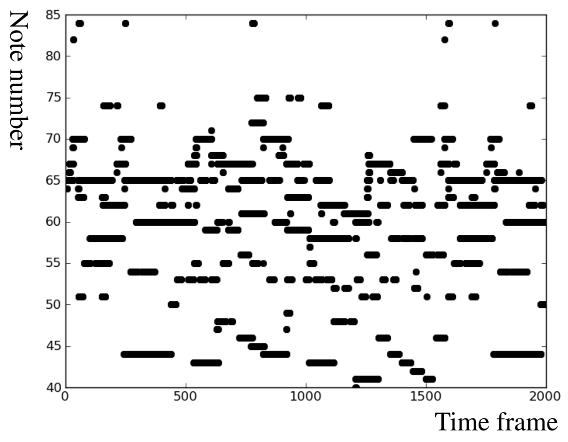


Figure 3. Example of estimated result of the proposed method with musical piece *RM-J007*.

appropriate prior for four of the five groups of musical pieces and with non-informative for the remaining group. This indicates the importance of the joint estimation of the harmonic and temporal structures. Figure 3 illustrates an example of the estimated result of the proposed model.

7. RELATED WORKS

Our method have a strong relation to the model proposed by Ochiai *et., al* [12,13]. Ours and theirs are notably different in three points. Firstly, they do not explicitly model the spectral continuity. Secondly, they use Dirac delta function instead of the ϵ-neighborhood to achieve the estimation of Gaussian mixture distribution. By following this strategy, the estimation is in conjugate form, but the interpretation of the model becomes difficult. Thirdly, they model the temporal structure based on the joint estimation of probabilistic context-free grammar and Gaussian mixture distribution.

8. CONCLUSION

We presented a new multipitch analyzer based on variational Bayes that explicitly models the harmonic and temporal structures separately based on Gaussian mixture model. Several priors were set to be not conjugate, in order to integrate NMF and LHA. The variational posterior distributions become conjugate under several approximations. Our method can be viewed as a Bayesian NMF method with adaptive harmonic basis. Evaluation results showed that the proposed method outperform the conventional multipitch analyzer, latent harmonic allocation (LHA). In the future, we will propose a more precise modeling using recent improvements to NMF, including the source-filter model [17] and nonparametric models [11]. This research is partially supported by KAKENHI (S) No. 24220006.

9. REFERENCES

[1] A. Bertin et al. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcripntion. *IEEE Trans. on ASLP*, 18(3):538–549, 2010.

[2] B. A. Casey et al. Content-based music information retrieval: Current directions and future challenges. *Proc. of the IEEE*, 96(4):668–696, 2008.

[3] A. T. Cemgil. Bayesian inference for nonnegative matrix factorization models. *Technical Report CUED/F-INFENG/TR.609*, 2008.

[4] D. FitzGerald et al. On the use of the beta divergence for musical source separation. In *Proc. ISSC*, pages 1–6, 2010.

[5] M. Goto. A real-time music-scene-analysis system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, 43(4):311–329, September 2004.

[6] M. Goto et al. RWC music database: Popular, classical, and jazz music databases. In *Proc. ISMIR*, pages 287–288, 2002.

[7] H. Kameoka et al. A multipitch analyzer based on harmonic temporal structured clustering. *IEEE Trans. on ASLP*, 15(3):982–994, 2007.

[8] Y. E. Kim et al. Music emotion recognition: A state of the art review. In *Proc. ISMIR*, pages 255–266, 2010.

[9] T. Kitahara et al. Instrument identification in polyphonic music: feature weighting to minimize influence of sound overlaps. *EURASIP J. Appl. Signal Process.*, 2007:1–15, 2007.

[10] A. Klapuri. Multipitch analysis of polyphonic music and speech signals using an auditory model. *IEEE Trans. on ASLP*, 16(2):255–266, 2008.

[11] M. Nakano et al. Nonparametric Bayesian music parser. In *Proc. ICASSP*, pages 461–464, 2012.

[12] K. Ochiai et al. Explicit beat structure modeling for non-negative matrix factorization-based multipitch analysis. In *Proc. ICASSP*, pages 133–136, 2012.

[13] K. Ochiai et al. Hierarchical Bayesian modeling of the generating process of music signals for automatic transcription. In *ASJ Spring Meeting*, 2012 (in Japanese).

[14] S. A. Raczyński et al. Multipitch analysis with harmonic nonnegative matrix approximation. In *Proc. ISMIR*, pages 381–386, 2007.

[15] Y. Ueda et al. HMM-based approach for automatic chord detection using refined acoustic features. In *Proc. ICASSP*, pages 5518–5521, 2010.

[16] E. Vincent et al. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Trans. on ASLP*, 18(3):528–537, 2010.

[17] T. Virtanen et al. Analysis of polyphonic audio using source-filter model and nonnegative matrix factorization. In *Advances in Models for Acoustic Processing*, 2006.

[18] N. Yasuraoka et al. I-Divergence-based dereverberation method with auxiliary function approach. In *Proc. ICASSP*, pages 369–372, 2011.

[19] K. Yoshii and M. Goto. A nonparametric Bayesian multipitch analyzer based on infinite latent harmonic allocation. *IEEE Trans. on ASLP*, 20(3):717–730, 2012.

A SCAPE PLOT REPRESENTATION FOR VISUALIZING REPETITIVE STRUCTURES OF MUSIC RECORDINGS

Meinard Müller
Bonn University and MPI Informatik
meinard@mpi-inf.mpg.de

Nanzhu Jiang
Saarland University and MPI Informatik
njiang@mpi-inf.mpg.de

ABSTRACT

The development of automated methods for revealing the repetitive structure of a given music recording is of central importance in music information retrieval. In this paper, we present a novel scape plot representation that allows for visualizing repetitive structures of the entire music recording in a hierarchical, compact, and intuitive way. In a scape plot, each point corresponds to an audio segment identified by its center and length. As our main contribution, we assign to each point a color value so that two segment properties become apparent. Firstly, we use the lightness component of the color to indicate the repetitiveness of the encoded segment, where we revert to a recently introduced fitness measure. Secondly, we use the hue component of the color to reveal the relations between different segments. To this end, we introduce a novel grouping procedure that automatically maps related segments to similar hue values. By discussing a number of popular and classical music examples, we illustrate the potential and visual appeal of our representation and also indicate limitations.

1. INTRODUCTION

The musical form describes a piece of music in terms of musical parts such as intro, chorus, and verse of a popular song or the first and second theme of a classical work. Such musical parts are typically repeated several times throughout the piece and evoke in the listener the feeling of familiarity. One major goal of audio structure analysis is to automatically derive the musical form directly from a given music recording. To this end, most procedures divide the music recording into repeating temporal segments and then group these segments according to musically meaningful categories [13].

Finding the repetitive structure of a music recording has been a central and well-studied task within the wide area of audio structure analysis, see, e. g., [2, 5, 7, 8, 11, 12] and the overview articles [3, 13]. Even though most of these approaches work well when repetitions largely agree, structure analysis becomes a hard and even ill-posed task when audio segments that refer to the same musical part reveal pronounced musical variations. One way to circumvent such problems is to only visualize structural elements and their relations without explicitly extracting them. For example, in [4] self similarity matrices are used to visualize overall structural patterns or, in [15], repeating and related elements are indicated by arc diagrams.

In this paper, we contribute to this line of research by introducing a novel representation that reveals the hierarchical repetitive structure of a given music recording. Inspired by the work by Sapp [14], we use the concept of a 2D scape plot, where each point represents an audio segment by means of its center and length. As our main contribution, we describe an automated procedure for assigning to each point a color value such that the repetitive structure of the music recording becomes apparent. On the one hand, we use the lightness component of the color to indicate the repetitiveness of the respective segment. This repetitiveness is expressed in terms of a fitness measure as recently introduced by Müller et al. [10]. On the other hand, we use the hue component of the color to reveal the relations across different segments, where we introduce a function that maps related segments to similar hue values. As a result, one obtains a hierarchical structure visualization of the underlying music recording referred to *structure scape plot*, see Figure 4g for an example. We hope that this representation not only visually appeals to the reader, but also brings valuable and even surprising insights into the structural properties of a recording.

The remainder of this paper is organized as follows. In Section 2, we review the underlying fitness measure and describe the corresponding fitness scape plot. Then, in Section 3, we introduce our structure scape plot representation which is based on a novel distance measure to compare different segments as well as on an efficient grouping and coloring procedure. Based on a number of explicit examples, we discuss benefits and limitations of our structure visualization in Section 4 and conclude with Section 5 by indicating future work.

2. FITNESS SCAPE PLOT

In this section, we summarize the construction of the fitness measure (Section 2.1) and then introduce the concept of a fitness scape plot (Section 2.2).

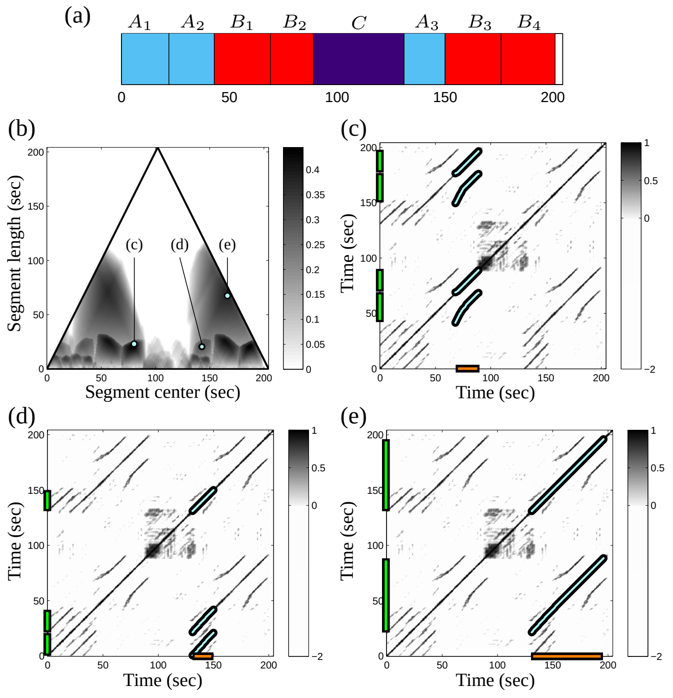


Figure 1: Various representations for an Ormandy recording of Brahms' Hungarian Dance No. 5. **(a)** Musical form $A_1A_2B_1B_2CA_3B_3B_4$. **(b)** Fitness scape plot. The remaining subfigures show the SSM with optimal path families for various segments α (horizontal axis) and induced segment families (vertical axis). **(c)** $\alpha = [68:89]$ (thumbnail, maximal fitness, corresponding to B_2). **(d)** $\alpha = [131:150]$ (corresponding to A_3). **(e)** $\alpha = [131:196]$ (corresponding to $A_3B_3B_4$).

2.1 Fitness Measure

Let $[1:N] = \{1, 2, \ldots N\}$ denote the (sampled) time axis of a given music recording. Then a segment is a subset $\alpha = [s:t] \subseteq [1:N]$ specified by its starting point s and its end point t. Let $|\alpha| := t-s+1$ denote the length of segment α. In [10], a fitness measure has been introduced that assigns to each audio segment α a fitness value $\varphi(\alpha) \in \mathbb{R}$ which simultaneously captures two aspects. Firstly, it indicates how well the given segment explains other related segments. Secondly, it indicates how much of the overall music recording is covered by all these related segments. In the computation of the fitness measure, an enhanced self-similarity matrix (SSM) is computed from the music recording based on chroma-based audio features. It is well known that each path of the SSM (a stripe of high score running parallel to the main diagonal) reveals the similarity of two segments (given by the two projections of the path onto the vertical axis and horizontal axis), see [13]. The main idea of [10] is to compute for each audio segment α a so-called *optimal path family* over α that simultaneously reveals the relations between α and all other similar segments. By projecting such optimal path family to the vertical axis, we get the corresponding induced segment family, where each element of this family defines a segment similar to α.

As an example, we consider a recording of the Hungarian Dance No. 5 by Johannes Brahms, which has the musical form $A_1A_2B_1B_2CA_3B_3B_4$, see Figure 1a. Figure 1c shows an optimal path family (cyan stripes) for the B_2-segment $\alpha = [68:89]$ (horizontal axis) as well as the induced segment family (vertical axis). The induced segmentation consists of four segments corresponding to the four occurrences of the B-part in this recording. Note that repeating segments may be played in different tempi. For example, the B_2-part is played much faster than the B_1-part. Similarly, Figure 1d shows the optimal path family for the segment $\alpha = [131:150]$ (corresponding to the A_3-part) and the induced segmentation (consisting of the three A-part segments). Finally, Figure 1e reveals that, for the long segment $\alpha = [131:196]$ (corresponding to $A_3B_3B_4$), there exists a similar segment (corresponding to $A_2B_1B_2$).

The fitness value of a given segment is derived from the corresponding optimal path family and the values of the underlying SSM. Intuitively, one considers the overall score accumulated by the path family and the total length covered by the induced segmentation. After a suitable normalization, the fitness is defined as the harmonic mean of of coverage and score. For further details, we refer to [10].

2.2 Scape Plot Representation

We now describe how a compact fitness representation for the entire music recording can be obtained showing the fitness $\varphi(\alpha)$ for all possible segments α. Note that each segment $\alpha = [s:t]$ is specified by its center $c(\alpha) := (s+t)/2$ and its length $|\alpha|$. Using the center as horizontal coordinate and the length as vertical coordinate, each segment can be represented as a point in some triangular representation also referred to as *scape plot*. Such scape plots were original introduced by Sapp [14] to represent harmony in musical scores in a hierarchical way. In our context, we define a scape plot Φ by setting $\Phi(c(\alpha), |\alpha|) := \varphi(\alpha)$ for segment α. Figure 1b shows a visualization of the fitness scape plot for our Brahms example, where the fitness is represented by a lightness grayscale ranging from white (fitness is zero) to black (fitness is high). The points corresponding to the three segments discussed above are marked within the scape plot by small circles. For example, the segment $\alpha = [68:89]$ (corresponding to B_2) has the scape plot coordinates $c(\alpha) = 78.5$ (horizontal axis) and $|\alpha| = 22$ (vertical axis). Actually, this segment has the highest fitness among all possible segments and is also referred to as *thumbnail* [10].

The fitness scape plot represents the repetitiveness of each segment in a compact and hierarchical form. For example, in our Brahms example, the repeating segments corresponding to the A-parts and B-parts are reflected by local maxima in the scape plot. Also the repetitions of the superordinate segments corresponding to ABB are captured by the plot. However, so far, the visualization of the fitness scape plot does not reveal the relations *across* different segments. In other words, nothing is said about groups of pairwise similar segment corresponding to the various musical parts.

3. STRUCTURE SCAPE PLOT

Actually, the grouping information is implicitly encoded by the optimal path families underlying the fitness measure. To make these relations more explicit, we now extend the grayscale of the fitness scape plot by a color component that reflects the cross-segment relations. Based on the induced segmentations, we first introduce a distance measure that allows for comparing two arbitrary segments (Section 3.1). Then the objective is to map similar segments to similar colors and dissimilar segments to distinct colors. In the following, we proceed in several steps including a color mapping step (Section 3.2), a point sampling and interpolation step (Section 3.3), and a color combination step (Section 3.4). The overall pipeline of our procedure is also illustrated by Figure 4.

3.1 Segment Distance Measure

Recall from Section 2.1 that for a given segment α there is an optimal path family along with an induced segment family, where each segment of this family is similar to α. Let $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_K\}$ denote the induced segment family of α, then the segments α_k, $k \in [1:K]$, can be thought of as the (approximate) repetitions of α. Note that, by definition, overlaps between repetitions are not allowed, see [10].

Now, let α and β be two arbitrary segments. Intuitively, we consider these two segments to be close if they are approximately repetitions of each other (or at least if some repetitions of α and β have a substantial overlap), otherwise α and β are considered to be far apart. More precisely, let $\mathcal{A} = \{\alpha_1, \ldots, \alpha_K\}$ and $\mathcal{B} = \{\beta_1, \ldots, \beta_L\}$ be the respective induced segment families. Then, we define the distance $\delta(\alpha, \beta)$ between α and β to be

$$\delta(\alpha, \beta) := 1 - \max_{k\in[1:K],\ell\in[1:L]} \frac{|\alpha_k \cap \beta_\ell|}{|\alpha_k \cup \beta_\ell|}, \qquad (1)$$

see also Figure 2 for an illustration. In other words, the distance is obtained by subtracting the maximal overlap (relative to the union) over all repetitions of α and β from the value 1. For example, the B_1-segment and B_2-segment for the Brahms recording have a small distance (close to zero) since the induced segment families more or less coincide (consisting of the four B-part segments). In contrast the B_1-segment and the A_1-segment have a large distance (close to one) since none of their repetitions have a substantial overlap.

3.2 Color Mapping

Based on the distance measure δ, we now introduce a procedure for mapping the scape plot points (segments) to color values in such a way that distance relations are preserved. To this end, we first need to specify a suitable color space. Because of its perceptual relevance, we revert to the HSL model, which is a cylindric parametrization of the RGB color space [6]. Here the angle coordinate $H \in [0, 360]$ (given in degrees) refers to the hue, the coordinate $S \in [0, 1]$ to the saturation, and the coordinate

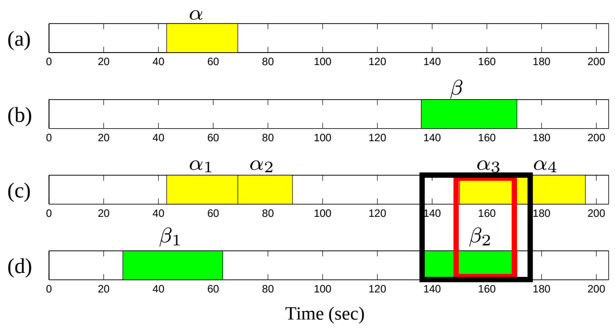


Figure 2: Illustration of the computation of the distance measures $\delta(\alpha, \beta)$ used to compare two segments α (shown in **(a)**) and β (shown in **(b)**). The respective induced segment families are shown in **(c)** and **(d)**, respectively. The black box indicates the union and the red box the overlap of the two segments which are used to compute distance value $\delta(\alpha, \beta)$.

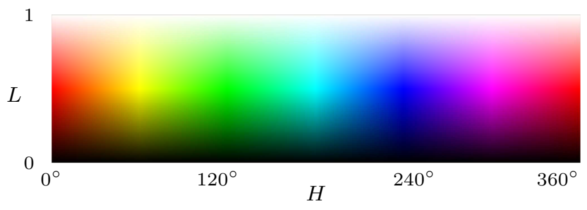


Figure 3: Cylindric HSL (hue, saturation, lightness) color representation. The figure shows only the outside surface of the cylinder corresponding to the saturation $S = 1$.

$L \in [0, 1]$ (with 0 being black and 1 being white) to the lightness of the color. To obtain "full" saturated colors, we fix the parameter $S = 1$. Figure 3 shows the color space for $S = 1$ spanned by the coordinates H and L. Note that the hue angle coordinates $H = 0$ and $H = 360$ encode the same color (by definition this is the color "red"). In the following, we reserve the lightness coordinate to represent the fitness value and only use the hue coordinate to represent the cross-segment relationships.

The problem of mapping the scape plot points to the hue color coordinates (which topologically corresponds to the unit circle) in a distance preserving way can be seen as an instance of *multidimensional scaling* (MDS), see [1]. Generally, MDS refers to a family of related techniques which allow for mapping a set of points with pairwise distance values onto a low-dimensional Euclidean space (often dimension 2 or 3 for visualization purposes) such that the distances between the original points are approximated by the Euclidean distances of the mapped points.

In the following, we use basic MDS techniques to map the scape plot points onto the unit circle (representing the hue color space). Let M denote the number of scape plot points to be considered in the mapping, see Figure 4b. First, we compute an $M \times M$-distance matrix Δ by comparing the M points in a pairwise fashion using δ. Next, we perform a principal component analysis (PCA) of Δ and consider the two eigenvectors corresponding to the two largest eigenvalues. The columns of Δ (which are indexed by the M scape plot points) are then projected onto the two-dimensional Euclidean space defined by these two eigenvectors, see Figure 4c. Using PCA, the variance across the mapped column vectors is maximized. Therefore, scape plot points that have a distinct distance distri-

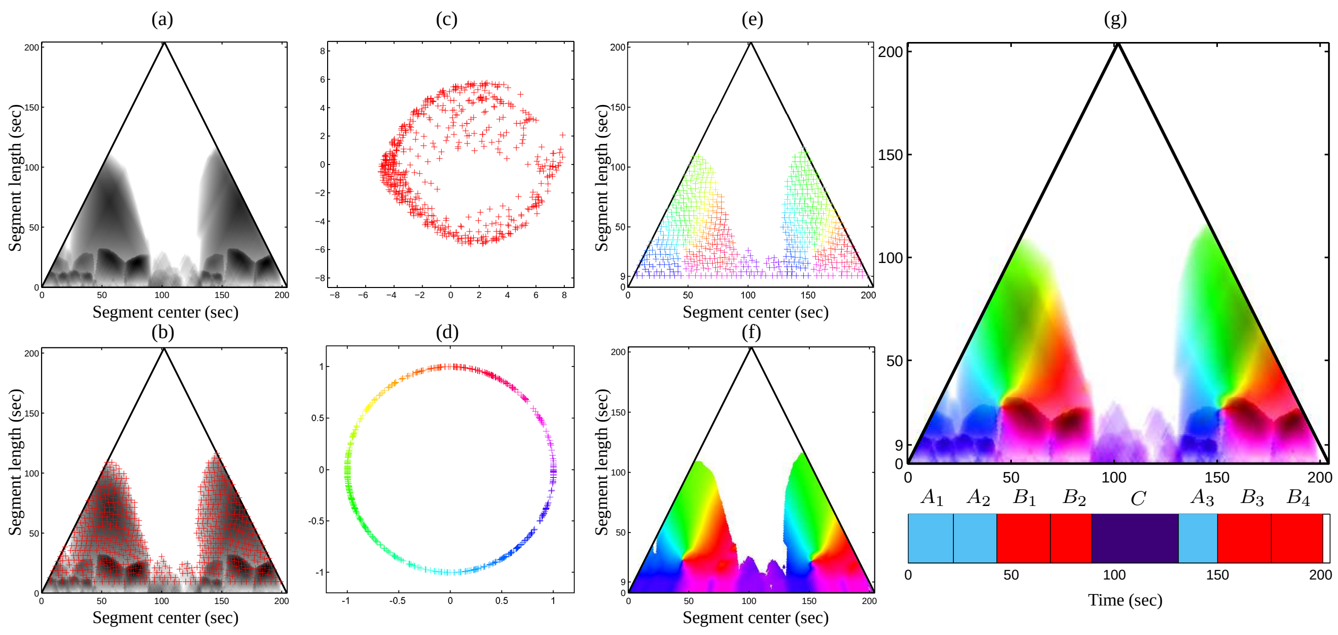


Figure 4: Illustration of the pipeline for computing the structure scape plot for Brahms. **(a)** Fitness scape plot. **(b)** Fitness scape plot with sampled anchor points. **(c)** Anchor points projected onto the first two principal components. **(d)** Anchor points projected to the unit circle colored with the resulting hue value. **(e)** Hue-colored anchor points. **(f)** Hue-colored scape plot using interpolation techniques. **(g)** Structure scape plot combining fitness (lightness) and cross-segment relation (hue) information.

bution to the other points (encoded by its respective column vectors) are likely to be mapped to different regions in the 2D space, see [1] for details. Furthermore, as shown in Figure 4c, the projected points are usually distributed in a circular fashion (even though this is not guaranteed and crucially depends on the distance distributions of the original points). Finally, we normalize the projected points with respect to the Euclidean norm to obtain points on the unit circle, which yields angle parameters that are associated to hue values, see Figure 4d. Figure 4e shows the original scape plot points colored with the derived hue values.

3.3 Sampling and Interpolation

Using all scape plot points in the described color mapping procedure may be problematic because of two reasons. Firstly, using a large number M of points would not only make the computation of the $M \times M$ distance matrix Δ but also of the subsequent PCA rather expensive. Therefore, the number M of used points should be kept small. Secondly, using all scape plot points may over-represent segments of short lengths that are located in the lower part of the triangular scape plot. As a result, the distance relations of the short segments may dominate the selection of the eigenvectors obtained in the PCA step. Therefore, we only choose a suitable subset of scape plot points, also referred to as *anchor points*, and then transfer the obtained hue color information to the other points using interpolation techniques.

Note that scape plot points of higher fitness are structurally more relevant than scape plot points of lower fitness. Therefore, in the anchor point selection step, we sample the scape plot by taking the fitness into account. To this end, we use a greedy procedure that consists of two steps. Firstly, we select the scape plot point of maximal fitness as an anchor point. Secondly, around this anchor point, we specify a neighborhood of size $\rho > 0$ and set the fitness values of all points in this neighborhood to zero excluding them for the subsequent procedure. The role of the neighborhood is to avoid a sampling that is locally too dense. This procedure is repeated until either all of the remaining scape plot points have a fitness of zero, or until a specified maximal number of points M_0 is reached, see also Figure 4b.

Sometimes the fitness values of short segments are rather "noisy." This may also have musical reasons since such segments often correspond to highly repetitive fragments like a short riff or a single chord of dominant harmony. Therefore, it is often beneficial to exclude such short segments in the anchor point selection by only considering scape plot points whose length coordinate lies above a certain lower bound $\lambda > 0$. The influence of the parameters M_0, ρ, and λ on the resulting number of anchor points M is discussed in Section 4.

The color mapping as described in Section 3.2 is now applied only to the anchor points. In the next step, the color information is transferred to arbitrary scape plot points by simply interpolating color values of the nearest neighborhood anchor points. However, since the hue values live on a unit circle (rather than in the two-dimensional Euclidean space), one needs to use spherical interpolation instead of linear interpolation. Figure 4f shows the interpolation result obtained from the anchor points of Figure 4e.

3.4 Color Combination

So far, we have derived two scape plot visualizations: one indicating the repetitive properties (fitness value represented by lightness, see Figure 4a) and the other indicating the cross-segment relations (represented by hue colors, see Figure 4f). We now combine this information within a single scape plot representation, which we also refer to as

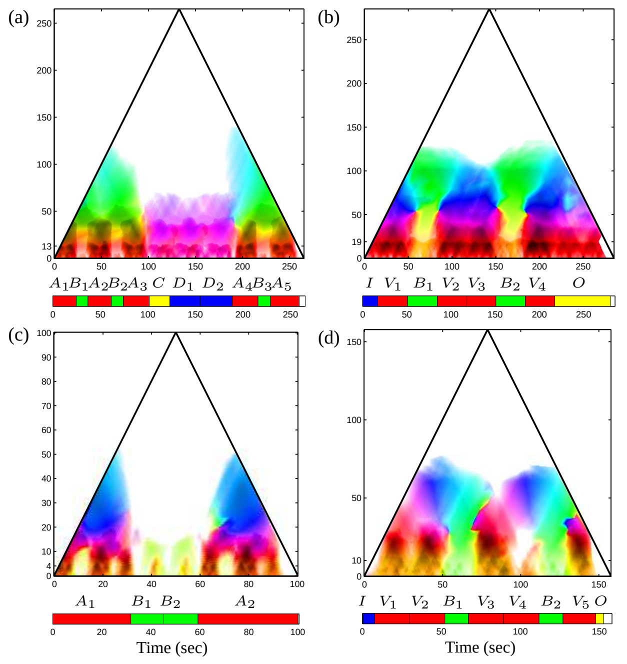


Figure 5: Structure scape plots and structure annotations for recordings of various pieces. **(a)** Chopin Mazurka Op. 17 No. 3. **(b)** Beatles song "While My Guitar Gently Weeps." **(c)** Chopin Mazurka Op. 33 No. 3. **(d)** Beatles song "You Can't Do That."

structure scape plot. To this end, we first linearly map the fitness values onto the lightness parameter space $[0, 1]$ of the HSL model such that $L = 1$ (white) corresponds to the fitness value 0 and $L = 0$ (black) to the maximal fitness value occurring in the fitness scape plot. Furthermore, by rotating the hue parameter space (unit circle) we normalize the color assignment such that the thumbnail (fitness-maximizing scape plot point) is mapped to the color "red" (angle $H = 0$). Finally, for each scape plot point we use the saturation $S = 1$, the computed lightness L, and the normalized hue angle H to obtain a single color value.

Figure 4g shows the final result of the structure scape plot for our Brahms example. Note that the four B-part segments (repetitions of the B_2-thumbnail) are represented by red, the three A-part segments by blue, and the superordinate two ABB-part segments by green. Furthermore, the visualization reveals some substructures of the A-parts, each actually consisting of two (approximate) repetitions. Finally, note that smaller segments within the C-part are assigned to the color violet. Since the C-part contains many fragments sharing the same harmony, our procedure has captured some repetitiveness also in this middle part.

4. EXAMPLES AND DISCUSSION

In this section, we indicate the potential and some limitations of our visualization procedure by discussing representative examples. In our experiments, we used audio recordings considering popular music as well as classical music. On the one hand, we employed the dataset consisting of recordings of the 12 studio albums by "The Beatles" using the structure annotations as described by [9]. On the other hand, we used the complete Rubinstein (1966) recordings of the 49 Mazurkas composed by Frédéric Chopin, where we manually generated some structure annotations for each piece. Note that these annotations are not needed to generate the structure scape plots, but are only used to compare our visualizations with some sort of ground truth. As mentioned in the introduction, the purpose of the scape plot visualizations is to yield a compact and intuitive representation without the necessity of explicitly extracting the structure.

As for the parameter settings, we choose M_0, ρ, and λ in a relative fashion depending on the duration of the respective music recording. In particular, we determined the upper bound M_0 and the neighborhood parameter ρ to result in a number M of anchor points ranging between 200 and 250 for each recording. Furthermore, the lower bound λ was set to correspond to 5-$7\,\%$ of the recording's total duration. Figure 5 shows structure scape plots for some representative music recordings. For example, Figure 5a shows the scape plot for a Rubinstein performance of Chopin's Mazurka Op. 17 No. 3. The five A-part segments, which also comprise the thumbnail, are represented by red. Furthermore, the three B-part segments are indicated by a lighter orange color, and the superordinate ABA-part segments are represented by green. Also substructures of the A-part segments are visible: indeed each A-part consists of two similar subparts. Interestingly, the segments corresponding to the C- and the two D-parts are all represented by pink. Actually this is musically meaningful, since each of the two repeating D-parts is only a slight extension of the C-part.

Figure 5b visualizes the structure scape plot for the Beatles song "While My Guitar Gently Weeps." Also in this example, the structure scape plot nicely reflects the overall musical form. Each of the four verse segments (V-part) consists of two (approximately) repeating subparts, say $V = WW$. Actually, the intro also corresponds to such a subpart ($I = W$) and the outro corresponds to three of these subparts ($O = WWW$), which also explains the red coloring of these segments. Furthermore, the color blue corresponds to WWW-segments and the color green to VBV-segments.

The structure scape plot of a recording of the Mazurka Op. 33 No. 3 is shown in Figure 5c, which indicates a number of substructures not reflected in the structure annotation (see both A parts). Finally, Figure 5d correctly reproduces the overall structure of the Beatles song "You Can't Do That." Only the V_4-segment has not been captured well. Actually, V_4 corresponds to an instrumental section with some vocal interjections, which make the V_4-segment spectrally quite different to the other four V-part segments.

Next, we discuss some limitations and problems that may occur in our visualization approach. As an illustrating example, we consider the Beatles song "Hello Goodbye." Figure 6b shows the structure scape plot using our standard parameter setting as described above. The red color corresponds to the four VR-part segments, which also comprise the thumbail. However, the individual V-part and R-part segments are all represented by green and are not distin-

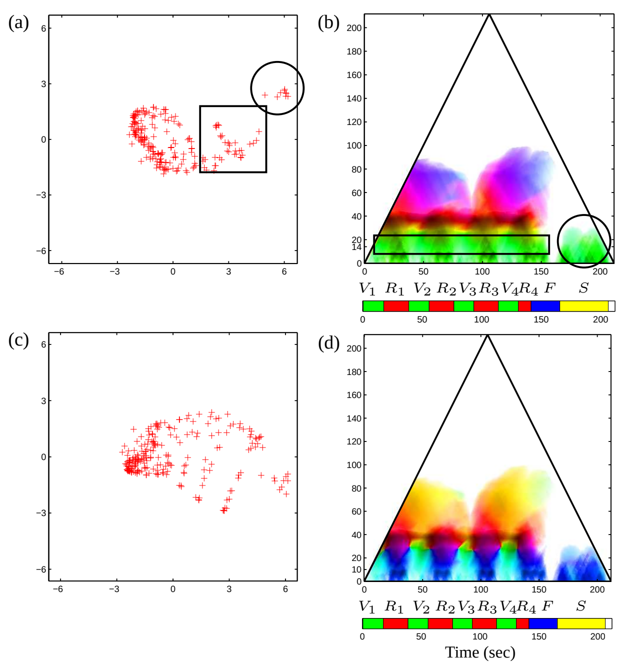


Figure 6: Anchor points projected onto the first two principal components (left) and resulting structure scape plot (right) for the Beatles song "Hello Goodbye." **(a)/(b)** Using $\lambda = 14$ seconds. **(c)/(d)** Using $\lambda = 10$ seconds.

guishable. The reason for this is that the lower bound for the anchor points was set to $\lambda = 14$ seconds, which is too high to capture the finer structures. By decreasing this parameter to $\lambda = 10$ seconds, V-part and R-part segments are separated, see Figure 6d. As this example shows, the choice of the parameter λ may have a significant impact on the final visualization. The Beatles example also indicates a second problem that may arise in our color mapping procedure. Usually, the anchor points projected to the two principal components are homogeneously distributed along the unit circle as in our Brahms example, see Figure 4c. Therefore, projecting these points to the unit circle (to yield the desired hue values) does not destroy too much of the neighborhood relations. However, in the Beatles example, the projected anchor points are rather scattered in the two-dimensional Euclidean space with some outliers as indicated by the boxed and circled points shown in Figure 6a. Therefore, projecting these points onto the unit circle may result in the same hue value for anchor points that are actually far apart. This explains, why the substructures within the S-part are mapped to the same color as substructures of the VR-part, see Figure 6b.

5. FUTURE WORK

These problems indicate some future research directions. Possible improvements of the color mapping step may be achieved by applying more involved generalized multidimensional scaling techniques which directly map the anchor points to a smooth manifold (in our case the unit circle). Also, the one-dimensional hue color space may not suffice to suitable capture more intricate cross-segment relations. Here, a more flexible usage of the color space or an extension to 3D scape plot representations may help to better represent more complex structures. So far, we have only given a qualitative evaluation to demonstrate the potential of our techniques. In this context, user studies may be necessary to better understand the actual user needs and the applicability of our concepts. Besides introducing a novel segment distance function as well as a grouping and coloring procedure, the main contribution of this paper was to introduce the concept of a structure scape plot for visualizing repetitive structures of music recordings. We hope that our visualization is not only aesthetically appealing, but also may allow a user to explore and browse musical structures in novel ways.

Acknowledgments: This work has been supported by the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University and the German Research Foundation (DFG MU 2686/5-1).

6. REFERENCES

[1] Ingwer Borg and Patrick J. F. Groenen. *Modern Multidimensional Scaling Theory and Applications*. Springer, 2005.

[2] Matthew Cooper and Jonathan Foote. Summarizing popular music via structural similarity analysis. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 127–130, New Paltz, NY, USA, 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] Jonathan Foote. Visualizing music and audio using self-similarity. In *Proc. ACM International Conference on Multimedia*, pages 77–80, Orlando, FL, USA, 1999.

[5] Masataka Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Trans. Audio, Speech and Language Processing*, 14(5):1783–1794, 2006.

[6] Allan Hanbury. Constructing cylindrical coordinate colour spaces. *Pattern Recognition Letters*, 29:494–500, 2008.

[7] Mark Levy, Mark Sandler, and Michael A. Casey. Extraction of high-level musical structure from audio data and its application to thumbnail generation. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 13–16, Toulouse, France, 2006.

[8] Namunu C. Maddage. Automatic structure detection for popular music. *IEEE Multimedia*, 13(1):65–77, 2006.

[9] Matthias Mauch, Chris Cannam, Matthew E.P. Davies, Simon Dixon, Christopher Harte, Sefki Kolozali, Dan Tidhar, and Mark Sandler. OMRAS2 metadata project 2009. In *Late Breaking Demo, International Conference on Music Information Retrieval (ISMIR)*, Kobe, Japan, 2009.

[10] Meinard Müller, Peter Grosche, and Nanzhu Jiang. A segment-based fitness measure for capturing repetitive structures of music recordings. In *Proc. 12th International Conference on Music Information Retrieval (ISMIR)*, pages 615–620, Miami, FL, USA, 2011.

[11] Meinard Müller and Frank Kurth. Towards structural analysis of audio recordings in the presence of musical variations. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007.

[12] Jouni Paulus and Anssi P. 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.

[13] Jouni Paulus, Meinard Müller, and Anssi P. Klapuri. Audio-based music structure analysis. In *Proc. 11th International Conference on Music Information Retrieval (ISMIR)*, pages 625–636, Utrecht, The Netherlands, 2010.

[14] Craig Stuart Sapp. Harmonic visualizations of tonal music. In *Proc. International Computer Music Conference (ICMC)*, pages 423–430, 2001.

[15] Ho-Hsiang Wu and Juan P. Bello. Audio-based music visualization for music structure analysis. In *Proceedings of Sound and Music Computing Conference (SMC)*, Barcelona, Spain, 2010.

A SYSTEMATIC COMPARISON OF MUSIC SIMILARITY ADAPTATION APPROACHES

Daniel Wolff*, Tillman Weyde
MIRG, School of Informatics
City University London, UK
daniel.wolff.1@city.ac.uk

Sebastian Stober*, Andreas Nürnberger
Data & Knowledge Engineering Group
Otto-von-Guericke-Universität Magdeburg, DE
stober@ovgu.de

ABSTRACT

In order to support individual user perspectives and different retrieval tasks, music similarity can no longer be considered as a static element of Music Information Retrieval (MIR) systems. Various approaches have been proposed recently that allow dynamic adaptation of music similarity measures. This paper provides a systematic comparison of algorithms for metric learning and higher-level facet distance weighting on the *MagnaTagATune* dataset. A cross-validation variant taking into account clip availability is presented. Applied on user generated similarity data, its effect on adaptation performance is analyzed. Special attention is paid to the amount of training data necessary for making similarity predictions on unknown data, the number of model parameters and the amount of information available about the music itself.

1. INTRODUCTION

Musical similarity is a central issue in MIR and the key to many applications. In the classical retrieval scenario, similarity is used as an estimate for relevance to rank a list of songs or melodies. Further applications comprise the sorting and organization of music collections by grouping similar music clips or generating maps for a collection overview. Finally, music recommender systems that follow the popular "find me more like…"-idea often employ a similarity-based strategy as well. However, music similarity is not a simple concept. In fact there exist various frameworks within musicology, psychology, and cognitive science. For a comparison of music clips, many interrelated features and facets can be considered. Their individual importance and how they should be combined depend very much on the user and her or his specific retrieval task. Users of MIR systems may have various (musical) backgrounds and experience music in different ways. Consequently, when comparing musical clips with each other, opinions may diverge. Apart from considering individual users or user groups, similarity measures also should be tailored to their specific retrieval task to improve the performance of the retrieval system. For instance, when looking for cover versions of a song, the timbre may be less interesting than the lyrics. Various machine learning approaches have recently been proposed for adapting a music similarity measure for a specific purpose. They are briefly reviewed in Section 2. For a systematic comparison of these approaches, a benchmark experiment based on the *MagnaTagATune* dataset has been designed, which is described in Section 3. Section 4 discusses the results of the comparison and Section 5 finally draws conclusions.

*The two leading authors contributed equally to this work.

2. ADAPTATION APPROACHES

The approaches covered in this paper focus on learning a distance measure, which (from a mathematical perspective) can be considered as a dual concept to similarity. The learning process is guided by so-called *relative distance constraints*. A relative distance constraint (s, a, b) demands that the object a is closer to the seed object s than object b, i.e.,

$$d(s, a) < d(s, b) \qquad (1)$$

Such constraints can be seen as atomic bits of information fed to the adaptation algorithm. They can be derived from a variety of higher-level application-dependent constraints. For instance, in the context of interactive clustering, assigning a song s to a target cluster with the prototype c_t can be interpreted by the following set of relative distance constraints as proposed by Stober et al. [11]:

$$d(s, c_t) < d(s, c) \qquad \forall c \in C \setminus \{c_t\} \qquad (2)$$

where C is the set of cluster prototypes. Bade et al. describe how relative distance constraints can be derived from expert classifications of folk songs [1] or from an existing personal hierarchy of folders with music files [2]. Alternatively, it is also possible to directly ask the users to state the opinion for a triplet of songs as in the bonus round of the *TagATune* game [7]. (Section 3.2 covers this in detail.) McFee et al. [8] use artist similarity triples collected in the web survey described in [5]. They also describe a graph-based technique to detect and remove inconsistencies within sets of constraints such as direct contradictions.

Using relative distance constraints, the task of learning a suitable adaptation of a distance measure can be formulated mathematically as constraint optimization problem.

In the following, the two general approaches covered in this comparison are briefly reviewed.

2.1 Linear Combinations of Facet Distances

Stober et al. model the distance $d(a, b)$ between two songs as weighted sum of *facet distances* $\delta_{f_1}(a, b), \ldots, \delta_{f_l}(a, b)$:

$$d(a, b) = \sum_{i=1}^{l} w_i \delta_{f_i}(a, b) \tag{3}$$

Each facet distance refers to an objective comparison of two music clips with respect to a single facet of music information such as melody, timbre, or rhythm. Here, the facet weights $w_1, \ldots, w_l \in \mathbb{R}^+$ serve as parameters of the distance measure that allow to adapt the importance of each facet to a specific user or retrieval task. These weights obviously have to be non-negative so that the aggregated distance cannot decrease where a single facet distance increases. Furthermore, the sum of the weights should be constant such as

$$\sum_{i=1}^{l} w_i = l \tag{4}$$

to avoid arbitrarily large distance values.

The small number of parameters somewhat limits the expressivity of the distance model. However, at the same time, the weights can easily be understood and directly manipulated by the user. Stober et al. argue that this design choice specifically addresses the users' desire to remain in control and not to be patronized by an intelligent system that "knows better". In [11], they describe various applications and respective adaptation algorithms which they evaluate and compare in [12] using the *MagnaTagATune* dataset. Three of these approaches are covered by the comparison in this paper.

2.1.1 *Gradient Descent*

Here, if a constraint is violated by the current distance measure, the weighting is updated by trying to maximize

$$obj(s, a, b) = \sum_{i=1}^{l} w_i(\delta_{f_i}(s, b) - \delta_{f_i}(s, a)) \tag{5}$$

which can be directly derived from Equation 1. This leads to the following update rule for the individual weights:

$$w_i = w_i + \eta \Delta w_i, \quad \text{with} \tag{6}$$

$$\Delta w_i = \frac{\partial obj(s, a, b)}{\partial w_i} = \delta_{f_i}(s, b) - \delta_{f_i}(s, a) \tag{7}$$

where the learning rate η defines the step width of each iteration. As in [12], the optimization process is restarted 50 times with random initialization and the best result is chosen to reduce the risk of getting stuck in a local optimum.

2.1.2 *Quadratic Programming*

Of the various quadratic programming approaches covered in [12], only the one minimizing the quadratic slack is considered here because it was the best performing one in the original comparison. In this approach, an individual slack variable is used for each constraint, which allows violations. As optimization objective, the sum of the squared slack values has to be minimized.

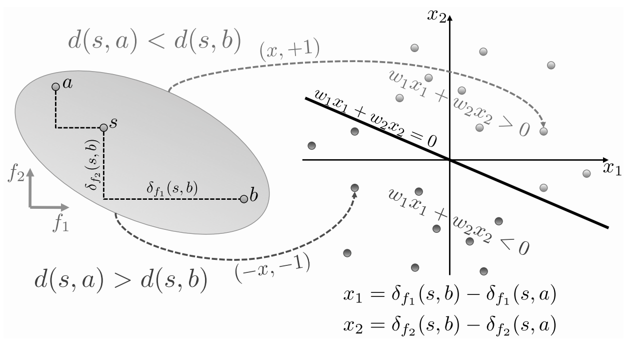


Figure 1. Transformation of a relative distance constraint for linear combination models into two training instances of the corresponding binary classification problem as described by Cheng et al. [3].

2.1.3 *Linear Support Vector Machine (LibLinear)*

The third approach takes a very different perspective. As described by Cheng et al. [3], the learning task can be reformulated as a binary classification problem, which opens the possibility to apply a wide range of sophisticated classification techniques such as (linear) Support Vector Machines (SVMs). Figure 1 illustrates this idea to rewrite each relative distance constraint $d(s, a) < d(s, b)$ as

$$\sum_{i=1}^{m} w_i(\delta_{f_i}(s, b) - \delta_{f_i}(s, a)) = \sum_{i=1}^{m} w_i x_i = \mathbf{w}^T \mathbf{x} > 0 \tag{8}$$

where x_i is the *distance difference* with respect to facet f_i. The positive training example $(\mathbf{x}, +1)$ then represents the satisfied constraint whereas the negative example $(-\mathbf{x}, -1)$ represents its violation (i.e., inverting the relation sign). For these training examples, the normal vector of the hyperplane that separates the positive and negative instances contains the adapted facet weights. As in [12], the *LibLinear* library is used here, which finds a stable separating hyperplane but still suffers from the so far unresolved problem that the non-negativity of the facet weights cannot be enforced.

2.2 Metric Learning

Alternative approaches to weighting predefined facet distance measures include direct manipulation of parametrized vector distance measures. All features are concatenated to a single combined feature vector per clip. We model a clip's feature vector by $g(a) : \mathbb{N} \mapsto \mathbb{R}^N$. This corresponds to assigning a single facet to each feature dimension. Frequently, the mathematical form of Mahalanobis metrics is used to specify a parametrized vector distance measure. In contrast to the approaches described in the previous section, adaptation is performed in the (combined) feature space itself: Given two feature vectors $\mathbf{a} = g(a)$, $\mathbf{b} = g(b) \in \mathbb{R}^N$, the family of Mahalanobis distance measures can be expressed by

$$d_{\mathbf{W}}(\mathbf{a}, \mathbf{b}) = \sqrt{(\mathbf{a} - \mathbf{b})^T \mathbf{W} (\mathbf{a} - \mathbf{b})}, \tag{9}$$

where $\mathbf{W} \in \mathbb{R}^{N \times N}$ is a positive semidefinite matrix, parametrizing the distance function. Generic variants of the

Euclidean metric, Mahalanobis metrics allow for linear transformation of the feature space when accessing distance. An important property of this approach is that the number of adjustable parameters directly depends on the dimensionality N of the feature space. As this number grows quadratically with N, many approaches restrict training to the N parameters of a diagonal matrix $\mathbf{W}$, only permitting a weighting of the individual feature dimensions.

2.2.1 *Linear Support Vector Machine (SVMLight)*

The SVM approach explained in Section 2.1.3 has been shown as well suited to learning a Mahalanobis distance measure: Schultz et al. [10] adapted a weighted kernelized metric towards relative distance constraints. We follow the approach of Wolff et al. [13], where a linear kernel is used. This simplifies the approach of Schultz et al. to learning a diagonally restricted Mahalanobis distance (Equation 9).

Like the SVM for the facet distances, a large margin classifier is optimized to the distance constraints. Here, for each constraint (s, a, b), we replace the facet *distance difference* vector $\mathbf{x}$ in Equation 8 with the difference of the pointwise squared [1] feature difference vectors $\mathbf{x} = (\mathbf{s} - \mathbf{b})^2 - (\mathbf{s} - \mathbf{a})^2$.

Given the vector $\mathbf{w} = \text{diag}(\mathbf{W}), \mathbf{w}_i \geq 0$ and slack variables $\xi_{(s,a,b)} \geq 0$, optimization is performed as follows:

$$\min_{\mathbf{w},\xi} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + c \cdot \sum_{(s,a,b)} \xi_{(s,a,b)} \qquad (10)$$
$$\text{s.t.} \, \forall \, (s,a,b) \qquad \mathbf{w}^T\mathbf{x}_{(s,a,b)} \geq 1 - \xi_{(s,a,b)}$$

Here, c determines a trade-off between regularization and the enforcement of constraints. For the experiments below, the SVM^{light} framework [2] is used to optimize the weights $\mathbf{w}_i$. As for *LibLinear*, $\mathbf{w}_i \geq 0$ cannot be guaranteed.

2.2.2 *Metric Learning to Rank*

McFee et al. [9] developed an algorithm for learning a Mahalanobis distance from rankings. [3] Using the constrained regularization of Structural SVM, the matrix $\mathbf{W}$ is optimized to an input of clip rankings and their feature vectors. Given a relative distance constraint (s, a, b) (see Equation 1), the corresponding ranking assigns a higher ranking score to a than to b, when querying clip s. For a set X of training query feature vectors $\mathbf{q} \in X \subset \mathbb{R}^N$ and associated training rankings y_q^*, Metric Learning to Rank (MLR) minimizes

$$\min_{\mathbf{W},\xi} \quad tr(\mathbf{W}^T\mathbf{W}) + c\frac{1}{n}\sum_{q \in X} \xi_q, \qquad (11)$$
$$\text{s.t.} \quad \forall \mathbf{q} \in X, \, \forall y \in Y \setminus \{y_q^*\} :$$
$$H_{\mathbf{W}}\left(\mathbf{q}, y_q^*\right) \geq H_{\mathbf{W}}\left(\mathbf{q}, y\right) + \Delta(y_q^*, y) - \xi_q,$$

with $W_{i,j} \geq 0$ and $\xi_q \geq 0$. Here, the matrix $\mathbf{W}$ is regularized using the trace. Optimization is subject to the constraints creating a minimal slack penalty of ξ_q. c determines the trade-off between regularization and the slack penalty for the constraints below. $H_{\mathbf{W}}(q, y)$ [4] assigns a score to the validity of ranking y given the query $\mathbf{q}$ with regard to the Mahalanobis matrix $\mathbf{W}$. This enforces $\mathbf{W}$ to fulfill the training rankings y_q^*. The additional ranking-loss term $\Delta(y_q^*, y)$ assures a margin between the scores of given training rankings y_q^* and incorrect rankings y. The method is kept efficient by selecting only a few possible alternative rankings $y \in Y$ for comparison with the training rankings: A separation oracle is used for predicting the worst violated constraints (see [6]). In our experiments, an MLR variant *DMLR* restricts $\mathbf{W}$ to a diagonal shape.

3. EXPERIMENT DESIGN

3.1 The *MagnaTagATune* Dataset

MagnaTagATune is a dataset combining mp3 audio, acoustic feature data, user votings for music similarity, and tag data for a set of 25863 clips of about 30 seconds taken from 5405 songs provided by the Magnatune [5] label. The bundled acoustic features have been extracted using version 1.0 of the *EchoNest API* [6] . The tag and similarity data has been collected using the TagATune game [7]. TagATune is a typical instance of an online "Game With A Purpose". While users are playing the game mainly for recreational purposes, they annotate the presented music clips. The tag data is collected during the main mode of the game, where two players have to agree on whether they listen to identical clips. Their communication is saved as tag data. The bonus mode of the game involves a typical odd one out survey asking two players to independently select the same outlier out of three clips presented to them. The triplets of clips presented to them vary widely in genre, containing material from ambient and electronica, classical, alternative, and rock.

3.2 Similarity Data

The comparative similarity data in *MagnaTagATune* can be represented in a constraint multigraph with pairs of clips as nodes [8, 12]. The vote for an outlier k in the clip triplet (i, j, k) is transformed into two relative distance constraints: (i, j, k) and (j, i, k). Each constraint (s, a, b) is represented by an edge from the clip pair (s, a) to (s, b). This results in 15300 edges of which 1598 are unique. In order to adapt similarity measures to this data, the multigraph has to be acyclic, as cycles correspond to inconsistencies in the similarity data. The *MagnaTagATune* similarity data only contains cycles of length 2, corresponding to contradictive user statements regarding the same triplet. In order to remove these cycles, the contradicting multigraph edge numbers are consolidated by subtracting the number of edges connecting the same vertices in opposite directions. The remaining 6898 edges corresponding to 860 unique relative distance constraints constitute the similarity data we work with. [7]

[1] $(\mathbf{a}^2)_i := (\mathbf{a}_i)^2$

[2] http://svmlight.joachims.org/

[3] http://cseweb.ucsd.edu/~bmcfee/code/mlr/

[4] For simplification, $H_{\mathbf{W}}(q, y)$ substitutes the Frobenius product $\langle W, \psi(q, y)\rangle_F$ in [9].

[5] http://magnatune.com/

[6] http://developer.echonest.com/

[7] In [12], the authors report that the number of consistent constraints is 674. This differing number was caused by a software bug in the filtering algorithm, which led to the removal of more constraints than necessary.

3.3 Data Partitioning

In order to assess the training performance of the approaches described in Section 2, we compare two cross-validation variants to specify independent test and training sets.

A straightforward method, randomly sampling the constraints into cross-validation bins and therefore into combinations of test and training sets has been used on the dataset before by Wolff et al. [13]. We use this standard method (sampling A) to perform 10-fold cross validation, sampling the data into non-overlapping test and training sets of 86 and 774 constraints respectively

For the second sampling, it is considered that two constraints were derived from each user voting, as such are related to the same clips. Assigning one of such two constraints to training and the remaining one to a test set might introduce bias by referring to common information. In our second validation approach, (sampling B) it is assured that the test and training sets also perfectly separate on the clip set. The 860 edges of the *MagnaTagATune* similarity multigraph connect 337 components of three vertices each. These correspond to the initial setup of clip triplets presented to the players during the *TagATune* game.

As the removal of one clip causes the loss of all similarity information (maximally 3 constraints) within its triplet, the sampling of the test data is based on the triplets rather than the constraints. On the 337 triplets, we use 10-fold cross validation for dividing these into bins of 33 or 34 triplets. Due to the varying number of 2-3 constraints per triplet, the training set sizes vary from 770-779 constraints, leaving the test sets at 81-90 constraints.

For evaluation of generalization and general performance trends, the training sets are analyzed in an expanding subset manner. We start with individual training sets of either 13 constraints (sampling A) or 5 triplets (sampling B), corresponding to 11-15 constraints. The size of the training sets is then increased exponentially, including all the smaller training sets' constraints in the larger ones. Constraints remaining unused for each of the smaller training set sizes are used for further validation, and referred to as *unused training constraints*. For both sampling methods, all test and training sets are fixed, and referred to as *sampling A* and *sampling B*.

3.4 Features and Facets

As features, we use those defined in [12] plus the genre features used by Wolff et al. [13]. This results in the set of features shown in Table 1.

Of the 7 global features, "danceability" and "energy" were not contained in the original clip analysis information of the dataset but have become available with a newer version of the *EchoNest API*. Furthermore, the segment-based features describing pitch ("chroma") and timbre have been aggregated (per dimension) resulting in 12-dimensional vectors with the mean and standard deviation values. This has been done according to the approach described in [4] for the same dataset. The 99 tags were derived from annotations collected through the *TagATune* game [7] by applying

feature	dim.	value description
key	1	0 to 11 (one of the 12 keys) or -1 (none)
mode	1	0 (minor), 1 (major) or -1 (none)
loudness	1	overall value in decibel (dB)
tempo	1	in beats per minute (bpm)
time signature	1	3 to 7 ($\frac{3}{4}$ to $\frac{7}{4}$), 1 (complex), or -1 (none)
danceability	1	between 0 (low) and 1 (high)
energy	1	between 0 (low) and 1 (high)
pitch mean	12	dimensions correspond to pitch classes
pitch std. dev.	12	dimensions correspond to pitch classes
timbre mean	12	normalized timbre PCA coefficients
timbre std. dev.	12	normalized timbre PCA coefficients
tags	99	binary vector (very sparse)
genres	44	binary vector (very sparse)

Table 1. Features for the *MagnaTagATune* dataset. Top rows: Globally extracted *EchoNest* features. Middle rows: Aggregation of *EchoNest* features extracted per segment. Bottom row: Manual annotations from *TagATune* game and the *Magnatune* label respectively.

the preprocessing steps described in [12]. The resulting binary tag vectors are more dense than for the original 188 tags but still very sparse. The genre labels were obtained from the *Magnatune* label as described by Wolff et al. [13]. A total of 42 genres was assigned to the clips in the test set with 1-3 genre labels per clip. This also results in very sparse binary vectors.

For the facet-based approaches described in Section 2.1, two different sets of facets are considered consisting of 26 and 155 facets respectively. In both sets, the 7 global features are represented as individual facets (using the distance measures described in [12]). As the genre labels are very sparse, they are combined in a single facet using the Jaccard distance measure. The set of 155 facets is obtained by adding 99 tag facets (as in [12]) and a single facet for each dimension of the 12-dimensional pitch and timbre features. For the set of 26 facets, the pitch and timbre feature are represented as a single facet each (combining all 12 dimensions). Furthermore, 14 tag-based facets are added of which 9 refer to aggregated tags that are less sparse (solo, instrumental, voice present, male, female, noise, silence, repetitive, beat) and 5 compare binary vectors for groups of related tags (tempo, genre, location, instruments, perception / mood). This results in a realistic similarity model of reasonable complexity that could still be adapted manually by a user. The more complex model with almost six times as many facet weight parameters serves as the upper bound of the adaptability using a linear approach for the given set of features.

4. RESULTS

For the similarity data sampling A, Figure 2 shows all of the algorithms to improve the baseline of 63% satisfied unknown constraints by 7% to 10%. Plotted are the performance averages over 10-fold cross-validation as described in Section 3.3. Except for SVM^{light}, most of the final generalization success is achieved within the first 250 training constraints. Only diagonal MLR shows notable non-monotonic behaviour for larger training sets >200 constraints. Further tests on the unused training data reproduce the results on the static test sets shown here. As shown in Figure 3, all algorithms are able to satisfy the initial training constraints. With the exception of MLR, (see Section 4.2), the training performance decreases for growing training sets, asymptotically approaching the test set performance. Such effects have been shown in [13] not to contradict good generalization results.

4.1 Impact of Model Complexity

For the facet-based linear approaches (Figure 3, left), a strong impact of the number of facet weight parameters can be observed. Whilst the performance for the model with 155 facets is significantly superior on the training data, it is generally worse on the test data. Only for a high number of training constraints, the simpler model with 26 facets can be matched or slightly outperformed. This is a strong indicator for model overfitting. With its many parameters, the complex model adapts too much to the training data at the cost of a reduced ability to generalize. In contrast, the simple model is able to generalize much quicker. This is especially remarkable for the quadratic programming approach with the quickest generalization of all approaches. Its adaptation performance on the test data also comes closest to the training performance, which can be seen as an upper bound. It appears as if this limit is increased by about 5%, if 155 facets are used instead, but more training examples would be needed to get closer to this value. Here lies great potential for future research: By adapting the model complexity (i.e., the number of parameters) depending on the number of training examples and the performance on some unseen constraints, the ability of simple models to quickly generalize could be combined with the superior adaptability of more complex ones.

4.2 Effects of Similarity Sampling

For most of the algorithms tested, the effect of choosing sampling A or sampling B is small. Best performing are MLR (samling A) and quadratic programming($m = 155$) for sampling B. Except for MLR, decrease in test set performance is limited to 1% when trained with the clip-separating sampling B. In the right column of Figure 3, the metric learning algorithms are compared. The bottom black curves represent the test set results for sampling A (dashed, $\cdot - \cdot$) and sampling B (solid, —). The training performances for these samplings are plotted on the top of the graphs. While SVM^{light} (d) and *DMLR* (f) only loose 2-3% in performance, MLR (e) drops by more than 6%. Exclusively among the algorithms tested, the fully parametrized MLR(e) variant shows a 100% training performance for all training sizes. In line with results from Wolff et al. [13,14], the algorithm generalizes well on the similarity data with sampling A. Even with further permutations of the data, this capability to generalize reduces significantly when using MLR with our sampling method B, possibly caused by the lack of feature reoccurence in the training data.

5. CONCLUSIONS

The results of the experiment show that all approaches can adapt a similarity model to training data and generalize the learned information to unknown test data. Training performance curves can be used as an indicator for the maximal generalization outcome to expect, which depends on the number of facets and the features used. Sensitivity with respect to the sampling method of the test data was observed for MLR, which requires further investigation. Another promising direction for future work is to dynamically adapt the model complexity, e.g., by regularization. The feature data and sampling information are available online [8] for benchmarking of approaches developed in the future.

6. REFERENCES

[1] K. Bade, J. Garbers, S. Stober, F. Wiering, and A. Nürnberger. Supporting folk-song research by automatic metric learning and ranking. In *Proc. of ISMIR'09*, 2009.

[2] K. Bade, A. Nürnberger, and S. Stober. Everything in its right place? learning a user's view of a music collection. In *Proc. of NAG/DAGA'09, Int. Conf. on Acoustics*, 2009.

[3] W. Cheng and E. Hüllermeier. Learning similarity functions from qualitative feedback. In *Proc. of ECCBR'08*, 2008.

[4] J. Donaldson and P. Lamere. Using visualizations for music discovery. Tutorial at *ISMIR'09*, 2009.

[5] D. P. W. Ellis, B. Whitman, A. Berenzweig, and S. Lawrence. The quest for ground truth in musical artist similarity. In *Proc. of ISMIR'02*, 2002.

[6] T. Joachims, T. Finley, and C.-N.J. Yu. Cutting-plane training of structural SVMs. In *Machine Learning*, 2009.

[7] E. Law and L. von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proc. of CHI '09*, 2009.

[8] B. McFee and G. Lanckriet. Heterogeneous embedding for subjective artist similarity. In *Proc. of ISMIR'09*, 2009.

[9] B. McFee and G. Lanckriet. Metric learning to rank. In *Proc. of ICML'10*, 2010.

[10] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *NIPS*, 2003.

[11] S. Stober. Adaptive distance measures for exploration and structuring of music collections. In *Proc. of AES 42nd Conference on Semantic Audio*, 2011.

[12] S. Stober and A. Nürnberger. An experimental comparison of similarity adaptation approaches. In *Proc. of AMR'11*, 2011.

[13] D. Wolff and T. Weyde. Adapting metrics for music similarity using comparative judgements. In *Proc. of ISMIR 2011*, 2011.

[14] D. Wolff and T. Weyde. Adapting Similarity on the MagnaTagATune Database In *ACM Proc. of WWW'12*, 2012.

[8] http://mi.soi.city.ac.uk/datasets/ismir2012/

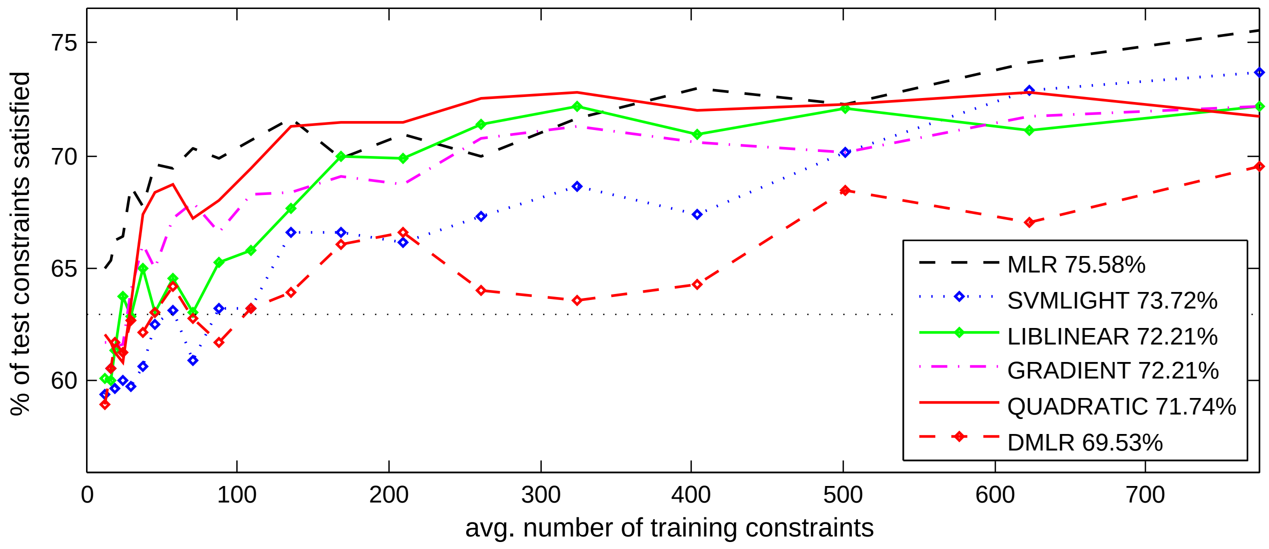


Figure 2. Performance comparison of facet-based approaches (with 26 facets) and metric learning. Values are averaged over all 20 folds of sampling A. The baseline at 63% refers to the mean performance of random facet weights ($n = 1000$).

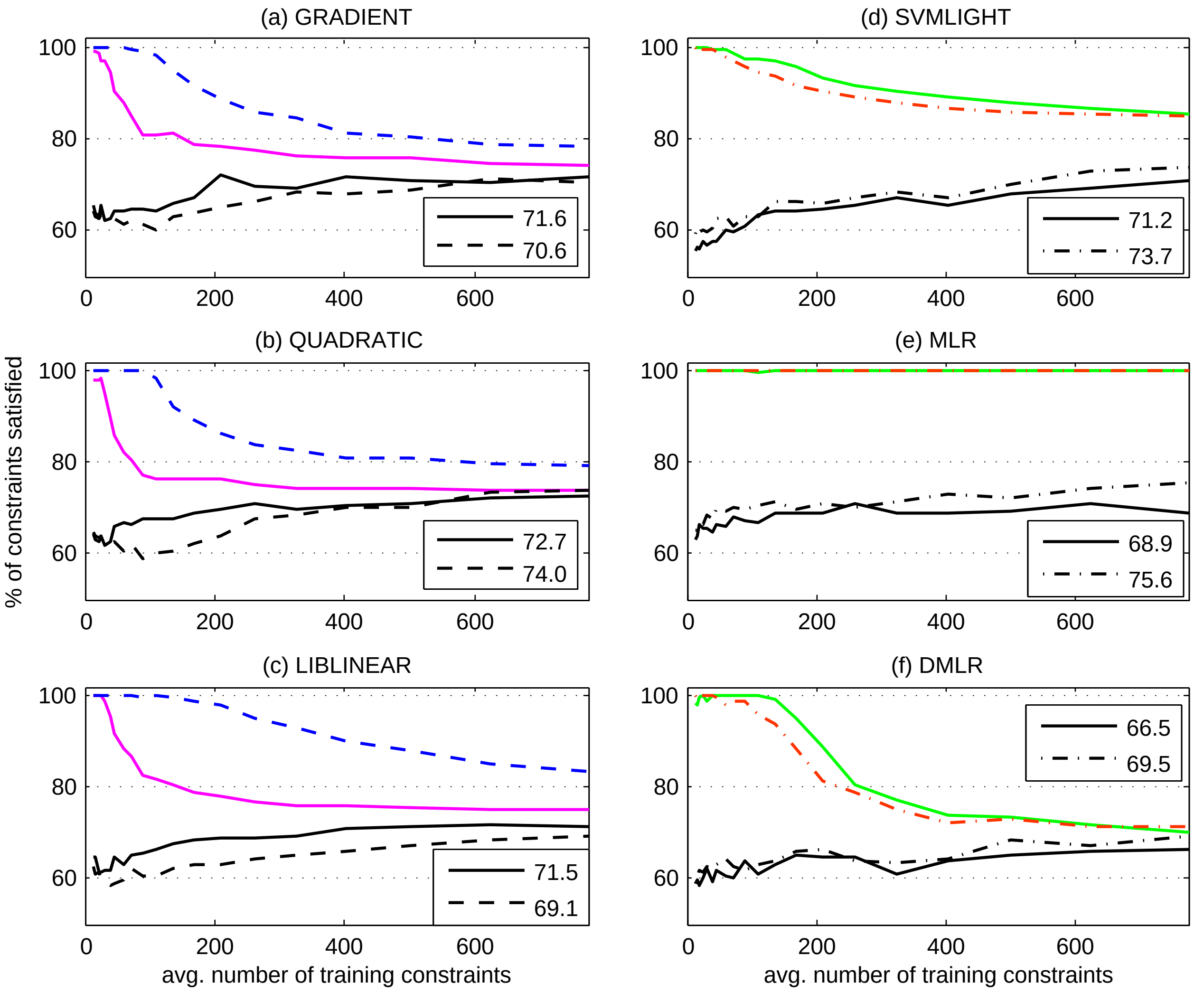


Figure 3. Detailed performance of the individual approaches under different conditions. Top curves show training performance, bottom curves and legend show test set performance. **Left column (a, b, c):** Performance of the facet-based approaches using 26 facets (—) and 155 facets (– –). Comparison based on sampling B. **Right column (d, e, f):** Performance of the metric-based approaches. Effects of sampling A(· – ·) and B(—) are compared.

USING HYPER-GENRE TRAINING TO EXPLORE GENRE INFORMATION FOR AUTOMATIC CHORD ESTIMATION

Yizhao Ni, Matt Mcvicar, Raúl Santos-Rodríguez and Tijl De Bie
Intelligent Systems Laboratory
University of Bristol, U. K.
{enxyn, matt.mcvicar, enrsr, tijl.debie}@bristol.ac.uk

ABSTRACT

Recently a large amount of new chord annotations have been made available. This raises hopes for further development in automatic chord estimation. While more data seems to imply better performance, a major challenge however, is the wide variety of genres covered by these new data. As a result, the genre-independent training scheme as is common today is bound to fail. In this paper we investigate various options for exploring genre information for chord estimation, while also maximally exploiting the full dataset. More specifically, we propose a hyper-genre training scheme in which each genre cluster has its own parameters, tied together by hyper parameters as a Bayesian prior. The results are promising, showing significant improvements over other prevailing training schemes.

1. INTRODUCTION

Identifying musical chords from audio recordings is a challenging task and has recently attracted the interest of many researchers in the *music information retrieval* (MIR) field. The general approach of *automatic chord estimation* (ACE) involves two stages: the extraction of spectral features such as chromagram from audio; and the estimation of chords based on these features, via e.g. *Hidden Markov Models* (HMMs). In the past few years, while developing chroma extraction techniques has become a fruitful topic [2, 7–9], researchers have also explored a variety of musical factors such as key [5, 6, 10] and bassline [7, 9] that are related to chord progressions to build up richer ACE systems.

Nevertheless, one issue that has cramped the development in automatic chord estimation is the limited amount of the data available. Since most of the studies so far were carried out on a collection of The Beatles, Queen and Zweieck songs (i.e. the MIREX dataset), it is becoming increasingly probable that the existing ACE researches are overfitting this dataset. Recently a large amount of new chord annotations have been released by the *structural analysis of large amounts of music information* (SALAMI) project [14], raising hopes for further development of the ACE systems. A major challenge it also brings however, is the wide variety of genres covered by the data.

This paper is devoted to the study of the new dataset. Distinct from feature extraction and decoding research, we investigate various *training schemes* for exploring genre information to aid automatic chord estimation. We begin by giving an overview of the ACE task.

1.1 Automatic Chord Estimation

Let $\mathbf{x} = [x_1, \ldots, x_s, \ldots, x_S]$ be a mono audio signal with x_s indicating the value of the s-th sample. In ACE research, the signal is usually converted into a 12-dimensional representation of the harmonic content, with one such vector for each time frame. This vector is known as a *chroma* [2] vector, and it is intended to reflect the distribution of salience over the 12 pitch classes. The chroma vectors for the audio signal $\mathbf{x}$ are then gathered as the columns of a matrix $\mathbf{X} \in \mathbb{R}^{d \times T}$, with T denoting the number of frames and $d = 12$. In the target domain the chord annotations are denoted by $\mathbf{c} \in \mathcal{A}^{1 \times T}$, with $\mathcal{A}$ representing the chord alphabet. To model the relationship between observed variables $\mathbf{X}_t$ and hidden variables c_t, a standard HMM [13] with the parameter set $\Theta = \{\mathbf{P}_i \in \mathbb{R}^{|\mathcal{A}|}, \mathbf{P}_t \in \mathbb{R}^{|\mathcal{A}| \times |\mathcal{A}|}, \mathbf{P}_e\}$ is commonly used. $\mathbf{P}_i$, $\mathbf{P}_t$ denote the initialization and the transition probabilities respectively, and the emission probability for chord c_t is frequently modelled as a single Gaussian

$$p_e(\mathbf{X}_t | c_t) = \mathbf{X}_t \sim \mathcal{N}(\boldsymbol{\mu}^{c_t}, \boldsymbol{\Sigma}^{c_t}) \tag{1}$$

with the distribution parameters $\{\boldsymbol{\mu}^c, \boldsymbol{\Sigma}^c\}_{c \in \mathcal{A}}$. Under this framework, the joint probability of the feature vectors $\mathbf{X}$ and the corresponding chord sequence $\mathbf{c}$ is of the form

$$P(\mathbf{X}, \mathbf{c} | \Theta) = p_i(c_1) \prod_{t=2}^{T} p_t(c_t | c_{t-1}) \prod_{t=1}^{T} p_e(\mathbf{X}_t | c_t). \tag{2}$$

Given the optimal parameters Θ^*, the ACE task is equivalent to finding $\mathbf{c}^*$ that maximizes the joint probability $\mathbf{c}^* = \arg\max_{\bar{\mathbf{c}}} P(\mathbf{X}, \bar{\mathbf{c}} | \Theta^*)$, which can be done efficiently by the Viterbi algorithm [13].

By restricting our interest to a standard HMM, the *training scheme* as we define it is a strategy to derive the optimal parameters Θ^* from the training data. Given N audio clips for which the chromagrams $\mathcal{X} = \{\mathbf{X}^n \in \mathbb{R}^{d \times T_n}\}_{n=1}^{N}$ and

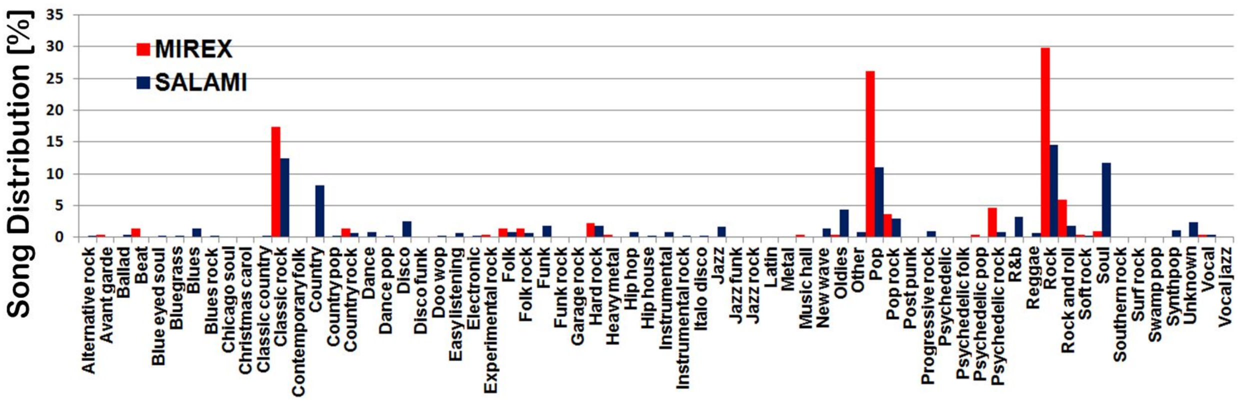


Figure 1. Genre distribution of the MIREX and the SALAMI datasets. The MIREX dataset is dominated by Rock and Pop genres, whereas the SALAMI dataset has a much wider variety of genres such as Country and Blues/Soul.

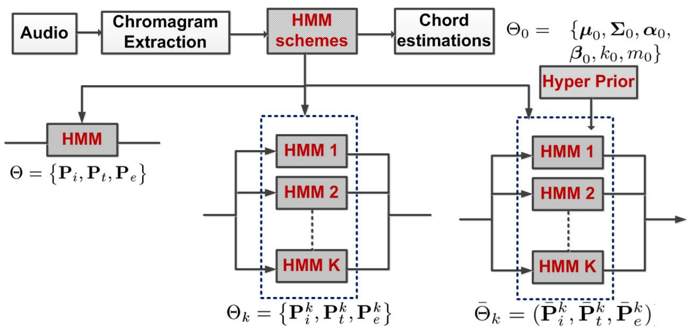

Audio
Chromagram Extraction
HMM schemes
Chord estimations
$\Theta_0 = \{\mu_0, \Sigma_0, \alpha_0, \beta_0, k_0, m_0\}$
Hyper Prior
HMM
$\Theta = \{\mathbf{P}_i, \mathbf{P}_t, \mathbf{P}_e\}$
HMM 1
HMM 2
HMM K
$\Theta_k = \{\mathbf{P}_i^k, \mathbf{P}_t^k, \mathbf{P}_e^k\}$
HMM 1
HMM 2
HMM K
$\bar{\Theta}_k = (\bar{\mathbf{P}}_i^k, \bar{\mathbf{P}}_t^k, \bar{\mathbf{P}}_e^k)$

Figure 2. Training schemes for automatic chord estimation: universal training (left), genre-specific training (middle) and the proposed hyper-genre training (right).

the chord annotations $\mathcal{C} = \{\mathbf{c}^n \in \mathcal{A}^{1\times T_n}\}_{n=1}^N$ are both available, the prevailing scheme is the *universal training* (denoted by UN, cf. left block in Fig. 2) that derives one Θ^* from all available data $\{\mathcal{X}, \mathcal{C}\}$.

1.2 Why a new training scheme?

The effectiveness of UN-training on the MIREX dataset has now been established. Since this collection is highly genre-biased and only small variations exist (cf. Fig. 1), UN-training can make full use of the data without confounding chord characteristics. However, the SALAMI data has a much wider variety of genres [1] (cf. Fig. 1), this casts doubts on the effectiveness of UN-training for two reasons: first of all, when reducing the chords to an alphabet such as triads (cf. Fig. 3), the variety of chords is noticeably different between genres. For instance, Rock and Country genres mainly use simple chords (e.g. major), whereas Jazz tends to uses more complex ones (e.g. major 7th). This subsequently incurs a large variation on the chromas of the same chord among different genres (cf. Fig.4). In addition, chord progressions of different genres can vary dramatically [11]. Neither of these can be solved by UN-training, since the scheme ignores the connection between musical genre and chord variety and progression.

[1] The genre information was obtained with thanks from `http://www.last.fm/` and `http://www.wikipedia.org/`.

A potential solution to these issues is to apply a more complex model such as a Mixture of Gaussians (MOG) to Eqn. (1), but this risks the probability functions of different chords being confused [e.g. E:dim= (E, G, Bb) may also yield high probability in an MOG model for C:maj due to the modelling of C:7= (C, E, G, Bb)]. One can also respect this musical factor with a more rigorous approach: training a different Θ for each genre, an example of which is the *genre-specific training* (denoted by GS, cf. middle block in Fig.2) presented in [5]. However, this method often suffers from the problems caused by data sparseness, because it can not maximally exploit the full dataset (see the discussion in Sec. 3.1).

As an alternative, we develop a new training scheme which we call *hyper-genre training* (denoted by HG, cf. right block in Fig. 2). Instead of using independent parameters for each genre as it is in GS-training, HG-training constructs a hierarchical probabilistic model and connect the genres using hyper parameters. This framework is similar to a Hierarchical Dirichelet Process (HDP) [15], which has been applied to music similarity measurement [12] and timbral similarity estimation [4] in the MIR domain. The main difference here is that the proposed approach uses a well-defined cluster structure on the basis of musical knowledge, hence avoiding the massive sampling and uncertain clustering process in HDP.

The rest of the paper is organized as follows. In Sec. 2 we apply the proposed HG-training to the standard HMM and detail the corresponding parameter estimation. We then evaluate the approach and compare it with the other two training schemes in Sec. 3. Finally the conclusions and future work are drawn in Sec. 4.

2. HYPER-GENRE HMM

To take into account the fact that songs belong to different genres, the data is divided into K clusters according to the genre information. Suppose the k-th cluster contains n_k songs $\sum_{k=1}^K n_k = N$, we then denote the collection of chromagrams and chord annotations for cluster k as $\mathcal{X}_k = \{\mathbf{X}^n \in \mathbb{R}^{d\times T_n}\}_{n=1}^{n_k}$, and $\mathcal{C}_k = \{\mathbf{c}^n \in \mathcal{A}^{1\times T_n}\}_{n=1}^{n_k}$.

In general, a UN-HMM trains one Θ on all available

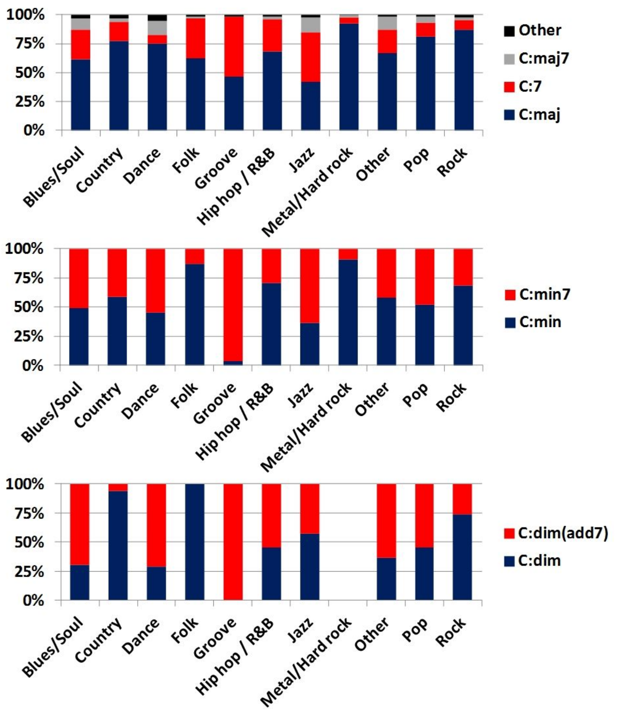


Figure 3. Comparison of chord varieties of chord C between different genres. Each figure includes the chords that can be reduced to the same triad. The percentages of these chords in a cluster then suggest the genre-specific chord variety. Note that in this figure the genres appeared in the SALAMI dataset have been grouped manually and formed 11 genre clusters.

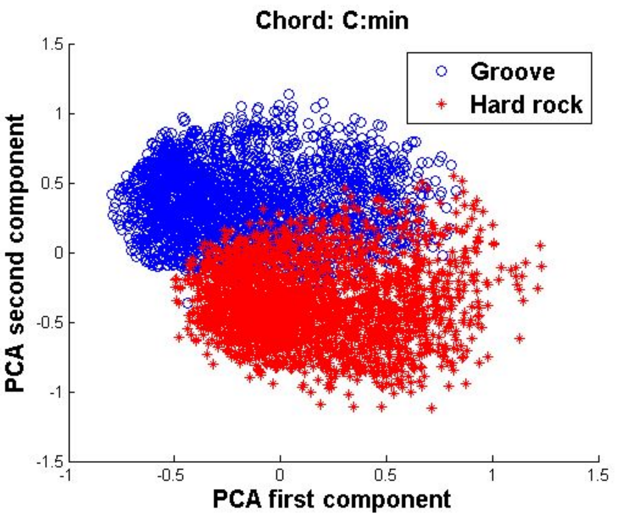


Figure 4. Chroma features for all occurrences of C:min-like chords (cf. middle plot in Fig. 3) for the Groove and the Hard rock genres. To aid visualization, the 12 dimensional feature space has been reduced to 2 using Principal Component Analysis. We found that in the Groove cluster, there were many more complex variants (e.g. C:min7), whilst in Hard rock simple chords such as C:min were more common. Owing to this, there is a large variation between their chroma features.

data $\{\mathcal{X},\mathcal{C}\}$; while a GS-HMM has a set of parameters $\mathbf{\Theta} = \{\Theta_k\}_{k=1}^{K}$, each of which is trained on the cluster examples $\{\mathcal{X}_k,\mathcal{C}_k\}$. The HG-HMM also has a set of genre-specific parameters $\bar{\mathbf{\Theta}} = \left\{\bar{\Theta}_k = (\bar{\mathbf{P}}_i^k, \bar{\mathbf{P}}_t^k, \bar{\mathbf{P}}_e^k)\right\}_{k=1}^{K}$, but it ties them together by hyper parameters as a Bayesian prior.

One implementation of the HG-HMM is depicted in Fig. 5, in which the genre clusters are connected via a hyper parameter set $\Theta_0 = \{\boldsymbol{\mu}_0 \in \mathbb{R}^{d\times|\mathcal{A}|}, \mathbf{\Sigma}_0 \in \mathbb{R}^{d\times d\times|\mathcal{A}|}, \boldsymbol{\alpha}_0 \in \mathbb{R}^{|\mathcal{A}|}, \boldsymbol{\beta}_0 \in \mathbb{R}^{|\mathcal{A}|\times|\mathcal{A}|}, \mathbf{k}_0 \in \mathbb{N}^K, m_0\}$ to propagate information. Under this framework, the parameter estimates of $\bar{\mathbf{\Theta}}$ are computed as follows.

2.1 Parameter Estimation

For the emission probability $\bar{\mathbf{P}}_e^k$ of cluster k, the hyper link (dash line in Fig. 5) is equivalent to applying a conjugate prior to the distribution parameters $\{\boldsymbol{\mu}_k^c, \mathbf{\Sigma}_k^c\}_{c\in\mathcal{A}}$:

$$\begin{array}{l}\mathbf{\Sigma}_k^c \sim \mathcal{W}(\mathbf{\Sigma}_0^c, m_0)\\ \boldsymbol{\mu}_k^c|\mathbf{\Sigma}_k^c \sim \mathcal{N}(\boldsymbol{\mu}_0^c, \frac{1}{k_0^c}\mathbf{\Sigma}_k^c)\end{array} \forall c \in \mathcal{A}, \tag{3}$$

where $\mathcal{W}$ denotes the Wishart distribution [1]. The Bayesian update of the emission probability then becomes

$$\begin{array}{rl}\bar{p}_e^k(\mathbf{X}_t|c) = & \mathbf{X}_t \sim \mathcal{T}(m_0 + m_k^c - d + 1,\\ & \bar{\boldsymbol{\mu}}_k^c, \dfrac{k^c+1}{k^c(m_0+m_k^c-d+1)}\bar{\mathbf{\Sigma}}_k^c).\end{array} \tag{4}$$

In (4) $\mathcal{T}$ denotes the multivariate Student-t distribution with the following parameters

$$\begin{array}{l}m_k^c = \#(c_t = c), \forall c_t \in \mathcal{C}_k,\\ k^c = k_0^c + m_k^c,\\ \bar{\boldsymbol{\mu}}_k^c = \frac{k_0^c\boldsymbol{\mu}_0^c + m_k^c\boldsymbol{\mu}_k^{*c}}{k_0^c + m_k^c},\\ \bar{\mathbf{\Sigma}}_k^c = \mathbf{\Sigma}_0^c + m_k^c\mathbf{\Sigma}_k^{*c} + \frac{k_0^c m_k^c}{k^c}\|\boldsymbol{\mu}_k^{*c} - \boldsymbol{\mu}_0^c\|^2,\end{array} \tag{5}$$

where $\#$ indicates 'the number of' and $\{\boldsymbol{\mu}_k^{*c}, \mathbf{\Sigma}_k^{*c}\}_{c\in\mathcal{A}}$ are the maximum likelihood (ML) estimations of the parameters $\{\boldsymbol{\mu}_k^c, \mathbf{\Sigma}_k^c\}_{c\in\mathcal{A}}$ using the cluster examples $\{\mathcal{X}_k, \mathcal{C}_k\}$.

Similarly, the hyper priors applied to the initialization and the transition parameters of cluster k are given by

$$\begin{array}{l}\mathbf{P}_i^k|\boldsymbol{\alpha}_0 = \{p_i^k(c)|\, c \in \mathcal{A}\} \sim \mathrm{Dir}(|\mathcal{A}|, \boldsymbol{\alpha}_0),\\ \mathbf{P}_t^k|\boldsymbol{\beta}_0 = \{p_t^k(c|\bar{c})|\, c \in \mathcal{A}\} \sim \mathrm{Dir}(|\mathcal{A}|, \boldsymbol{\beta}_0^{\bar{c}}), \forall \bar{c},\end{array} \tag{6}$$

where Dir is the Dirichlet distribution. The Bayesian update of these probabilities are then computed by: $\forall c_1, c_{t-1}, c_t \in \mathcal{C}_k$

$$\begin{array}{l}\bar{p}_i^k(c) = \dfrac{\#(c_1=c)+\alpha_0^c}{\sum\limits_{c'\in\mathcal{A}}\#(c_1=c')+\sum\limits_{c'\in\mathcal{A}}\alpha_0^{c'}},\\ \bar{p}_t^k(c|\bar{c}) = \dfrac{\#(c_t=c\,\&\,c_{t-1}=\bar{c})+\beta_0^{\bar{c},c}}{\sum\limits_{c'\in\mathcal{A}}\#(c_t=c'\,\&\,c_{t-1}=\bar{c})+\sum\limits_{c'\in\mathcal{A}}\beta_0^{\bar{c},c'}}.\end{array} \tag{7}$$

Note that if a non-informative prior is used (i.e. $\alpha_0^c = 1$ and $\beta_0^{\bar{c},c} = 1$), the Bayesian update (7) is the ML estimations of the initialization and the transition parameters trained on the cluster examples. This is equivalent to using $\{\mathbf{P}_i^k, \mathbf{P}_t^k\}_{k=1}^K$ as used in the GS-HMM.

Eqns. (5) and (7) provide the insight of the hyper parameters: they reflect our prior belief about the genre cluster parameters. Hence when few data are available to estimate the parameters of cluster k, the HG-HMM can benefit

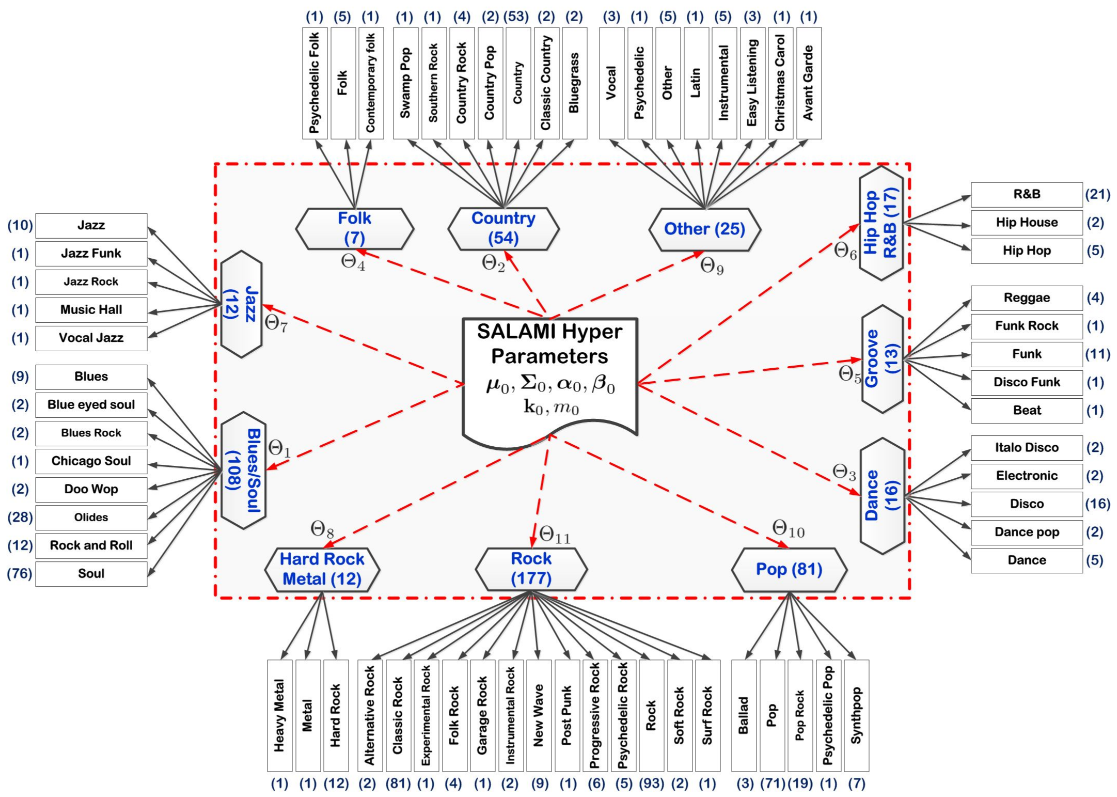


Figure 5. The implementation (dash-dot box) of the hyper-genre HMM for the SALAMI dataset. Ideally each genre should be regarded as a cluster, but in practice this is difficult to achieve with limited data. In order to assure a reasonable cluster size, we grouped the genres and created 11 genre-related clusters. These clusters are then connected via the hyper parameters so as to share information. The number of songs in a genre (or cluster) is shown in the bracket.

from the hyper parameter set. The clusters (e.g. Rock) can also share information with related ones (e.g. Blues), while retaining their intrinsic chord varieties and progressions. Ideally the set Θ_0 should has several subsets, one for each group of related genres. However, restricted by the data available we had to follow the suggestions in [4, 12] and used a single Θ_0 that reflects the distribution of the whole dataset instead: that is, $(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ are set to the mean and the covariance matrices of all training data respectively; $\boldsymbol{\alpha}_0, \boldsymbol{\beta}_0$ counts for all chord initializations and transitions, $m_0 = d$ and $k_0^c = \#(c_t = c), \forall c_t \in \mathcal{C}$.

2.2 Decoding

Given the updated parameters $\{\bar{\Theta}_k\}_{k=1}^K$, the decoding process is given by

$$\begin{aligned}\mathbf{c}^* &= \arg\max_{\bar{\mathbf{c}}} P(\mathbf{X}, \bar{\mathbf{c}}|\bar{\Theta}_k) \\ &= \arg\max_{\bar{\mathbf{c}}} \bar{p}_i^k(c_1) \prod_{t=2}^{T} \bar{p}_t^k(c_t|c_{t-1}) \prod_{t=1}^{T} \bar{p}_e^k(\mathbf{X}_t|c_t)\end{aligned} \tag{8}$$

The decoder (8) requires the cluster label k of the test example. This requirement can be easily waived by using the maximum likelihood inference as suggested in [5]:

$$\{\mathbf{c}^*, k^*\} = \arg\max_{\bar{\mathbf{c}}, k} P(\mathbf{X}, \bar{\mathbf{c}}|\bar{\Theta}_k). \tag{9}$$

3. EXPERIMENTS

Here we describe the main experiments conducted. The dataset investigated is the SALAMI data, which contains 522 songs along with the ground truth chord annotations [2]. In the experiments, we restrict the ACE system to a standard HMM with the loudness based chromagram [9]. In order to capture intrinsic chord varieties and progressions between genres but retaining a controllable complexity, we restricted ourselves to an alphabet of triads [3], with 73 unique chords in total. To evaluate the proposed approach, we randomly split 2/3 of songs from each genre cluster to form the training set, while the remaining 1/3 were used for testing. The frame-based chord estimation accuracy is used as the evaluation metric and in total 102 train-test runs were done to assess variance [4].

3.1 Comparison of different training schemes

There are three training schemes we can apply to the emission parameters: universal training (UN), genre-specific training (GS) and hyper-genre training (HG). Similarly, they can be applied to the initialization and the transition parameters, resulting in $3 \times 3 = 9$ combinations. Suppose the

[2] The readers are referred to the Appendix for our process of extracting the SALAMI chord annotations. These annotations are available online at `https://patterns.enm.bris.ac.uk/files/SALAMI_522_chord_annotations.zip`.

[3] Chord types: maj, min, dim, sus2, sus4, aug and N.

[4] That is, each song in the dataset would be tested 34 times.

cluster labels are known, in this subsection we compared these combinations using the decoder (8).

Table. 1 presents the overall performances for each combination, from which we observed a consistent improvement of the hyper-genre training over the other schemes. In particular, the HG-GS combination achieves the best performance, amounting to 9.3% and 14.1% reductions on the error rate compared with the universal (UN-UN) and the genre-specific (GS-GS) trainings respectively. Although applying a transition prior learnt from all the data still yields better results than the other schemes, it is worse than merely using a non-informative one. We postulate this is because a transition prior learnt from all examples is a mixed chord progression of all genres, applying it would inevitably confound some typical progressions such as "I (tonic) - V (dominant) - IV (subdominant)" commonly seen in the Blues genre. This suggests that further improvement might be gained by applying musical knowledge based transition priors on different genres, which can be obtained from e.g. the synthetic MIDI data used in [5].

E\T	UN	GS	HG
UN	$\boxed{63.61} \pm 1.31$	64.39 ± 1.29	64.3 ± 1.30
GS	60.61 ± 1.60	$\boxed{61.56} \pm 1.68$	61.05 ± 1.67
HG	65.93 ± 1.17	$\mathbf{66.98} \pm 1.21$	66.47 ± 1.19

Table 1. Performances $[\%]$ of different scheme combinations on the SALAMI dataset (best result in bold). The vertical axis shows the schemes applied to the emission parameters and the horizontal axis shows that to the initialization/transition parameters. The improvement of the HG-HMM with non-informative priors (HG-GS) is significant at a level $< 10^{-34}$ over the performances of the other scheme combinations under a paired t-test.

Figure 6 further depicts the performances of different training schemes on each genre cluster. For GS-training, the performance gradually increases when more examples are available for a cluster. The only exception is Groove, possibly due to the fact that the complex chords in this cluster are difficult to estimate. Limited by the amount of the data available for each cluster, GS-training is generally inferior to UN-training. This problem was not experienced or explored in [5] (Sec. 4.3), since their experiments were based on synthetical MIDI data such that the GS-HMM had sufficient examples to train the parameters. Alternatively, HG-training is slightly worse than UN-training when the data is very limited, by means of sharing information from the hyper parameters. For other clusters such as Rock, HG-training allows it to obtain information from related clusters such as Blues/Soul, while retaining the genre-specific chord variety and progression. This benefit makes it outperform UN-training and improve the performance by an absolute 4.4%.

3.2 Bypassing the genres

In practice the genre information of the test data is unknown, hence there is no choice but to use a genre-independent

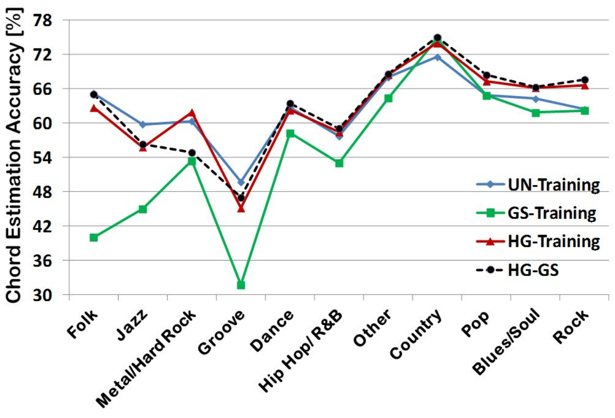


Figure 6. Performances of the universal (UN-UN), the genre-specific (GS-GS), the hyper-genre (HG-HG) and the combined (HG-GS) training schemes on each cluster. The clusters from left to right are sorted by the number of examples in the clusters.

model (e.g. UN-training); or increase the model complexity and infer the genre as well. In this subsection we investigate how much we can gain by inferring the genre with maximum likelihood technique (9). The experiment setup was the same as that in Sec. 3.1, and we compared the following models with the ones presented in Tab. 1:

1) The GS-HMM (using the schemes GS-GS) with the decoder (9). This is the model suggested in [5] (Sec. 4.3).

2) The HG-HMM (using HG-GS) with the decoder (9).

In addition to chord estimation accuracies, the genre prediction accuracies of the models are also evaluated.

Tab. 2 shows the results and we observed a mild decrease in performance when compared with the same models using genre information in Tab. 1. This reflects the benefit obtained from the genre information used. However, the performances of using ML inference are very close to that of using genre information directly, suggesting that this technique is reliable to use when the genre for a test example is unknown.

It is worth pointing out that the genre prediction accuracies are very low for both models, probably for two reasons. Firstly, since the numbers of examples in different clusters are highly imbalanced, some clusters might not have enough data to train the parameters (e.g. Folk and Jazz). In this case, a test example from that cluster could be better decoded by other cluster models. Although applying the hyper parameters can ease this problem and improve the chord estimation accuracy, it makes the cluster models closer to the hyper prior and inevitably confounds their intrinsic chord varieties. This consequently worsens the genre prediction accuracies.

Another potential explanation is that since the genre clusters are highly overlapped, it is very difficult to rigorously classify an example into just one cluster. This seems to suggest using a more flexible forest structure (e.g. the genre Country Rock should be connected to both Country and Rock clusters) instead of the directed tree depicted in Fig. 5, which will be investigated in our future work.

	GS-GS	HG-GS
C-Acc	61.27 ± 1.50	$\mathbf{66.83} \pm 1.27$
G-Acc	$\mathbf{16.05} \pm 2.98$	10.52 ± 1.81

Table 2. Performances [%] of different models on the SALAMI dataset (best result in bold). C-Acc denotes the frame-based chord estimation accuracy and G-Acc is the genre prediction accuracy.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a new training scheme – *hyper-genre training* for automatic chord estimation, capable of testing on multiple varied genres. The principle is to construct a hierarchical probabilistic model and connect the genre clusters using hyper parameters. Compared with the prevailing universal training scheme, HG-training is able to retain chord variety and progression characteristics of musical styles. Compared with genre-specific training, HG-training can benefit from the hyper prior and it resolves the problem of data sparseness often encountered in real world data. Both benefits have been verified in our experiments on a large and varied chord annotation dataset, where HG-training achieved significant improvements over the other two schemes.

For future work, we aim to improve the hierarchical structure of the proposed approach. This can be done by employing a more flexible forest structure instead of the directed tree graph. An alternative direction of research is to learn such hierarchical structure from the data automatically, which might lead to a more robust and powerful ACE system. Finally, we are also interested in how incorporating musical knowledge based transition priors may improve chord estimation accuracy.

5. APPENDIX: EXTRACTING CHORD ANNOTATIONS FOR THE SALAMI DATASET

In this appendix we summarize how we extracted the ground truths from the SALAMI chord annotation files.

There are two processes: obtaining the chord labels and inferring the durations. For the former, we followed the description in [14] and parsed the chord labels with the C. Harte's chord parser [3]. There were several exceptions, which we revised manually. A more difficult task is to extract chord durations. The SALAMI chord annotation files do not offer time stamps for each chord (as used in the MIREX annotation files). Instead, time instances are given over multiple bars, where each bar contains one or more chord. Our assumption is that the bars between two time stamps would have equal durations (if they are not in the same meter, then the durations are adjusted according to the meter). Under this assumption we extracted chord durations from the annotation files.

The original SALAMI dataset contains 649 songs, from which we found 21 songs having ambiguous tuning. Additionally there are some duplications and cover songs. For our experiments, we removed the 21 songs and the duplicate and cover songs so that each unique song only appeared once in the dataset. After this process we obtained a set of 522 songs.

6. REFERENCES

[1] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] T. Fujishima. Real time chord recognition of musical sound: a system using common lisp music. In *Proc. of ICMC*, pages 464–467, 1999.

[3] C. Harte, M. Sandler, and S. Abdallah. Symbolic representation of musical chords: a proposed syntax for text annotations. In *Proc. of ISMIR*, pages 66–71, 2005.

[4] M. Hoffman. *Probabilistic graphical models for the analysis and synthesis of music audio*. PhD thesis, Princeton University, 2010.

[5] K. Lee. *A system for acoustic chord transcription and key extraction from audio using hidden Markov models trained on synthesized audio*. PhD thesis, Stanford University, 2008.

[6] K. Lee and M. Slaney. Acoustic chord transcription and key extraction from audio using key-dependent hmms trained on synthesized audio. *The IEEE Transactions on Audio, Speech and Language Processing*, 2008.

[7] M. Mauch. *Automatic chord transcription from audio using computational models of musical context*. PhD thesis, Queen Mary University of London, 2010.

[8] M. Müller and S. Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Trans. Audio, Speech, Lang. Process.*, 18(3):649–662, 2010.

[9] Y. Ni, M. Mcvicar, R. Santos-Rodriguez, and T. De Bie. An end-to-end machine learning system for harmonic analysis of music. *IEEE Trans. Audio, Speech, Lang. Process.*, 20(5), 2012.

[10] K. Noland and M. Sandler. Key estimation using a hidden Markov model. In *Proc. of ISMIR*, 2006.

[11] W. Piston. *Harmony*. Norton, New York, 1978.

[12] Y. Qi, J. Paisley, and L. Carin. Music analysis using hidden markov mixture models. *IEEE Transactions on Signal Processing*, 55(11):5209–5224, 2007.

[13] L. R. Rabiner. A tutorial on hidden Markov models and selected application in speech recognition. In *Proc. of the IEEE*, 1989.

[14] J. Smith, J. Burgoyne, I. Fujinaga, D. Roure, and J. Downie. Design and creation of a large-scale database of structural annotations. In *Proc. of ISMIR*, 2011.

[15] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2007.

SEMI-SUPERVISED NMF WITH TIME-FREQUENCY ANNOTATIONS FOR SINGLE-CHANNEL SOURCE SEPARATION

Augustin Lefèvre
INRIA team SIERRA
augustin.lefevre@inria.fr

Francis Bach
INRIA team SIERRA
francis.bach@ens.fr

Cédric Févotte
LTCI/Telecom ParisTech
fevotte@telecom-paristech.fr

ABSTRACT

We formulate a novel extension of nonnegative matrix factorization (NMF) to take into account partial information on source-specific activity in the spectrogram. This information comes in the form of masking coefficients, such as those found in an ideal binary mask. We show that state-of-the-art results in source separation may be achieved with only a limited amount of correct annotation, and furthermore our algorithm is robust to incorrect annotations. Since in practice ideal annotations are not observed, we propose several supervision scenarios to estimate the ideal masking coefficients. First, manual annotations by a trained user on a dedicated graphical user interface are shown to provide satisfactory performance although they are prone to errors. Second, we investigate simple learning strategies to predict the Wiener coefficients based on local information around a given time-frequency bin of the spectrogram. Results on single-channel source separation show that time-frequency annotations allow to disambiguate the source separation problem, and learned annotations open the way for a completely unsupervised learning procedure for source separation with no human intervention.

1. INTRODUCTION

During the past decade, nonnegative matrix factorization (NMF) has become the core algorithm in single-channel source separation. A rich literature has been developed to adapt NMF to difficult scenarios in which sources are highly synchronized, and little or no development data is available.

In the past years, intensive research on Bayesian modelling and parameterized methods have been conducted to improve the identifiability of basis elements by restricting the complexity of the estimated model. More recently, another category of contributions consider incorporating information that is directly relevant to the data at hand, and specified by the user. In [2], time activation of the sources is used to specify direct constraints on the activation coefficients of the decomposition. Pitch estimates [5] were used for lead voice extraction. In [8], detailed score information is provided so that each individual note can be separated. While these contributions may use different NMF models, a common trait is that user information is used to specify the support of decomposition coefficients at the coding stage. A quite different line of work is proposed in [1, 3], where isolated signals are used as proxy for the source signals, so that information on both the basis functions and the activation coefficients can be used to constrain the factorization.

In this paper, we propose to annotate directly the time-frequency representation that is used to perform source separation. We assume that we are given recordings where a large fraction of time-frequency bins of the spectrogram may be assigned unambiguously to one dominant source. This hypothesis holds as long as there are not too many sources, and post-processing of the recording does not involve heavily non-linear effects. As illustrated in Figure 1, some patches in the spectrogram are cues for source-specific activity, which may be exploited as information on the optimal binary mask.

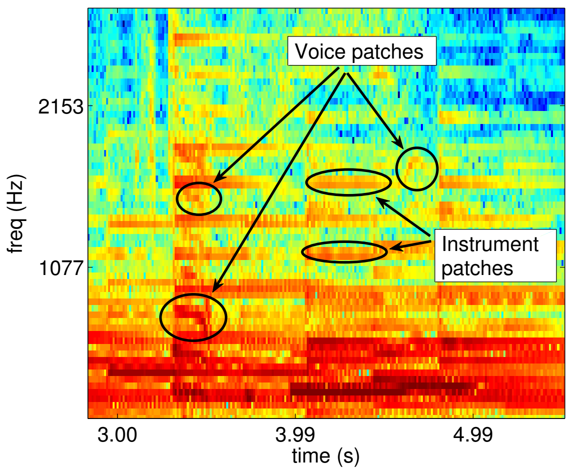


Figure 1: Cues from computational audio source analysis may be used as information on the optimal masking coefficients

In this article we make three contributions : we propose in Section 2 a novel modification of NMF (semi-supervised NMF) to take into account time-frequency annotations of the spectrogram, that is robust to errors in the annotations. In Section 2.2, we present a graphical user interface to retrieve such time-frequency annotations. In Section 3, we propose supervised learning algorithms to automatize annotations, and explain how to combine them with semi-supervised NMF. Finally, we illustrate our contributions on publicly available source separation databases in Section 4.

2. SEMI-SUPERVISED NMF

2.1 Model and interpretation

In this section we propose a novel modification of NMF to incorporate annotations in the spectrogram. Let us first briefly summarize our NMF model and introduce mathematical notations, before proceeding to the main part of the contribution.

Given the short time Fourier transform of a signal $X \in \mathbb{C}^{F \times N}$ (in the following f indexes frequency and n time), we assume that $X = \sum_g S^{(g)}$, where $S^{(g)} \in \mathbb{C}^{F \times N}$ is the spectrogram of each source signal for $g \in \{1, \ldots, G\}$. Define the power spectrograms of the sources $V_{fn}^{(g)} = |S^{(g)}|^2$. They are assumed to follow a linear model : $V_{fn}^{(g)} = \sum_{k=1}^{K_g} W_{fk}^{(g)} H_{kn}^{(g)}$, where $W^{(g)} \in \mathbb{R}_+^{F \times K_g}$, $H^{(g)} \in \mathbb{R}^{K_g \times N}$. Define $K = \sum_g K_g$, $W = (W^{(1)}, \ldots, W^{(G)}) \in \mathbb{R}_+^{F \times K}$ and $H^\top = ((H^{(1)})^\top, \ldots, (H^{(G)})^\top) \in \mathbb{R}_+^{K \times N}$. Then, depending on the assumed distribution of $S^{(g)}$, estimation of W and H amounts to minimizing $d(V, WH)$ where d is a measure of fit between data and the underlying model. In this article we will use the Itakura-Saito divergence, but actually any β-divergence may be used.

Given estimates $\hat{V}_{fn}^{(g)}$ of the power spectrogram of each source, time domain estimates of the sources are then computed by Wiener filtering, where the Wiener coefficients of the source in the time-frequency domain are given by : $M_{fn}^{(g)} = \frac{\hat{V}_{fn}^{(g)}}{\hat{V}_{fn}}$.

The key idea in our contribution is the following : suppose we have at hand a set $\mathcal{L}$ of annotated time-frequency bins and a set of time-frequency masks $M_{fn}^{(g)}$ such that : $M_{fn}^{(g)} \in [0, 1]$, and $\sum_g M_{fn}^{(g)} = 1$ if $(f, n) \in \mathcal{L}$, $\sum_g M_{fn}^{(g)} = 0$ otherwise.

For annotated time-frequency bins, we define target values for each source spectrogram : $\tilde{V}_{fn}^{(g)} = M_{fn}^{(g)} V_{fn}$.

The remaining, un-annotated entries of $\hat{V}$ are then computed so as to fit the observed spectrogram. This idea translates into the following optimization problem :

$$\min \sum_{(f,n)} d_{IS}(V_{fn}, \hat{V}_{fn}) + \lambda \sum_{\substack{(f,n) \in \mathcal{L} \\ g=1,\ldots,G}} \mu_{fn} d_{IS}(\tilde{V}_{fn}^{(g)}, \hat{V}_{fn}^{(g)}) , \tag{1}$$

where $d_{IS}(x, y) = \frac{x}{y} - \log \frac{x}{y} - 1$ is the Itakura-Saito divergence [1], and optimization is subject to the constraints that $W \geq 0$ (point-wise nonnegativity), $H \geq 0$, and $\sum_f W_{fk} = 1$ to avoid scaling ambiguity. We interpret the second term in Eq. (1) as a relaxed version of the constraints that $\hat{V}_{fn}^{(g)}$ be equal to their target value $M_{fn}^{(g)} V_{fn}$, for all annotated bins $(f, n) \in \mathcal{L}$.

[1] Given that some values are set to zero, we replace the IS divergence $d_{IS}(x, y)$ by $d_{IS}(\epsilon + x, \epsilon + y)$ (where $\epsilon = 10^{-7}$) in our optimization problem, in order to deal with ill-conditioning of the objective function.

We may tune the relative importance of annotation by varying parameter λ, from $\lambda = 0$ (standard NMF), to $\lambda \to +\infty$ (in which case $(WH)_{fn} = V_{fn}^{(g)}$ is enforced exactly if there are any feasible solutions). Thus, robustness to uncertainty in the annotations is introduced by replacing hard constraints by penalty terms in the NMF optimization problem. Note that since annotations dictate the assignment of components to sources, there is no need to group components by hand. We will discuss the role of μ_{fn} in the next section : in the case of user annotations, $\mu_{fn} = 1$. Let us discuss two cases :

(a) $\mathbf{M_{fn}^{(g)}} \in \{\mathbf{0}, \mathbf{1}\}$**:** this is the case of user annotations, where time-frequency bins are labelled by hand. In this case, there can be only one active source at each time-frequency bin, since $\sum_g M_{fn}^{(g)} = 1$. This is a strong assumption, which is verified for a large fraction of the mixtures that are found in blind source separation.

(b) $\mathbf{M_{fn}^{(g)}} \in [\mathbf{0}, \mathbf{1}]$**:** this general case is relevant to the learning procedures we introduce in the next section, since they output decision values in $[0, 1]$.

Discussing the algorithm is beyond the scope of this paper : we used a multiplicative updates algorithm with appropriate modifications to deal with the additional terms in Eq. (1) [6].

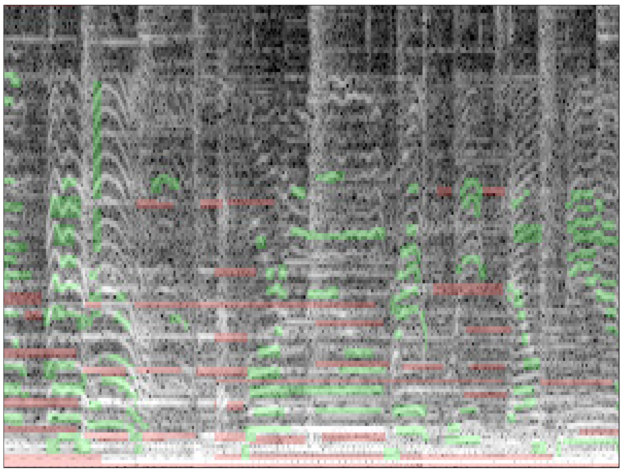


Figure 2: Example of user annotations in a ten seconds' audio track: green regions are assigned to voice, and red regions to accompaniment (**best seen in color**).

2.2 Relation with previous work

As in [2, 8, 5], annotations are used to constraint some sources to be inactive. In fact, time annotations are a special case of our model, where annotations are such that $M_{fn}^{(g)} = M_{f'n}^{(g)}$ for all (f, f') (i.e., zeroes come in columns). Our model deals with that case when there are two sources. The only difference between our model and [2] is that instead of enforcing $H_{kn} = 0$ as a hard constraint, we introduce a soft penalty to enforce $W_{fk} H_{kn} = 0$, with the added benefit that incorrect annotations are dealt with in a robust fashion. The case of more than two sources is dealt with a simple extension of Eq. (1), which we omit here for lack of space.

2.3 A graphical user interface for time-frequency annotation of spectrograms

In this section, we investigate manual annotation of the spectrogram. A GUI was designed in Matlab to annotate spectrograms (see Figure 2), with some extra sound functionalities to help the user. It takes sound files as input, applies some basic preprocessing (re-sampling at user-specified rate, down-mixing to mono), computes a time-frequency representation via user-specified parameters, and displays the spectrogram. Zooming and slide-rule navigation are enabled for better visualization. Annotation of sources is done with a simple rectangle drawing utility : one color for each source, as illustrated in Figure 2. Annotations are stored in an annotation mask of dimension $F \times N \times G$ (where (F, N) is the size of the spectrogram and G the number of sources). Several annotation masks may be loaded into memory and displayed alternatively, so the user can compare, for instance, manual annotations with the output of a blind source separation algorithm. Annotation masks may be exported to .mat format for further processing. Finally, we implemented playback functionalities to help the user annotate the spectrogram.

We designed the GUI to make the annotation process easier and faster : indeed, in our experience, while time annotations are easy and require only listening once or twice to the mix, time-frequency annotations are hard even for trained users : it takes up to one hour to annotate 20% of a twenty seconds track.

3. TOWARDS A SUPERVISED ALGORITHM FOR ANNOTATION

Research in computational audio scene analysis (CASA) has emphasized the role of frequency tracks in source identification : indeed by looking at a spectrogram, it is easy to assign a significant number of frequency tracks either to a voiced source or a musical source (see Figure 1). In previous works, such cues have been used to compute a similarity matrix that would then be used to perform clustering see [9, 4]. We propose here a supervised learning procedure to predict annotations automatically. At train stage, we have at hand separate sources so that we observe not only the mix, but also the Wiener coefficients $M_{fn}^{(g)}$ computed on the ground truth, while at test stage we only observe V. Thus, the goal is to predict $\mathbb{E}(M^{(g)}|V)$. In order to alleviate the computational burden [2], we make two restrictions on the learning procedure : each vector $(M_{fn}^{(1)}, \ldots, M_{fn}^{(G)})$ for a given time-frequency bin (f, n) is predicted independently of the others, and based only on the values of patches centered at that time-frequency bin.

We now introduce the features and algorithms used to train our predictor.

3.1 Features

The basic input to our learning algorithms consists in rectangular time-frequency blocks extracted from the input power spectrogram. The size of the rectangular blocks is fixed as a parameter of the algorithm. They are then normalized to have unit ℓ_1-norm so the features are scale invariant. We also considered taking the log of patches, adding coordinates of the patch as additional information, and taking a Gabor transform of the patches. The Gabor transform in particular was introduced so that correlations between pixels in each time-frequency blocks is taken into account. Finally, we also tried averaging the ground truth Wiener coefficients before learning, so that predicted regression surfaces are smoother in time-frequency space.

[2] indeed even for ten seconds' excerpts, there are more than 500×1000 time-frequency bins for standard STFT parameters

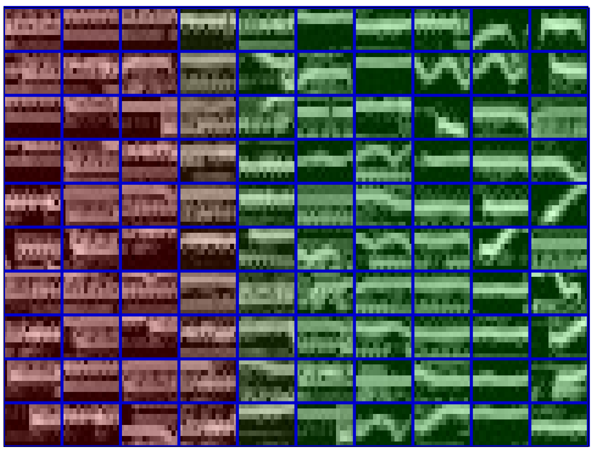


Figure 3: Samples of patches extracted from the SISEC database. Intensity reflects amplitude, patches which are labeled as accompaniment are in red, while patches which are labeled as voice are in green. Patches in brown have mixed Wiener coefficients (**best seen in color**).

3.2 Learning algorithms

Due to space limitation, we restrict ourselves to naming the algorithms we chose and highlighting the key parameters to tune. We refer the reader to standard textbooks on machine learning for more details (e.g., [7]).
K-nearest neighbors (knn): for each test point $x_i^{(test)}$, the C nearest points $x_j^{(train)}, j \in \{1, \ldots, C\}$ from the train set are used to predict $M_i^{(g)} = 1/C \sum_j M_j^{(g)}$.
Quantized knn (km): We learn C clusters from the train set using K-means; for each cluster, we compute average prediction coefficients $M_c^{(g)}$. For each test point, we predict $M_c^{(g)}$ from the nearest cluster c.
Random Forests (rf): We learn C regression trees of depth d from the train set and average over the C predictions for each test point.

We will refer to this supervised learning procedure as automatic annotations, no matter which algorithm is used.

3.3 Computation of μ_{fn} for automatic annotations

While the learning algorithms presented above predict Wiener coefficients, output values near 0.5 reflect uncertainty in the Wiener coefficients rather than prediction of mixed volumes. For this reason we introduce an additional tuning parameter μ_{fn} in Eq. (1), so that output values near 0.5

are less taken into account than values near $\{0,1\}$. As a rule, we choose $\mu_{fn} = 1 - \frac{G}{G-1}\sum_g M_{fn}^{(g)}(1 - M_{fn}^{(g)})$, so that $0 \leq \mu_{fn} \leq 1$ and $\mu_{fn} = 0$ if all $M_{fn}^{(g)}$ are equal. Moreover, when annotations are in $\{0,1\}$, we always have $\mu_{fn} = 1$.

4. EXPERIMENTAL RESULTS

4.1 Description of music databases

We used two publicly available databases in our experiments: the QUASI database [3] and the SISEC database for Professionally Produced Music Recordings [4]. All source tracks were down-sampled from 44100 Hz to 16000 Hz, and down-mixed to mono by taking the average of left and right channels. A voice track and accompaniment track are then created by aggregating the various source files, and then a final mix is created by summing the two tracks. Sine-bell windows of size 1024 with 512 overlap were used to compute short time Fourier transforms. The QUASI database contains longer tracks that are amenable to time annotations. The SISEC database contains short tracks where only time-frequency annotations can be used. Although detailed instrumental tracks are provided for most of the mixtures, we work only on single-channel signals. Since we are dealing with under-determined mixtures, we restrict ourselves to separating voice from accompaniment in each track, in order to alleviate the difficulty of the problem.

4.2 Ideal performance of semi-supervised NMF and robustness to wrong annotations

	SDR1	SDR2	SIR1	SIR2	SAR1	SAR2
0.1 %	-0.02	-0.60	5.15	5.16	3.62	2.33
1 %	0.70	0.24	4.59	6.25	4.39	2.85
10 %	6.71	6.68	13.57	16.53	7.95	7.40
100 %	10.40	10.41	19.88	20.88	11.00	10.88

Table 1: Mean results on the SISEC database, as the proportion of annotation increases.

Table 1 displays source separation results achieved by semi-supervised NMF on the SISEC database when fed with the actual Wiener coefficients computed from the ground truth sources. Source separation performance is measured by Source to Distortion Ratio (SDR), Source to Interference Ratio (SIR), and Source to Artefact Ratio (SAR). Higher values indicate better performance. As we can see, satisfactory results are obtained with as little as 10% of annotations. When 100% of annotations are given, NMF does nothing and the computed masks are simply the ideal Wiener coefficients computed from the sources.

We study the robustness of our NMF routine by replacing part of the ideal annotations by noise to simulate human errors. Table 2 displays average SDRs obtained when fixing the annotation rate to 10% and varying either the rate wrong annotations p or the optimization parameter λ. As expected, for fixed λ the average SDR drops as p increases. When p is fixed, there is an optimal value of λ that trades off the benefits and drawbacks of annotations. Fixing the target annotation rate to 10%, satisfactory results are obtained with up to 10% of wrong annotations (i.e.1% of the spectrogram).

λ	$p = 0$	$p = 0.05$	$p = 0.1$	$p = 0.2$	$p = 0.5$
10^{-1}	0.11	-0.08	-1.76	-1.47	-1.47
10^{0}	5.59	4.10	3.50	2.29	1.20
10^{1}	7.59	6.53	5.32	3.43	0.59
10^{2}	7.07	5.66	4.54	3.15	0.77

Table 2: Mean SDR value as λ and the proportion of wrong annotations vary. The proportion of annotations is set to 0.1

4.3 Automatic annotation : comparison of algorithms and experimental results

method	**mean error (% improvement)**
4 8 loggabor km avg	0.141 ±0.018 (14.9)
4 16 wcoords knn avg	0.140 ±0.015 (15.9)
4 8 wcoords knn avg	0.138 ±0.015 (16.8)
4 32 loggabor rf avg	0.137 ±0.013 (17.4)
4 32 loggabor knn avg	0.137±0.010 (17.4)

Table 3: Mean error on Wiener coefficient predictions on the SISEC database (% improvement over random prediction), for various learning strategies .

Learning algorithms were trained by dividing the SISEC database in two sets of tracks. For each set, we train detectors and test them on the other set. Thus we may compute annotations and run semi-supervised NMF for all tracks without the risk of overfitting. We emphasize the fact that each track is annotated with a detector that has never seen the spectrogram before : our method is purely supervised with no adaptation to test data. Parameters of the learning algorithms were selected at train stage by cross-validation. Time-frequency patches of size in $\{4,8\}\times\{8,16,32\}$ were extracted. Out of each track we extract 5×10^3 patches at train time, and 10^5 patches a test time, so approximately 10% of the track is annotated at test time when semi-supervised NMF is called.

We display in Table 3 the results of the best 5 detectors, in terms of mean prediction error (first column) and in terms of relative improvement over a purely random predictor. Detectors are named after the following rule : {patch size} {feature} {learning method} {averaging or identical}. For instance, the tag **loggabor** corresponds to taking log then Gabor transform of patches, and **wcoords** adding frequency coordinates of the patches as side information. Note that we used exact Wiener coefficients to compute errors, so that all detectors can be compared even when averaging was used at train stage. The improvement over a random predictor is consistent across the features and the algorithms that were used. Figure 4 compares annotations provided by the best detectors from Table 3 with

[3] `www.tsi.telecom-paristech.fr/aao/`
[4] `sisec.wiki.irisa.fr`

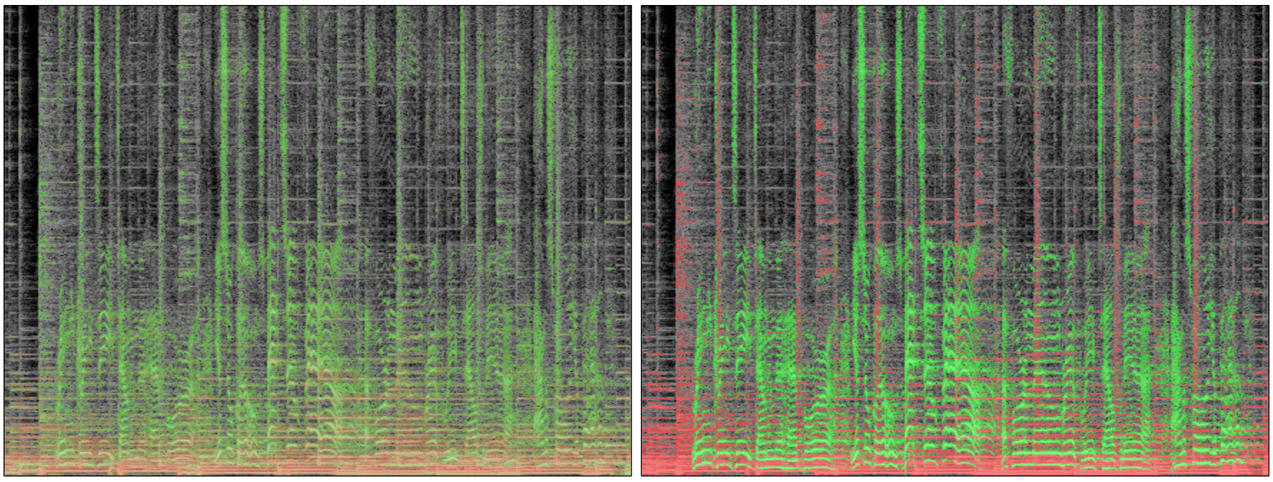

(a) Automatic (b) Correct

Figure 4: Comparison of automatic annotations and correct annotations (at the same time-frequency bins). Gray-scale time-frequency bins are not annotated, red bins are annotated as accompaniment, green bins as voice(**best seen in color**).

ideal annotations at the same points were automatic annotations were made. Red time-frequency bins correspond to accompaniment, and green to voice. The most striking observation is that, while ideal annotations are in very bright colors (few Wiener coefficients are different from 0 or 1), automatic annotations, on the other hand, are generally biased towards 0.5. This is to be expected since predicting 0.5 incurs a risk of losing at most 0.25 (since we use a regression loss), while predicting 0 or 1 incurs a maximum loss of 1. The main asset of automatic annotations is that pitch tracks with varying frequency are successfully predicted as voice. Automatic annotations are biased towards predicting voice in the higher frequencies : however the learning algorithm in this example did not have the information of frequency. This might be because transients “look” a lot like patches of unvoiced speech. Finally, one may spot inconsistencies in the predictions in the sense that points belonging to the same pitch tracks are sometimes classified incoherently, which is not surprising since the learning algorithms we have proposed predict time-frequency bins independently.

To sum up, predictions of Wiener coefficients from local patches are not perfect but provide a good starting point for further modelling of the spectrogram. We expect that better performance could be obtained by using more advanced cues from CASA, such as pre-clustering the spectrogram into pitch tracks and transient tracks, before learning[5].

4.4 Overall results

We now turn to results obtained by semi-supervised NMF combined with various annotation methods. On the SISEC database, manual time-frequency annotations were done with the GUI presented in Section 2.2. On the QUASI database, tracks were amenable to significant time annotations, so by comparing results on both databases we can compare the respective benefits of time-frequency annotations VS time annotations.

In both scenarios, we compare five methods :

auto : Automatic annotations and semi-supervised NMF. The best detector from Table 3 was chosen.

user : User annotations and semi-supervised NMF (time-frequency annotations for SISEC, manual annotations for QUASI). We tried $K \in \{5, 10, 20\}$ for the SISEC database and $\{10, 20, 50\}$ for the QUASI database, as well as $\lambda \in \{1, 10, 100\}$, and selected parameters yielding highest SDR for fair comparison with the baseline.

baseline : Run NMF and permute factors to obtain optimal SDR. We set $K = 8$ because it already takes a 10 times as long to evaluate SDRs for all permutation on a single track as it takes to run semi-supervised NMF.

self : set $s^{(g)} = \frac{1}{G}x$ as estimates for the sources, it serves to estimate the difficulty of the source separation problem for a given database.

oracle : results obtained with Wiener coefficients computed from the ground truth. In addition we display track by track annotation accuracy for user annotations, for comparison with Table 2. For each method, we ran NMF three times for 1000 iterations to avoid local minima, and kept the run with the lowest objective cost value.

Tables 5a and 5b display average evaluation metrics for each source (source 1 is always the accompaniment, and source 2 is always the voice), on two different databases :

	% annotated	% correct
track 1	0.23	0.91
track 2	0.10	0.89
track 3	0.29	0.91
track 4	0.17	0.81
track 5	0.22	0.95

Table 4: Evaluation of user annotations on the SISEC database.

[5] This is very similar to what is done in vision, where super-pixels help deal with consistency in prediction and alleviate the computational burden of predicting all pixel values.

	auto	**user (t-f)**	**baseline**	**self**	**oracle**
SDR1	0.97	6.21	6.16	3.09	14.79
SDR2	0.51	2.58	1.61	-3.18	11.53
SIR1	3.17	18.64	9.91	3.09	24.00
SIR2	4.57	11.35	5.09	-3.18	23.90
SAR1	6.74	6.91	9.26	279.17	15.41
SAR2	4.18	3.91	5.58	279.17	11.84
% ann.	8.69	19.81	0.00	0.00	100.00

(a) SISEC

	auto	**user (t)**	**baseline**	**self**	**oracle**
SDR1	6.76	7.59	6.29	6.21	16.88
SDR2	-4.33	-4.57	-1.71	-6.22	10.37
SIR1	6.97	15.05	13.81	6.21	25.62
SIR2	-3.75	4.09	1.88	-6.22	24.83
SAR1	21.91	9.00	7.71	268.45	17.66
SAR2	10.28	0.21	4.29	268.45	10.60
% ann.	6.91	100.00	0.00	0.00	100.00

(b) QUASI

Table 5: Results on the evaluated databases: (a) time-frequency annotations, (b) time annotations.

on the SISEC database, we experimented with time-frequency annotations since the tracks were too short for time annotations. Overall, results on the SISEC database are better than those on QUASI. Our interpretation is that since most of the time the accompaniment is active, the dictionaries tend to overfit the accompaniment and underfit the voice. Time-frequency annotations on SISEC yield SDRs that are a few points below that predicted by our benchmark from Table 2 : indeed human errors are not distributed randomly as was the case in our benchmark. Time-frequency annotations outperform the baseline by 1 point in SDR, which is significant because in semi-supervised NMF there is no manual grouping of the components. Time annotations loose to the baseline by -1 in SDR, but they are still significantly correlated with the true sources when compared with the baseline.

On the SISEC database, automatic annotations are also below the baseline, however they are also significantly correlated with the true sources, when compared with the "self" column. Signal to Interference Ratios are even comparable with those of the baseline on the SISEC database. Automatic annotations do not perform as well on the QUASI database since we trained detectors only on tracks from SISEC, so that more supervision would significantly improve those figures.

To conclude, we have shown that time-frequency annotations can improve significantly over NMF with ideally grouped components. On longer tracks, time only annotations yield reasonable results, but even when 100% of the track is annotated, the estimated sources contain strong interferences. Automatic annotations yield similar results, but leave considerable room for improvement, since with time-frequency annotations there will always be a point where enough annotations with limited errors will provide audible estimates of the sources.

5. CONCLUSION

We have proposed a novel formulation of semi-supervised NMF that successfully takes into account annotations to enhance the discriminative power of NMF. Semi-supervised NMF is defined so that when a certain amount of annotations is reached, source separation quality is near that of ideal binary masks. Manual annotations retrieved with our graphical user interface yield satisfactory results. We are investigating ways to define annotations independently of the particular time-frequency representation that is used.

Finally, semi-supervised NMF opens the way for interaction with methods from computational audio scene analysis. As such, the simple features and textbook pattern matching algorithms we have presented show promising results.

6. REFERENCES

[1] P. Smaragdis and G. Mysore. "Separation by "humming": User-guided sound extraction from monophonic mixtures.", *WASPAA*, 2012.

[2] B. Wang. "Musical Audio Stream Separation", *Msc Thesis*, 2009.

[3] D. Fitzgerald. "User Assisted Source Separation using Non-negative Matrix Factorisation", *Irish Signals and Systems Conference*, 2011.

[4] F. Bach and M.I. Jordan. "Blind one-microphone speech separation: a spectral learning approach." *NIPS*, 2004.

[5] J.-L. Durrieu and J.-P. Thiran. "Musical audio source separation based on user-selected f0 track." *(LVA/ICA)*, 2012.

[6] C. Févotte, N. Bertin, and J.-L. Durrieu. "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis." *Neural Computation*, 2009.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2nd edition edition, 2009.

[8] R. Hennequin, B. David, and R. Badeau. "Score informed audio source separation using a parametric model of non-negative spectrogram." *ICASSP*, 2011.

[9] M. Lagrange, L.G. Martins, J. Murdoch, and G. Tzanetakis. "Normalized cuts for predominant melodic source separation." *TASLP*, 2008.

NEON.JS: NEUME EDITOR ONLINE

Gregory Burlet, Alastair Porter, Andrew Hankinson, Ichiro Fujinaga
Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
McGill University, Montréal, Québec, Canada
{gregory.burlet,alastair.porter,andrew.hankinson}@mail.mcgill.ca, ich@music.mcgill.ca

ABSTRACT

This paper introduces Neon.js, a browser-based music notation editor written in JavaScript. The editor can be used to manipulate digitally encoded musical scores in square-note notation. This type of notation presents certain challenges to a music notation editor, since many neumes (groups of pitches) are ligatures—continuous graphical symbols that represent multiple notes. Neon.js will serve as a component within an online optical music recognition framework. The primary purpose of the editor is to provide a readily accessible interface to easily correct errors made in the process of optical music recognition. In this context, we envision an environment that promotes crowdsourcing to further the creation of editable and searchable online symbolic music collections and for generating and editing ground-truth data to train optical music recognition algorithms.

1. INTRODUCTION

Music notation editors allow users to manipulate the arrangement of symbols within a digitally encoded musical score. Given the complexity of some musical works, the arrangement of symbols may have intricate relationships. Consequently, a music notation editor must have knowledge of these symbols and their relationships in a musical context to ensure that transformations to the symbols in the editor yield a music score that is syntactically correct. For example, the editor should ensure that notes are placed correctly on the staff and that musical properties are not violated.

There are three main issues that a music notation editor must address: the notation encoding schema, the rendering of symbols, and the relationship between the notation encoding and the graphical representation. First, the digital representation of symbols in the music notation system must be systematically defined so that it represents the desired musical structures and hierarchies. Second, each symbol must be graphically rendered as glyphs on the screen. Finally, changes made by the user in the graphical interface must be translated accurately and completely to the underlying encoding—a non-trivial task since there may be cascading effects on other symbols in the encoded document.

This paper introduces Neon.js, a music notation editor for an early notation system known as *square-note notation*, which originated in the 13th century. In this system of notation, individual notes may be grouped into larger structures called *neumes*, which represent the pitches of a vocal phrase spanning a single syllable. Neumes have different names that are determined by the pitch contour of the individual notes making up the neume. Most neumes are *ligatures*—symbols that represent multiple connected notes. Square-note notation editors must be able to correctly render these ligatures, while still allowing access to individual notes in the neume. An overview of this notation system and example scores can be found in the *Liber Usualis*, a compilation of plainchant used by the Roman Catholic Church [3].

Neon.js is a web application. It displays square-note notation in the web browser and accepts user input to modify the underlying digital representation of the score. Neon.js can be used to create new musical scores, or to correct errors from automated transcriptions in an optical music recognition (OMR) workflow.

By making the editor easily accessible online, OMR error correction tasks can be distributed amongst many people, potentially accelerating the creation of ground-truth training data and errorless symbolic music collections. The process of outsourcing a simple, defined task to the general population is known as crowdsourcing. We predict that by providing tools to enable crowdsourcing of the correction of OMR transcription errors, early music researchers will be able to obtain transcriptions more quickly in order to perform computer-assisted analyses of these works.

We begin with a review of music notation editors and online crowdsourcing systems. Next, we discuss the benefits of a browser-based editor, the requirements for a musical document-encoding scheme, and available techniques for drawing in a web browser. Finally, the software architecture of Neon.js is introduced and the main functions of the editor are described, with a focus on functionality that is unique to square-note notation.

2. RELATED WORK

2.1 Music Notation Editors

An editor for square-note notation contains some functionality that is similar to editors of common music notation. The core functionality required of a notation editor is to allow the user to insert, delete, and change symbols in a piece of notated music. Well-known music notation editors such as Finale[1], Sibelius[2], MuseScore[3], and NoteFlight[4] provide these basic functions and more. Of these, NoteFlight is the only known example of a web-based editor for common music notation. All of these editors allow the user to save files containing the digital encoding of the edited musical document. These editors allow changes to individual symbol attributes, like note pitch and ornamentation, as well as changes to global musical properties, such as tempo and time signature. Since certain properties of common music notation do not exist in square-note notation, an editor for neume notation does not need to handle features such as tempo, time signature, chords, multiple voices on a staff, beams, slurs, or tuplet marks.

Editors for square-note notation do exist, although they are significantly less prevalent than editors for common music notation. The Medieval plugin[5] for Finale allows square-note notation to be edited in Finale. To input and render this notation in Finale, the plugin must work around some syntactic musical requirements that Finale imposes on scores. For example, the plugin alters the time signature of each bar in the score (each of which may have a different number of notes), to prevent Finale from automatically inserting bar lines.

Another neume notation editor has been developed as part of the NEumed Unicode Manuscript Encoding Standard (NEUMES) project [2]. The editor is developed as a Java applet and is therefore available for use in a web browser[6]. The editor uses NeumesXML, an encoding format that describes square-note notation as well as earlier neume notation systems without staff lines. NeumesXML is an XML-based format that uses Unicode characters to represent different neumes. Although the editor did not advance beyond the prototyping phase of development, the user interface allows scanned images of musical scores to be displayed side-by-side with the rendered notation for reference.

As part of the Tübingen project [7], the MEI-Neumes-Viewer[7] renders neume notation in the web browser. In the preliminary stages of the project, the *Humdrum Toolkit* was used to develop a data representation specific to their repertoire. In later stages of the project[8] they developed the neume-encoding scheme that is now part of the Music Encoding Initiative (MEI). The MEI-Neumes-Viewer is an engraving system, not an editor, and so does not provide the ability to interactively edit a score.

Many music notation encoding schemes have been proposed over the years, most of which are tailored to a specific notation system. However, few formats seek to provide a universal data representation that encompasses and describes all musical notation systems. A music notation editor requires a digital representation of the symbolic notation because it needs to be stored and processed by a computer. Digital encoding systems should be explicit and declarative to prevent loss of information [11]. An encoding system should not require software applications to derive relationships between elements in the musical score. Instead, all relationships should be explicitly described. For neume notation, the digital encoding should preserve neume types and pitch information [7].

Aruspix, an OMR system for recognizing early printed music [10], contains an editor that is used to correct recognition errors in OMR output. Aruspix renders the recognized score over top of the original image, facilitating error detection and correction by the user.

2.2 Crowdsourcing Systems

Many projects have benefited from using online crowdsourcing techniques to harness the computational power of many humans (e.g., Wikipedia). The reCAPTCHA project [1] has successfully transcribed over 440 million words that were unrecognizable by optical character recognition algorithms. This was accomplished by presenting transformed words as "Completely Automated Public Turing test to tell Computers and Humans Apart" (CAPTCHA) challenges on the web. These tests are designed to prevent malicious software from performing actions that should only be performed by humans, who must transcribe the machine-unrecognizable words to prove they are human. In the process, optical character recognition errors are corrected [1]. When this is deployed over millions of websites, it becomes a highly effective method of performing large-scale text correction. A further advantage of online crowdsourcing for document digitization is that it is inexpensive and requires minimal training for the contributors, compared to the cost and training required to set up a scanning and recognition workstation [8].

3. ONLINE EDITING

3.1 Web Versus Desktop Applications

The AJAX programming technique [9] for websites enables the creation of interactive web applications that behave like desktop applications. This has lead to discussions, involving both users and software developers, about the advantages and disadvantages of creating web applications over desktop applications.

[1] http://www.finalemusic.com
[2] http://www.sibelius.com
[3] http://musescore.org
[4] http://www.noteflight.com
[5] http://www.klemm-music.de/notation/medieval
[6] http://www.scribeserver.com/medieval/staves_applet.html
[7] http://www.dimused.uni-tuebingen.de/hildegard
[8] http://www.dimused.uni-tuebingen.de/tuebingen_phase1_e.php

There are many features of web applications that are appealing to application users and developers. Unlike desktop applications, web applications can be continually updated by developers with new features and do not need to be updated by the end-user when a new version is released. Another feature is that any user with an Internet connection and a web browser can access and use the application, whereas desktop applications require time to install and typically have different installation procedures depending on the operating system of the client machine. While not necessarily browser independent, web applications are platform independent, creating a larger user base.

There are, however, disadvantages of hosting a music notation editor online. The most notable disadvantage is that the user must be connected to the Internet to use the application. Compared to desktop applications, web applications that have a client-server architecture can be architecturally complex since interactions and synchronicity between the client and server must be maintained. Advocates of desktop application development also claim that web applications exhibit slower performance and discontinuous user interactions in relation to compiled programs. These deficits are becoming negligible now that contemporary web browsers have become faster at executing JavaScript, and the use of AJAX has enabled seamless and transparent communication between the client and server.

Web applications have a lower barrier to entry than desktop applications. Fewer obstacles are presented to users interested in using the software, making them more suitable for use in crowdsourcing applications targeted to a large number of people.

3.2 Crowdsourcing online

Crowdsourcing may be used in an OMR workflow for the purposes of quickly and inexpensively correcting errors in digitized musical documents. Pattern recognition algorithms are not perfect—there will inevitably be recognition errors that must be fixed by a human. A single person can allocate a large amount of time to perform these corrections, or this unwieldy workload can be distributed amongst many people. The resulting digital encoding can then be archived, indexed for searching, or used as ground-truth data to further train OMR algorithms.

In creating an online crowdsourcing system for the correction of OMR errors, there are four challenges that developers of these systems must consider [4]. First, users must be recruited and their interest maintained. A potential recruitment problem exists for the correction of square-note notation, since early music is less popular than contemporary music. Next, large tasks must be decomposed into manageable sub-problems, which are easily solvable by a single person. This method of problem solving is called divide-and-conquer. For OMR, error correction of an entire corpus can be broken down into smaller tasks where segments of music, such as a single page, line, or bar, are corrected. Each subtask would require users to correct the position and pitch of erroneous notes. A typical correction task is displayed in Figure 1. By providing an image of the original musical document for reference, a task that would normally require domain-specific musical knowledge can be posed as a comparison problem. The task of correcting pitch and position errors involves dragging the incorrectly recognized notes to match the position of those notes on the source image. This increases the number of possible contributors that can be recruited to perform correction tasks. One potential limitation of this correction scheme occurs when the neume type is incorrectly recognized. In this case, the incorrect glyph must be deleted and the correct glyph inserted to match the original score—a task that may require musical knowledge and may be surrendered to another user. The encouragement and retention scheme known as instant gratification could be used to immediately show contributors how their contributions are making a difference [4]. After correcting errors within a segment of music, displaying or auralizing the full score would not only emphasize the contributions of the user, but also encourage future corrections of OMR output.

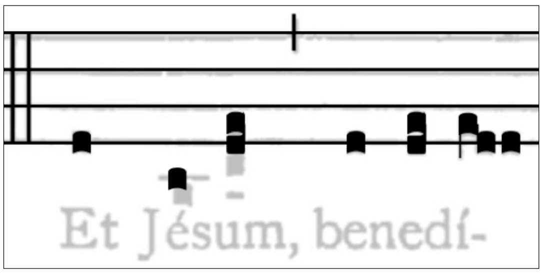


Figure 1. An example of a pitch correction task. Some recognized notes (dark) need to be moved to the same location as on the original score (light).

The last two challenges a crowdsourcing system must consider is the evaluation of user contributions, and the combination of these contributions to solve the target problem. To manage OMR error corrections by multiple contributors, an ideal system might present the same segment of music to more than one person for correction. An automated voting system [1] can be used to choose the most commonly made correction for a segment of music and then combine segments into the final digitized score.

3.3 Tools for Musical Engraving and Interaction

Adobe Flash, Scalable Vector Graphics (SVG), and the HTML canvas element are technologies available for dynamic drawing in the web browser. Flash is a popular web technology that supports interactive manipulation of the drawing surface with scripts to produce interactive

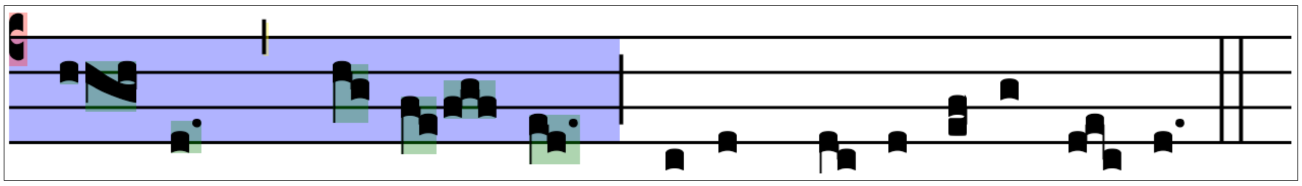


Figure 2. Neon.js rendering one system of the *Liber Usualis*, with and without bounding boxes.

and media-driven web applications. However, web applications that use Flash require an additional program to be installed as a browser plugin in order to be used. Some operating systems and devices do not support Flash and therefore cannot run these applications.

SVG is an XML format for describing images as vectors—mathematical descriptions of lines and curves. This representation enables SVG images to be rendered at any size, making the format a good choice for images that require a high level of detail. Most modern web browsers can display SVG images and manipulate them using JavaScript.

The HTML canvas element is a low-level raster drawing surface supported by modern web browsers, which can be controlled with JavaScript. JavaScript libraries such as jQuery [10] can be used to simplify common tasks such as event handling, animations, and AJAX interactions. The canvas element implements a "fire-and-forget" model, where a drawing surface is presented as a two-dimensional array of pixels that can be independently coloured. Although images drawn on the canvas can be easily manipulated, interaction with the drawings requires some additional overhead. Since the canvas "forgets" what was drawn, the state of objects drawn to the canvas needs to be maintained. When the state of an object changes (e.g., dimensions and colour), the canvas is repainted to reflect these changes.

4. A NEUME NOTATION EDITOR

This section introduces our web-based neume notation editor, Neon.js. We start with a description of music notation encoding and how it can be rendered in a web browser. We then describe how the music notation editor links the notation encoding and the graphical representation.

4.1 Notation Encoding

Neon.js reads and writes files encoded in the Music Encoding Initiative (MEI) format. MEI is an XML-based file format for the representation of many notation formats. The MEI schema is split into a "core", which defines features common to many notation formats, and a set of additional modules that each define a specific notation system [6]. Neon.js uses the Solesmes module, an extension to the MEI core that allows representation of square-note notation along with other specific practices particular to the notation system used by monks in Solesmes, France in the 19th century. These practices include *divisions* (breath marks), *episemata* (note stresses), and unique neume names.

[10] http://jquery.com

4.2 Notation Rendering

We decided to use the canvas element for rendering scores in Neon.js. We store images of neumes and ligatures as SVG so that the score can be rendered in detail at any zoom level. We use Fabric.js[11], a library that provides high-level drawing functions such as lines and boxes. Fabric.js is also used to render our SVG images directly onto the canvas drawing surface in the browser.

MEI files that have been transcribed by OMR contain the physical locations on the page of each recognized element, stored as a bounding box surrounding that element. We use these physical locations to calculate where to draw the musical symbols, including systems, clefs, neumes, and divisions. An example of Neon.js rendering one system of music from the *Liber Usualis* is shown in Figure 2.

Neon.js draws elements of the score on top of an image of the musical document that the score was transcribed from. The user can adjust the transparency of the background image to show just the rendered notation, or both the background and notation.

4.3 Software Architecture

Neon.js is built using a client-server architecture. The Neon.js client renders the musical score in the browser and transforms user input into edit commands that are sent to the server. We use AJAX to send edit commands from the client to the server without needing to reload the web page. The server receives these commands from the client and applies them to a stored MEI file, saving the changes. The server is written in Tornado[12], an open-source Python web server framework. To read, manipulate, and write MEI files on the server, Neon.js makes use of the Python bindings of libmei[13], an open-source C++ library [6].

The Neon.js client uses object-oriented programming techniques and the model-view-controller design pattern [5] to separate display from musical knowledge. The role

[11] http://fabricjs.com
[12] http://www.tornadoweb.org
[13] http://ddmal.music.mcgill.ca/libmei

of the model-view-controller design pattern is to isolate application logic in the model from display and user interface logic in the view through an intermediary controller that coordinates the two. The model keeps the state of all of the musical elements on a score. When a user modifies an element, the score is redrawn to reflect the changes to the object's state. This means that changing the data representation from MEI to NeumesXML, for example, would not affect the drawing code. Similarly, changing the drawing medium from canvas to SVG, for example, would not affect the data representation and editing functionality.

5. EDITING FUNCTIONS

To develop a more thorough understanding of the major functions of Neon.js, the structure of an instantiated neume object will be described. In Neon.js, a neume is represented as a sequence of puncta. A punctum is the simplest type of neume, consisting of only one note that is represented as a single square. Only the note name and octave of the first note is stored in the model. Subsequent notes are encoded as having a pitch relative to the first note. The following client-side functions operate on this information and then call a corresponding server function to update the underlying MEI file. In this section we focus on implementation details of the main editing functions that are specific to neume notation.

5.1 Inserting and Deleting Neumes

The only neume type that can be added to the score through the user interface is the punctum. Neumes that are composed of more than one note are entered by inserting a punctum for each pitch in the neume, then combining them with the neumify function (Section 5.2). This feature lets a user focus on the melodic content of the score without needing to identify the name of each neume to insert.

Neumes may be deleted. When a neume contains multiple notes, all of the notes in the neume are deleted. To delete individual notes within a neume, it must first be ungrouped (Section 5.3).

5.2 Neumify

In the same way that ligatures pose problems for OMR systems by obfuscating pitches of individual notes [12], ligatures within neumes pose problems for square-note notation editors. In many cases, the graphical representation of a neume is not a simple concatenation of the selected glyphs. Figure 3 shows three notes being selected and combined by the neumify function into a *porrectus* neume, identified by the downward then upward melodic contour of the notes. Neon.js needs to be able to recognize a neume by its contour in order to draw it with the correct ligatures. MEI also requires the name of a neume to be encoded along with the notes that make up the neume. The neumify function infers the name of a neume from the pitch differences in a sequence of notes.

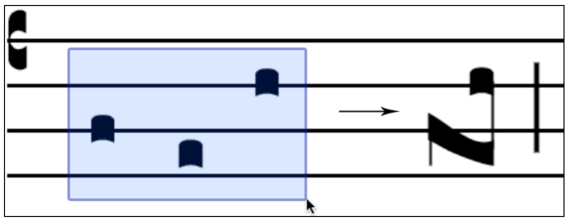


Figure 3. A use case where the user selects a set of puncta and applies the neumify function to create a *porrectus* neume.

When the neumify function processes a sequence of notes, the melodic contour of the notes is calculated. To describe the relationship of a note to its previous note, -1, 0, or 1 is used to represent a lower pitch, the same pitch, or a higher pitch, respectively. We create a prefix tree containing all of the neume types that Neon.js can render. The edges of the tree represent the direction of movement between two notes. For example, the *porrectus* neume from Figure 3 has a contour of -1, 1. The traversal of a partial prefix tree is shown in Figure 4 with bolded lines. Using a prefix tree for lookups means that the speed of identifying a neume type is dependent on the number of notes in the neume and not the size of the tree, which means that more neume types can be added without affecting the speed of lookups. When the melodic movement of a sequence of puncta cannot be matched to a known neume type, the neume type is labeled "compound neume". Instead of naïvely concatenating the selected glyphs to form the drawing of a compound neume, the longest matching prefix in the tree can be used to determine the best way of rendering notes within the neume. Neon.js is currently capable of differentiating between 44 distinct neume types.

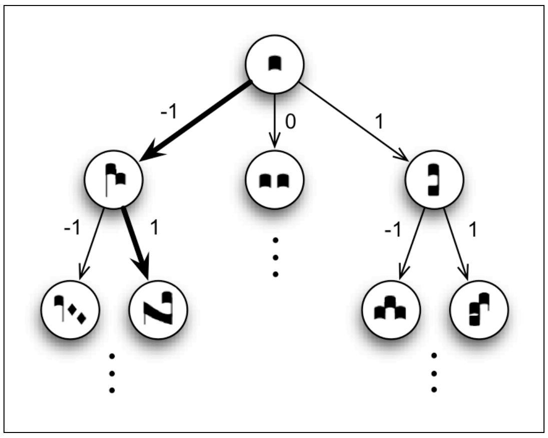


Figure 4. An example of deriving the *porrectus* neume type by searching a prefix tree. The bolded arrows reveal the traversal of the tree.

5.3 Ungroup

The ungroup function in Neon.js performs the inverse of the neumify function. When the ungroup function is applied to a neume, the neume is replaced by puncta having the same pitches as the underlying notes in the neume. This functionality lets a user adjust properties of a single note in a neume, such as pitch and ornamentation, and then regroup the neume with the neumify function.

5.4 Modify Pitch

Neon.js allows the user to modify pitches of neumes through a click and drag interface. The notes automatically snap to spaces or lines within the staff. When the user moves a neume comprised of multiple notes, all notes within the neume are shifted by the same amount.

6. CONCLUSION

Neon.js[14] is an online and open-source neume notation editor developed as a web application. The editor uses the HTML canvas element to render musical symbols in the web browser. Neon.js supports MEI as a digital data representation for square-note notation and uses open-source software libraries to facilitate manipulation of this data format. A minimal but powerful graphical user interface allows the user to digitally replicate or edit early music documents containing square-note notation. As the user modifies the arrangement of musical symbols, the underlying MEI file is modified to reflect these changes.

As a web application, the software is easily accessible, requires no installation, and enables large-scale crowdsourcing to distribute the task of correcting note pitch and position errors made in the process of optical music recognition. We hope that the resulting symbolic music collections will then be indexed and available for search at a single location, creating a substantial source of information for early music researchers. We also hope that this centralized collection of corrected symbolic musical documents is made available for use as ground-truth training data for the digitization of other manuscripts in square-note notation. Although developed as a component of an online optical music recognition workflow, Neon.js can also be used to manually transcribe or write new works.

7. ACKNOWLEDGEMENTS

The authors would like to thank Catherine Motuz for her answers to our early music notation questions. This project has been funded with the generous financial support of the Social Sciences and Humanities Research Council of Canada.

[14] http://ddmal.music.mcgill.ca/neon

8. REFERENCES

[1] Ahn, L. V., B. Maurer, C. McMillen, D. Abraham, and M. Blum. 2008. reCAPTCHA: Human-based character recognition via web security measures. *Science* 321 (5895): 1465–8.

[2] Barton, L. W. G. 2002. The NEUMES project: Digital transcription of Medieval chant manuscripts. In *Proceedings of the International Conference on WEB Delivering of Music*, 211–8.

[3] Catholic Church. 1963. *The Liber Usualis, with introduction and rubrics in English*. Tournai, Belgium: Desclée.

[4] Doan, A., R. Ramakrishnan, and A. Y. Halevy. 2011. Crowdsourcing systems on the world-wide web. *Communications of the ACM* 54 (4): 86–96.

[5] Gamma, E., R. Helm, R. Johnson, and J. Vlissides. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. Boston, MA: Addison-Wesley.

[6] Hankinson, A., P. Roland, and I. Fujinaga. 2011. The Music Encoding Initiative as a document-encoding framework. In *Proceedings of the International Conference on Music Information Retrieval*, Miami, FL, 293–8.

[7] Morent, S. 2001. Representing a Medieval repertory and its sources: The music of Hildegard von Bingen. *Computing in Musicology* 12: 19–31.

[8] Newby, G. B., and C. Franks. 2003. Distributed proofreading. In *Proceedings of the Joint Conference on Digital Libraries*. 361–3.

[9] Paulson, L. D. 2005. Building rich web applications with AJAX. *IEEE Computer* 38 (10): 14–7.

[10] Pugin, L. 2009. Editing Renaissance music: The Aruspix project. In *Digitale Edition zwischen Experiment und Standardisierung Musik - Text - Codierung*, 147–56.

[11] Roland, P. 2002. The Music Encoding Initiative (MEI). In *Proceedings of the International Conference on Musical Applications Using XML*, 55–9.

[12] Vigliensoni, G., J. A. Burgoyne, A. Hankinson, and I. Fujinaga. 2011. Automatic pitch detection in printed square notation. In *Proceedings of the International Society for Music Information Retrieval Conference*, Miami, FL, 423–8.

MODELING CHORD AND KEY STRUCTURE WITH MARKOV LOGIC

Hélène Papadopoulos and George Tzanetakis
Computer Science Department, University of Victoria
Victoria, B.C., V8P 5C2, Canada
helene.papadopoulos@lss.supelec.fr
gtzan@cs.uvic.ca

ABSTRACT

We propose the use of Markov Logic Networks (MLNs) as a highly flexible and expressive formalism for the harmonic analysis of audio signals. Using MLNs information about the physical and semantic content of the signal can be intuitively and compactly encoded and expert knowledge can be easily expressed and combined using a single unified formal model that combines probabilities and logic. In particular, we propose a new approach for joint estimation of chord and global key The proposed model is evaluated on a set of popular music songs. The results show that it can achieve similar performance to a state of the art Hidden Markov Model for chord estimation while at the same time estimating global key. In addition when prior information about global key is used it shows a small but statistically significant improvement in chord estimation performance. Our results demonstrate the potential of MLNs for music analysis as they can express both structured relational knowledge as well as uncertainty.

1. INTRODUCTION

Content-based music retrieval is an active and important field of research within the Music Information Retrieval (MIR) community, that deals with the extraction and processing of information from musical audio. Many applications, such as music classification or structural audio segmentation, are based on the use of musical descriptors, such as the key, the chord progression, the melody, or the instrumentation. Often regarded as an innate human ability, the automatic estimation of music content information proves to be a highly complex task, for at least two reasons. The first reason is the great variability of musical audio caused by the many modes of sound production and the wide range of possible combinations between the various acoustic events which make music signals extremely rich and complex from a physical point of view. The second reason is that the information of interest is generally very complex from a semantic point of view and many musical descriptors, that are strongly correlated, are necessary to characterize it. For instance, the chord progression is related to the metrical structure of a piece of music: chords change more often on strong beats than on other beat positions in the measure [9]. The chord progression is also related to the musical key: some chords are heard as more stable within an established tonal context [13]. Recent work has shown that the estimation of musical attributes would benefit from a unified musical analysis [4, 14, 15, 21]. However, most of existing MIR systems that estimate musical content from audio signals have relatively simple probabilistic structure and are constrained by limited hypotheses that do not model the underlying complexity of music. The idea of reinforcing the performance of object recognition by considering contextual information has been explored in other fields than MIR, such as computer vision [17].

As many real-world systems and signals, music signals exhibit both uncertainty and complex relational structure. Until recent years, these two aspects have been generally treated separately, probability being the standard way to represent uncertainty in knowledge, while logical representation being used to represent complex relational information. However, alternative approaches towards a unification have been proposed within the emerging field of Statistical Relational Learning (SRL) [8]. Models in which statistical and relational knowledge are unified within a single representation formalism have emerged [6, 10, 18]. Among them, Markov Logic Networks (MLNs) [27], that combine first-order logic and probabilistic graphical models (Markov networks) have received considerable attention in recent years. Their popularity is due to their expressiveness and simplicity for compactly representing a wide variety of knowledge and reasoning about data with complex dependencies. Moreover, multiple learning and inference algorithms for MLNs have been proposed, for which open-source implementations are available, for example the *Alchemy*[1] and *ProbCog*[2] software packages. MLNs have thus been used for many tasks in artificial intelligence (AI), such as meaning extraction [2], collective classification [5], or entity resolution [32].

As far as we know, MLNs have not been used yet for music content processing. Chord recognition is one of the most popular MIR tasks as reflected by the number of related papers and the increasing number of contributions to the annual MIREX[3] evaluation. We propose MLNs as a highly flexible and expressive modeling language for es-

[1] http://alchemy.cs.washington.edu
[2] http://ias.cs.tum.edu/research/probcog
[3] http://www.music-ir.org/mirex/

timating the chord progression of a piece of music. The main contribution is to show how various types of information about the physics and the semantics of the signal can be intuitively and compactly encoded in a unified formalism. In addition, MLNs allow incorporating expert knowledge in the model in a flexible fashion. In particular, we show how prior information about the main key of an analyzed excerpt can be used to enhance the chord progression. We also propose a new approach for the estimation of harmonic structure and global key, in which the two attributes are estimated jointly and benefit from each other.

2. BACKGROUND

Previous approaches for chord estimation can be classified into two categories: approaches based on pattern-matching and probabilistic approaches. One of the advantages of probabilistic approaches is that they can model uncertainty and variability. Indeed, the realization of a chord produced in different conditions (instrumentation, dynamics, room acoustics, etc.) can result in significantly different signal observations. Moreover, probabilistic models allow incorporating context information to improve chord estimation. For example, chord transitions based on musical rules can be embedded in the model to improve estimation. A large number of existing algorithms are based on the use of Hidden Markov Models (HMM), see *e.g.* [29, 31]. One of the reasons is that chord transition rules may be incorporated into the state transition matrix of the HMM. In the framework of HMMs, additional context information, such as the key [4, 14], the meter [23] or the structure [16], can also be incorporated to improve the estimation.

Other statistical machine learning approaches for chord estimation include conditional random fields [3], which compared to HMMs do not require the observation vectors to be conditionally independent. The use of N-grams [30, 33] allows information about longer range chord dependencies to be considered. In contrast, HMMs make the Markovian assumption that each chord symbol only depends on the preceding one. In some of these approaches, context information is incorporated, such as in the graphical probabilistic model [20] where contextual information related to the meter is used, or in [15], where a 6-layered dynamic Bayesian network jointly modeling key, metric position, chord and bass pitch class is proposed.

Existing approaches for chord recognition, in particular HMMs, have been quite successful in modeling chord sequences. However, their limited probabilistic structure makes the incorporation of additional contextual information a complex task. More specifically, concerning chords and key interaction, state-of-the-art approaches may not fully exploit interrelationship between musical attributes, as in [24] and [19] where key estimation is based on the chord progression, but the chord estimation part does not benefit from key information. Other approaches [28] do not allow easily introducing expert knowledge (such as musical information about the key progression) that could help music content analysis. In this paper, we intend to show how such relational cues can be compactly modeled within the framework of Markov logic.

3. MARKOV LOGIC NETWORKS

A Markov Logic Network (MLN) is a set of weighted first-order logic formulas [27], that can be seen as a template for the construction of probabilistic graphical models. We present a short overview of the underlying concepts with specific examples from the modeling of chord structure. A MLN is a combination of Markov networks and first-order logic. A *Markov network* is a model for the joint distribution of a set of variables $X = (X_1, X_2, ..., X_n) \in \mathcal{X}$ [25], that is often represented as a log-linear model:

$$P(X = x) = \frac{1}{Z} exp(\sum_j w_j f_j(x)) \tag{1}$$

where Z is a normalization factor, and $f_j(x)$ are features of the state x (x is an assignment to the random variables X). Here, we will focus on binary features, $f_j(x) \in 0, 1$.

A first-order domain is defined by a set of *constants* (that is assumed finite) representing objects in the domain (e.g., CMchord, GMchord) and a set of *predicates* representing properties of those objects (e.g., IsMajor(x), IsHappyMood(x)) and relations between them (e.g., AreNeighbors(x, y)). A predicate can be *grounded* by replacing its variables with constants (e.g., IsMajor(CMchord), IsHappyMood(CMchord), AreNeighbors(CMchord, GMchord)). A *world* is an assignment of a truth value to each possible ground predicate (or atom). A *first-order knowledge base* (KB) is a set of formulas in first-order logic, constructed from predicates using logical connectives and quantifiers. A first-order KB can be seen as a set of hard constraints on the set of possible worlds: if a world violates even one formula, it has zero probability. Table 1 shows a simple KB. In a real world scheme, logic formulas are *generally* true, but not *always* true. The basic idea in Markov logic is to soften these constraints to handle uncertainty: when a world violates one formula in the KB, it is less probable than one that does not violate any formulas, but not impossible. The weight associated with each formula reflects how strong a constraint is, i.e. how unlikely a world is in which that formula is violated.

Table 1. Example of a first-order KB and corresponding weights in the MLN.

Knowledge	*Logic formula*	*Weight*
A major chord implies an happy mood.	∀ x IsMajor(x) ⇒ IsHappyMood(x)	$w_1 = 0.5$
If two chords are neighbors, either the two are major chords or neither are.	∀ x ∀ y AreNeighbors(x, y) ⇒ (IsMajor(x) ⇔ IsMajor(y))	$w_2 = 1.1$

Formally, a *Markov logic network* L is defined [27] as a set of pairs (F_i, w_i), where F_i is a formula in first-order logic and w_i is a real number associated with the formula. Together with a finite set of constants C (to which the predicates appearing in the formulas can be applied), it defines a ground Markov network $M_{L,C}$, as follows:

1. $M_{L,C}$ contains one binary node for each possible grounding of each predicate appearing in L. The node value is 1 if the ground predicate is true, and 0 otherwise.

2. $M_{L,C}$ contains one feature for each possible grounding of each formula F_i in L. The feature value is 1 if the ground formula is true, and 0 otherwise. The feature weight is the w_i associated with F_i in L.

A ground Markov logic network specifies a probability distribution over the set of possible worlds $\mathcal{X}$. The joint distribution of a possible world x is:

$$\begin{aligned} P(X=x) &= \frac{1}{Z} exp(\sum_i w_i n_i(x)) \\ &= \frac{exp(\sum_i w_i n_i(x))}{\sum_{x' \in \mathcal{X}} exp(\sum_i w_i n_i(x'))} \end{aligned}$$

where the sum is over indices of MLN formulas and $n_i(x)$ is the number of true groundings of formula F_i in x. (i.e. $n_i(x)$ is the number of times the i^{th} formula is satisfied by possible world x).

Figure 1 shows the graph of the ground Markov network defined by the two formulas in Table 1 and the constants CMchord and GMchord. Each possible grounding of each predicate becomes a node in the corresponding Markov Network. There is an arc in the graph between each pair of atoms that appear together in some grounding of one of the formulas. The grounding process is illustrated in Figure 2.

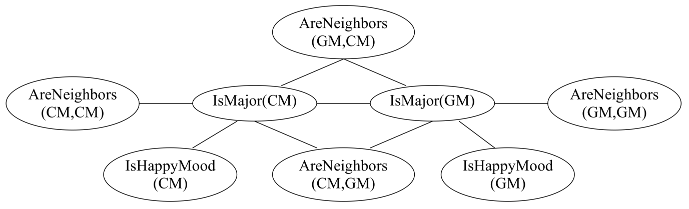


Figure 1. Ground Markov network obtained by applying the formulas in Table 1 to the constants CMchord (CM) and GMchord (GM).

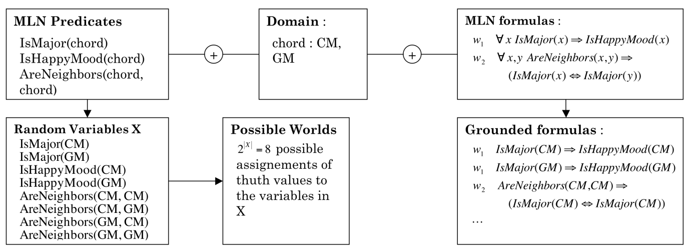

MLN Predicates
IsMajor(chord)
IsHappyMood(chord)
AreNeighbors(chord, chord)
Domain :
chord : CM, GM
MLN formulas :
w_1 $\forall x$ $IsMajor(x) \Rightarrow IsHappyMood(x)$
w_2 $\forall x,y$ $AreNeighbors(x,y) \Rightarrow (IsMajor(x) \Leftrightarrow IsMajor(y))$
Random Variables X
IsMajor(CM)
IsMajor(GM)
IsHappyMood(CM)
IsHappyMood(GM)
AreNeighbors(CM, CM)
AreNeighbors(CM, GM)
AreNeighbors(GM, CM)
AreNeighbors(GM, GM)
Possible Worlds
$2^{|X|} = 8$ possible assignements of thuth values to the variables in X
Grounded formulas :
w_1 $IsMajor(CM) \Rightarrow IsHappyMood(CM)$
w_1 $IsMajor(GM) \Rightarrow IsHappyMood(GM)$
w_2 $AreNeighbors(CM,CM) \Rightarrow (IsMajor(CM) \Leftrightarrow IsMajor(CM))$
...

Figure 2. Illustration of the grounding process of the Ground Markov network in Figure 1. Adapted from [12].

4. PROPOSED MODEL

In this section, we show how we can move from a standard HMM to a MLN, resulting in an elegant and concise representation with flexible modeling of context information.

4.1 Baseline HMM

We utilize a baseline model for chord estimation proposed in [22, 23] and briefly described here. The front-end of our model is based on the extraction of chroma feature vectors [7] that describe the signal. The chroma vectors are 12-dimensional vectors that represent the intensity of the twelve semitones of the Western tonal music scale, regardless of octave. We perform a *beat synchronous* analysis and compute one chroma vector per beat [4]. A chord lexicon composed of $I = 24$ major (M) and minor (m) triads is considered. The chord progression is then modeled as an ergodic 24-state HMM, each hidden state s_n (n denotes the time index) corresponding to a chord of the lexicon (CM, . . . , BM, Cm, . . . , Bm), and the observations being the chroma vectors o_n.

The HMM is specified using three probability distributions: the distribution $P(s_0)$ over initial states, the transition distribution $P(s_n|s_{n-1})$ and the observation distribution $P(o_n|s_n)$. The state-conditional observation probabilities $P(o_n|s_n)$ are obtained by computing the correlation between the observation vectors (the chroma vectors) and a set of chord templates which are the theoretical chroma vectors corresponding to the $I = 24$ major and minor triads. A state-transition matrix based on musical knowledge [19] is used to model the transition probabilities $P(s_n|s_{n-1})$, reflecting chord transition rules. The chord progression over time is estimated in a maximum likelihood sense by decoding the underlying sequence of hidden chords $S = (s_1, s_2, \ldots, s_N)$ from the sequence of observed chroma vectors $O = (o_1, o_2 \ldots, o_N)$ using the Viterbi decoding algorithm :

$$\hat{S} = \underset{S}{\operatorname{argmax}}(p(S,O)). \quad (2)$$

4.2 MLN for Chord Recognition

We now present a MLN for the problem of chord estimation, that is derived from the baseline HMM. MLNs are more general than HMMs, and we describe how the HMM structure can be expressed in a straightforward way using a MLN. Our MLN for chord recognition consists of a set of first-order formulas and their associated weights. It is described in Table 2. Given this set of rules with attached weights and a set of evidence literals, described in Table 3, Maximum A Posteriori (MAP) inference is used to infer the most likely state of the world.

Let $c_i, i \in [1, 24]$ denote the 24 chords of the dictionary, and $o_n, n \in [0, N-1]$ denote the succession of observed chroma vectors, with N being the total number of beat-synchronous frames of the analyzed song. The chord estimation problem can be formulated in Markov logic by defining formulas in the MLN using an unobserved predicate $Chord(c_i, t)$, meaning that chord c_i is played at frame t, and two observed ones, $Observation(o_n, t)$, meaning that we observe chroma o_n at frame t, and $Succ(t_1, t_2)$, meaning that t_1 and t_2 are successive frames. The constraints given by the prior, observation and transition probabilities of the baseline HMM form the abstract model. They are simply described by three MLN generic formulas. For each conditional distribution, only mutually exclusive and exhaustive sets of formulas are used, *i.e.* exactly one of them is true. For instance, there is one and only one possible chord per frame. This is indicated in Table 2 using the symbol !. The evidence consists of a set of ground atoms that give the chroma observations corresponding to each frame, and the temporal succession of frames over time. The query is the chord progression.

[4] This is done by integrating a beat-tracker as a front-end of the system [26].

Table 2. Chord recognition MLN used for inference.

Predicate declarations	
$Observation(chroma!, time)$	
$Chord(chord!, time)$	
$Succ(time, time)$	
Weight	*Formula*
Prior observation probabilities:	
$log(P(CM(t=0)))$	$Chord(CM, 0)$
$\cdots$	$\cdots$
$log(P(Bm(t=0)))$	$Chord(Bm, 0)$
Probability that the observation (chroma) has been emitted by a chord:	
$log(P(o_0\|CM))$	$Observation(o_0, t) \wedge Chord(CM, t)$
$log(P(o_0\|C\#M))$	$Observation(o_0, t) \wedge Chord(C\#M, t)$
$\cdots$	$\cdots$
$log(P(o_{N-1}\|Bm))$	$Observation(o_{N-1}, t) \wedge Chord(Bm, t)$
Probability to transit from one chord to another:	
$log(P(CM\|CM))$	$Chord(CM, t_1) \wedge Succ(t_2, t_1) \wedge Chord(CM, t_2)$
$log(P(C\#M\|CM))$	$Chord(CM, t_1) \wedge Succ(t_2, t_1) \wedge Chord(C\#M, t_2)$
$\cdots$	$\cdots$
$log(P(Bm\|Bm))$	$Chord(Bm, t_1) \wedge Succ(t_2, t_1) \wedge Chord(Bm, t_2)$

Table 3. Evidence for MLN chord estimation.

// We observe a chroma at each time frame:
$Observation(o_0, 0)$
$\cdots$
$Observation(o_{N-1}, N-1)$
// We know the temporal order of the frames:
$Succ(1, 0)$
$\cdots$
$Succ(N-1, N-2)$

In many existing MLNs weights attached to formulas are obtained from training. However, we follow the baseline approach and use weights based on musical knowledge. They are directly obtained using the conditional prior, observation and transition probabilities of the baseline HMM.

The conditional observation probabilities are described using a set of conjunctions of the form:

$$\forall t \in [0, N-1] \quad log(P(o_n|s_n = c_i)) \qquad (3)$$
$$Observation(o_n, t) \wedge Chord(c_i, t)$$

for each combination of observation o_n and chord c_i. Conjunctions, by definition, have but one true grounding each. According to Eq.(2), the weight associated with each conjunction is set to $w = log(P(o_n|s_n = c_i))$, with $P(o_n|s_n)$ denoting the corresponding observation probability.

The transition probabilities are described using:

$$\forall t_1, t_2 \in [0, N-1] \quad log(P(s_n = c_i|s_{n-1} = c_j)) \qquad (4)$$
$$Chord(c_i, t_1) \wedge Succ(t_2, t_1) \wedge Chord(c_j, t_2)$$

for all pairs of chords $(c_i, c_j), i, j \in [1, 24]$, and with $p = P(s_n|s_{n-1})$ denoting the corresponding transition probability.

The prior observation probabilities are described using:

$$log(P(s_0 = c_i)) \quad Chord(c_i, 0) \qquad (5)$$

for each chord $c_i, i \in [1, 24]$ and with $P(s_0)$ denoting the prior distribution of states.

4.3 Including Prior Information on Key

In this section, we show how prior information about the key of the excerpt can be incorporated in the model. We assume that we know the key $k_i, i \in [1, 24]$ of the excerpt. Key is added as a functional predicate in Table 2 ($Key(key!, time)$) and given as evidence in the MLN by adding evidence predicates in Table 3 of the form:

$$Key(k_i, 0), Key(k_i, 1), \cdots, Key(k_i, N-1) \qquad (6)$$

Relying on the hypothesis that some chords are heard as more stable within an established tonal context [13], additional rules about key and chord relationship are incorporated in the model. Let $k_i, i \in [1, 24]$ denote the 24 major and minor keys and $c_j, j \in [1, 24]$ denote the 24 chords. For each pair of key and chords (k_i, c_j), we add the rule:

$$log(p_{ij}) \quad Key(k_i, t) \wedge Chord(c_j, t) \qquad (7)$$

where the values $p_{ij}, i, j \in [1, 24]$ define the prior distribution of chords $(c_1, \ldots, c_{24})$ given a key k_i. They are obtained from a set of key templates that represent the importance of each triad within a given key. The key templates are 24-dimensional vectors, each bin corresponding to one of the 24 major and minor triads. Two key templates, originally presented in [24], are considered. The first one, referred to as "weighted main chords relative" (WMCR) template, is derived from music knowledge, and attributes non-zero values to the bins corresponding to the most important triads in a given key (those built on the tonic, the subdominant and the dominant, plus the chord relative to the one built on the tonic) [13]. The second one, referred to as "cognitive-based" (CB) template, is built relying on a cognitive experiment conducted by Krumhansl [13], giving values corresponding to the rating of chords in harmonic-hierarchy experiments. Templates corresponding to C major (top) and C minor (bottom) keys are shown in Figure 3.

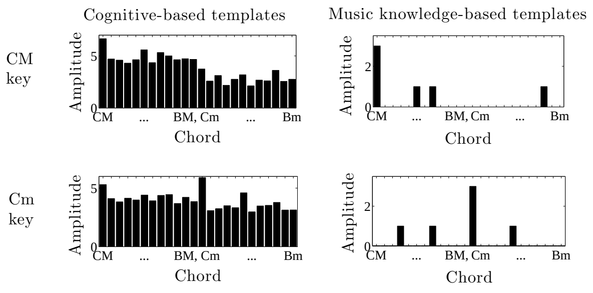


Figure 3. Key templates for chord and key modeling.

4.4 Joint Estimation of Chords and Key

The key can be estimated jointly with the chord progression by simply removing the evidence predicates about key listed in Eq. (6), that give prior information about the key context, and by considering Key as a query along with $Chord$. In addition, we add rules in Table 2 to model key modulations by using the set of formulas:

$$log(p_{ij}^{key}) \quad Key(k_i, t_1) \wedge Succ(t_2, t_1) \wedge Key(k_j, t_2)$$

for all pairs of keys $(k_i, k_j), i, j \in [1, 24]$. The values p_{ij}^{key}, that reflect probability to transit from one key to another, are derived from perceptual tests about proximity between the various musical keys [13]. However, because we focus on global key information in this paper, we manually give a high weight to the formulas corresponding to self-transitions (transition between two same keys) to favor constant key over the analyzed song.

4.5 Inference

The inference step consists of computing the answer to a query, here the chord progression and the key. Specifically, Maximum Probability Explanation (MPE), often denoted as Maximum A Posteriori (MAP) inference, finds the most probable state given the evidence. For inference, we used the toulbar2 branch & bound MPE inference [1], as implemented in the ProbCog toolbox. The graphic interface provided in ProbCog allows convenient editing of the MLN predicates and formulas, which are given as input to the algorithm. The answer to the query can then be directly computed. Although manageable on a standard laptop, the inference step has a high computational cost compared to the baseline algorithm (≈ 2 min (*chord only MLN*), ≈ 4 min (*key MLN*) against 6 sec (HMM, MATLAB) for processing 60s of audio on a MacBook Pro 2.4GHz Intel Core 2 Duo with 2GB RAM).

5. EVALUATION

The proposed model has been tested on a set of hand-labeled Beatles songs, a popular database used for the chord estimation task [11]. All the recordings are polyphonic, multi-instrumental songs containing drums and vocal parts. We map the complex chords in the annotation (such as major and minor 6^{th}, 7^{th}, 9^{th}) to their root triads. The original set comprises 180 Beatles songs but we reduced it to 141 songs, removing songs containing key modulations. The list of this subset can be found in [21].

Label accuracy (*LA*) is used to measure how the estimated chord/key is consistent with the ground truth. The *LA* chord estimation results correspond to the mean and standard deviation of correctly identified chords per song. The *LA* key estimation results indicate the percentage of songs for which the key has been correctly estimated. The results obtained with the various configurations of the proposed model are described in Tables 4 and 5. Paired sample t-tests at the 5% significance level are performed to determine whether there is statistical significance in the observed accuracy results between different configurations.

Table 4. Chords label accuracy (*LA*) results. *HMM*: baseline HMM, *Chord MLN*: chord-only MLN, *Prior key MLN*: MLN with prior key information, using the WMCR and CB key templates, *Joint chord/key MLN*: MLN for joint estimation of chords and key. *Stat. Sig.*: statistical significance between the model *Chord MLN* and others.

	Chord LA	*Stat. Sig.*
HMM	72.49 ± 14.68	**no**
Chord MLN	72.33 ± 14.78	
Prior key MLN, WMCR	$\mathbf{73.00 \pm 13.91}$	**yes**
Prior key MLN, CB	72.22 ± 14.48	no
Joint chord/key MLN	72.42 ± 14.46	no

Table 5. Key label accuracy (*LA*) results. *Joint chord/key MLN*: MLN for joint estimation of chords and key. *DTBM-chroma* and *DTBM-chord*: Direct Template-Based Method. Exact Estimation *EE*, Mirex Estimation *ME* and Exact + Neighbor *E+N* scores. *Stat. Sig.*: statistical significance between the model *Joint chord/key MLN* and others.

	EE	*EE*	*E+N*	*Stat. Sig.*
Joint chord/key MLN	82.27	88.09	94.32	
DTBM-chord	48.59	67.39	89.44	yes
DTBM-chroma	75.35	85.14	95.77	yes

The main interest of the proposed model lies in its simplicity and expressivity for compactly encoding physical content and semantic information in a unified formalism. Results show that the HMM structure can be concisely and elegantly embedded in a MLN. Although the inference algorithms used for each model are different, a song by song analysis shows that chord progressions estimated by the two models are extremely similar and the difference in the label accuracy results is not statistically significant.

To illustrate the flexibility of the MLN formalism, we also tested a scenario where some partial evidence about chords was added by adding evidence predicates of the form $Chord(c_i^{GT}, 0)$, $Chord(c_i^{GT}, 9)$, $Chord(c_i^{GT}, 19)$, $\cdots, Chord(c_i^{GT}, N-1)$, as prior information of 10% of the ground-truth chords $c_i^{GT}, i \in [1, 24]$. We tested this scenario on the song *A Taste of Honey*, for which the *chord only MLN* estimation results are poor. They were increased from 55.69% to 77.04%, which shows how additional evidence can be easily added and have a significant effect.

The MLN formalism incorporates prior information about key in a simple way. The CB key templates are not relevant for modeling chords given a key on our test-set, whereas the results are significantly better with the WMCR templates, that are more consistent with the harmonic/tonal content of our test-set by clearly favoring the main triads given a key. Incorporating prior information about key with minimal model changes improves the chord estimation results, and the difference is significant (Table 4).

In the *Prior key MLN*, coherent chords with the key context are favored, removing some errors obtained with the chord-only MLN. For instance, Figure 4 shows an excerpt of *Eleanor Rigby*, which is in E minor key. Between $24-30s$, the underlying Em harmony is disturbed by passing notes in the voice. The prior key information favors Em chords and reduces these errors. Prior key information can also reduce confusions due to ambiguous mapping. For instance, the song *The Word*, in DM key, contains several Ddom7 chords (D-F#-A-C), which are mapped to DM (D-F#-A) chords in our dictionary. Many of them are estimated as Dm chords with the *chord MLN*, whereas they are annotated as DM chords with the *Prior key MLN*. Introducing prior key information results in chord estimation that is more coherent with the tonal context.

By considering the key as a query, the proposed model can jointly estimate chords and key. Key estimation is based on the harmonic context, while the chords are estimated given a tonal context. Key information slightly improves the chord estimation results, but the difference is not statistically significant (see Table 4). Results in Table 5 show that the tonal context can be fairly inferred from the chords. Song by song analysis shows that harmonically close errors in the chord estimation (such as dominant or subdominant chords) do not affect the key estimation. Indeed, most of the keys are either correctly estimated or correspond to a neighboring key, as indicated by the MIREX 2007 key estimation score [5] (88.09%) and the *N+E* score (94.32%) that includes harmonically close keys [6] .

Following [24, 28], we compare our key estimation results to a *direct template-based method* (DTBM) that can be viewed as applying the Krumhansl-Schmuckler (K-S) key-finding algorithm [13] to the analyzed excerpt. We compute the correlation between a 12-dimensional vector that averages chroma vectors over time and the 24 key templates (*DTBM-chroma*) by Krumhansl. The estimated key is selected as the one that gives the highest value. To com-

[5] 1 for correct key, 0.5 for perfect fifth detection, 0.3 for relative major/minor, and 0.2 for parallel major/minor

[6] Parallel, relative, dominant or subdominant.

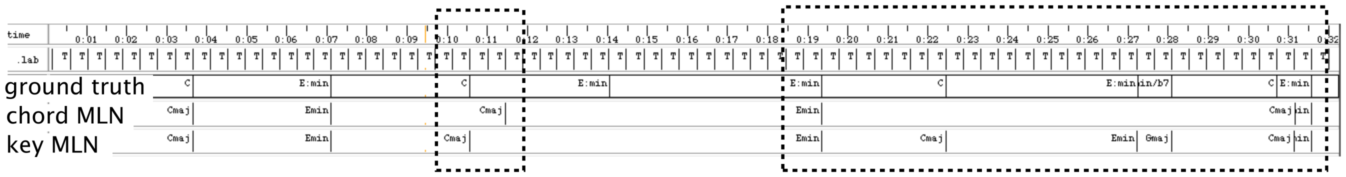


Figure 4. Chord estimation results for an excerpt of the song *Eleanor Rigby*.

pare the performances of the *Prior key MLN* with a baseline algorithm that estimates key from chords after they are predicted, we also report results obtained with a slightly modified version of the K-S algorithm that uses estimated chords instead of chroma: we compute the correlation between a 24-dimensional vector that accumulates the estimated chords over time (considering their duration) and the CB / WMCR templates (*DTBM-chord*)[7]. Results are presented in Table 5. In the *DTBM-chord* approach, errors in the estimation of the chord progression are propagated to the key estimation step, which explains the low *EE* results obtained. The results obtained with *DTBM-chroma* approach are higher, but in both cases, our model performs significantly better than the DTBM methods.

6. CONCLUSION AND FUTURE WORKS

In this article, we have introduced Markov logic networks as an expressive formalism to estimate music content from an audio signal. The results obtained with the *chord MLN* for the task of chord progression are equivalent to those obtained with the baseline *HMM*. Moreover, it allows introducing expert knowledge to enhance the estimation. We have focused on global key information. The model can be extended to local key estimation, which will be the purpose of future work. The proposed model has a great potential of improvement in the future. Context information (such as metrical structure, instrumentation, music knowledge, chord patterns, etc.) can be compactly and flexibly embedded in the model moving toward a unified analysis of music content. Training approaches will be considered. In particular, we will focus on the task of constructing new formulas by learning from the data and creating new predicates by composing base predicates, to compactly capture much more general regularities (predicate invention). As far as we know, Markov logic network have not been used for music content processing yet. We believe that this framework that combines ideas from logic and probabilities opens new interesting perspectives for our field.

7. ACKNOWLEDGMENT

The authors gratefully thank D. Jain for his help.

8. REFERENCES

[1] D. Allouche, S. de Givry, and T. Schiex. Toulbar2, an open source exact cost function network solver. Technical report, INRA, 2010.

[2] I.M. Bajwa. Context based meaning extraction by means of markov logic. *Int. J. Computer Theory and Engineering*, 2(1), 2010.

[3] J.A. Burgoyne, L. Pugin, C. Kereliuk, and I. Fujinaga. A Cross Validated Study Of Modelling Strategies For Auromatic Chord Recognition In Audio. In *ISMIR*, 2007.

[4] J.A. Burgoyne and L.K. Saul. Learning harmonic relationships in digital audio with Dirichlet-based hidden Markov models. In *ISMIR*, 2005.

[5] R. Crane and L.K. McDowell. Investigating markov logic networks for collective classification. In *ICAART*, 2012.

[6] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *IJCAI*, 1999.

[7] T. Fujishima. Real-time chord recognition of musical sound: a system using common lisp music. In *ICMC*, 1999.

[8] L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.

[9] M. Goto. An audio-based real-time beat tracking system for music with or without drum sounds. *J. New Music Res.*, 30(2), 2001.

[10] J.Y Halpern. An analysis of first-order logics of probability. In *IJCAI*, 1989.

[11] C. Harte, M. Sandler, S. Abdallah, and E. Gómez. Symbolic representation of musical chords: a proposed syntax for text annotations. In *ISMIR*, 2005.

[12] D. Jain. Knowledge engineering with markov logic networks: A review. In *KR*, 2011.

[13] C.L. Krumhansl. *Cognitive foundations of musical pitch*. Oxford University Press, New York, NY, USA, 1990.

[14] K. Lee and M. Slaney. Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio. *IEEE TASLP*, 16(2):291–301, 2008.

[15] M. Mauch and S. Dixon. Automatic chord transcription from audio using computational models of musical context. *IEEE TASLP*, 18(6), 2010.

[16] M. Mauch, K. Noland, and S. Dixon. Using musical structure to enhance automatic chord transcription. In *ISMIR*, 2009.

[17] Kevin Murphy, Antonio Torralba, and William T. Freeman. Graphical model for recognizing scenes and objects. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

[18] N.J. Nilsson. Probabilistic logic. *J. Artif. Intell*, 28:71–87, 1986.

[19] K. Noland and Sandler M. Key estimation using a hidden Markov model. In *ISMIR*, 2006.

[20] J.-F. Paiement, D. Eck, S. Bengio, and D. Barber. A graphical model for chord progressions embedded in a psychoacoustic space. In *ICML*, 2005.

[21] H. Papadopoulos. *Joint Estimation of Musical Content Information From an Audio Signal*. PhD thesis, Univ. Paris 6, France, 2010.

[22] H. Papadopoulos and G. Peeters. Large-Scale Study of Chord Estimation Algorithms Based on Chroma Representation and HMM. In *CBMI*, 2007.

[23] H. Papadopoulos and G. Peeters. Joint estimation of chords and downbeats. *IEEE TASLP*, 19(1), 2011.

[24] H. Papadopoulos and G. Peeters. Local Key Estimation from an Audio Signal Relying on Harmonic and Metrical Structures. *IEEE TASLP*, 2011.

[25] J. Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Francisco, CA: Morgan Kaufmann., 1988.

[26] G. Peeters. Beat-marker location using a probabilistic framework and linear discriminant analysis. In *DAFx*, 2009.

[27] M. Richardson and P. Domingos. Markov logic networks. *J. Machine Learning*, 62, 2006.

[28] T. Rocher, M. Robine, P. Hanna, and L. Oudre. Concurrent Estimation of Chords and Keys From Audio. In *ISMIR*, 2010.

[29] M.P. Ryynänen and A.P. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Comp. Mus. J.*, 32(3), 2008.

[30] R. Scholz, E. Vincent, and F. Bimbot. Robust modeling of musical chord sequences using probabilistic N-grams. In *ICASSP*, 2008.

[31] A. Sheh and D.P.W. Ellis. Chord segmentation and recognition using EM-trained HMM. In *ISMIR*, 2003.

[32] P. Singla and P. P. Domingos. Memory-efficient inference in relational domains. In *AAAI*, 2006.

[33] K. Yoshii and M. Goto. A Vocabulary-Free Infinity-Gram Model for Nonparametric Bayesian Chord Progression Analysis. In *ISMIR*, 2011.

[7] We tested several segment durations and chord/key templates and report the results for the best configuration (segment length of 45s).

A GEOMETRIC LANGUAGE FOR REPRESENTING STRUCTURE IN POLYPHONIC MUSIC

David Meredith
Aalborg University, Denmark
dave@create.aau.dk

ABSTRACT

In 1981, Deutsch and Feroe proposed a formal language for representing melodic pitch structure that employed the powerful concept of hierarchically-related pitch alphabets. However, neither rhythmic structure nor pitch structure in polyphonic music can be adequately represented using this language. A new language is proposed here that incorporates certain features of Deutsch and Feroe's model but extends and generalises it to allow for the representation of both rhythm and pitch structure in polyphonic music. The new language adopts a geometric approach in which a passage of polyphonic music is represented as a set of multidimensional points, generated by performing transformations on component patterns. The language introduces the concept of a periodic *mask*, a generalisation of Deutsch and Feroe's notion of a pitch alphabet, that can be applied to any dimension of a geometric representation, allowing for both rhythms and pitch collections to be represented parsimoniously in a uniform way.

1. INTRODUCTION

The problem addressed here is that of designing a formal *coding language* [9, p. 155] for precisely and parsimoniously describing structure in polyphonic music. In general, there are various ways in which a musical pattern can be perceived to be constructed, and a music coding language should be capable of representing these different interpretations. Moreover, it should be possible to compare and evaluate encodings of the different ways of interpreting a musical pattern. The various methods that have been proposed for carrying out such comparisons and evaluations fall into two categories: those based on the *likelihood principle* of preferring the most *probable* interpretations; and those based on the *minimum principle* of preferring the *simplest* interpretations [9, p. 152]. Typically, statistical approaches to musical structure analysis (e.g., [8]) apply the likelihood principle, whereas approaches in the tradition of Gestalt psychology (e.g., [1]) apply the minimum principle. Indeed, van der Helm and Leeuwenberg [9, p. 153] suggest that the fundamental principle of Gestalt psychology, Koffka's [2] law of *Prägnanz*, which favours the simplest and most stable interpretation, can be seen as an "ancestor" of the minimum principle.

The language proposed here is in the tradition of the Gestalt-based coding languages for music proposed by Simon and Sumner [7], Restle [5] and Deutsch and Feroe [1]. It attempts to generalise and extend earlier languages by adopting a geometric approach, along the lines of that proposed by Meredith *et al.* [4]. A passage of polyphonic music is represented in the proposed language as a set of multidimensional points, generated by performing geometric transformations on component patterns. The language introduces the concept of a periodic *mask*, a generalisation of Deutsch and Feroe's notion of a pitch alphabet, that can be applied to any dimension of a geometric representation, allowing for both rhythms and pitch collections to be represented parsimoniously in a uniform way.

A coding language of the type proposed here could be used in music information retrieval to allow for the discovery of patterns in a database that relate to a query pattern on a deeper structural level than the surface. For example, two patterns might be perceived to be related because they have a similar structure but use different pitch alphabets (e.g., where a melody is repeated in a different mode) or because they have the same pitch interval structure but different rhythms. If the music is first encoded in a way that represents these structures, then such relationships can be automatically discovered more efficiently.

2. BACKGROUND

The earliest coding language for music was proposed in 1968 by Simon and Sumner [7]. Simon and Sumner recognized that music is multidimensional and aimed to allow for the description of patterns in melody, harmony, rhythm and form. Their language treats each of these dimensions as an independent series of symbols, chosen from alphabets, for which a compact representation can be derived in terms of certain basic element operators, such as SAME and NEXT. Simon and Sumner defined the notion of a cyclic alphabet and recognized the usual tonal scales and chords as "common" pitch alphabets that are "ordered sets already defined in the culture".

In his experiments on serial pattern learning, Restle [5] found that subjects are particularly good at identifying "runs" (e.g., (2 3 4 5)) and "trills" (e.g., (5 4 5 4)) and tend to use these to segment a series of symbols. Restle explained this by proposing that runs and trills allow for particularly simple generative descriptions. He presented an "E-I theory" wherein a rule consists of a set E of "events" (roughly equivalent to an alphabet) and a set I of "intervals" (in the musical sense). Runs are then the set of products of E-I rule systems in which I contains only one interval. Restle's language uses the sequence operators M (for mirror), T (for transposition) and R (for repeat) for producing compact descriptions of sequences. For example, if the numbers 1 to 6 represent a row of 6 lights that can either be on or off, then the sequence (1 2 1 2 2 3 2 3 6 5 6 5 5 4 5 4) can be encoded as M(T(R(T(1)))). Restle represented structures in his language as hierarchical trees and presented an analysis of the theme of Bach's first Two-part Invention (BWV 772) which describes its tonal and motivic structure.

Deutsch and Feroe's [1] model is in the tradition of the serial pattern languages proposed by Restle [5], Leeuwenberg [3] and Simon [6]. The language can be used to encode arhythmic monophonic sequences of pitches (i.e., neither polyphony nor rhythm can be encoded). *Structures* are defined to be sequences of the elementary operators 'same' (s), 'next' (n) and 'predecessor' (p), that operate over *alphabets*, which are linearly ordered sets of symbols. Structures are decoupled from alphabets. A *sequence*, $\{S; \alpha\}$, is the application of a structure S to an alphabet, α; and a "sequence of notes" (actually a sequence of pitches) can be generated by applying a sequence to a *reference element*. *Compound sequences* can be built up from sequences by using the sequence operators, 'prime' (pr), 'retrograde' (ret), 'inversion' (inv) and 'alternation' (alt). Alphabets can be defined as sequences and 'stacked' hierarchically. For example, the C major scale would be defined as a subset of the chromatic scale (denoted by 'Cr'), using the expression $\mathrm{C} = \{\{(*, 2n^2, n, 3n^2, n); \mathrm{Cr}\}; \mathrm{c}\}$ and the C major triad could be defined relative to the C major scale by the expression $\{\{(*, 2n^2, n^3); \mathrm{C}\}; 1\}$.

The music encoding language presented in the next section extends and generalises these notions of structures and alphabets by re-casting them in geometric terms.

3. A GEOMETRIC MUSIC ENCODING LANGUAGE

In the language proposed here, a passage of music is represented as a set of *notes*. A *note*, n, is an ordered pair, $n = \langle t, p\rangle$, where $t = \mathrm{t}(n)$ is the *onset time* of the note in tatums and $p = \mathrm{p}(n)$ is the note's MIDI note number. A note is therefore a point in *note space* which is the two-dimensional Euclidean integer lattice in which the x co-ordinate represents time in tatums and the y co-ordinate represents pitch in terms of MIDI note number.

A *vector*, v, is an ordered 4-tuple, $v = \langle t, p, \mathbf{M}_\mathrm{t}, \mathbf{M}_\mathrm{p}\rangle$, where $t = \mathrm{t}(v)$ is the *time component* of v, $p = \mathrm{p}(v)$ is the *pitch component* of v, $\mathbf{M}_\mathrm{t} = \mathbf{M}_\mathrm{t}(v)$ is the *time mask sequence* of v and $\mathbf{M}_\mathrm{p} = \mathbf{M}_\mathrm{p}(v)$ is the *pitch mask sequence* of v. $\mathrm{t}(v)$ and $\mathrm{p}(v)$ are integers. $\mathbf{M}_\mathrm{t}(v)$ and $\mathbf{M}_\mathrm{p}(v)$ are *mask sequences*.

```
m($m, i$)
1  if $i$ = nil return nil
2  $j \leftarrow \mathrm{o}(m), k \leftarrow 0$
3  if $i \geq \mathrm{o}(m)$
4      while $j < i$
5          $k \leftarrow k + 1$
6          $j \leftarrow j + \mathrm{s}(m)[(k-1) \bmod |\mathrm{s}(m)|]$
7  else
8      while $j > i$
9          $k \leftarrow k - 1$
10         $j \leftarrow j - \mathrm{s}(m)[k \bmod |\mathrm{s}(m)|]$
11 if $j = i$ return $k$
12 return nil
```

Figure 1. The function $\mathrm{m}(m, i)$, where m is a mask and i is an integer or nil.

```
u($m, i$)
1  if $i$ = nil return nil
2  $j \leftarrow \mathrm{o}(m), k \leftarrow 0$
3  if $i \geq 0$
4      while $k < i$
5          $k \leftarrow k + 1$
6          $j \leftarrow j + \mathrm{s}(m)[(k-1) \bmod |\mathrm{s}(m)|]$
7  else
8      while $k > i$
9          $k \leftarrow k - 1$
10         $j \leftarrow j - \mathrm{s}(m)[k \bmod |\mathrm{s}(m)|]$
11 return $j$
```

Figure 2. The function $\mathrm{u}(m, i)$, where m is a mask and i is an integer or nil.

A *mask sequence*, $\mathbf{M}$, is an ordered set of *masks*, $\mathbf{M} = \langle m_1, m_2, \ldots, m_k\rangle$. A *mask*, m, is an ordered pair, $m = \langle o, s\rangle$, where $o = \mathrm{o}(m)$ is an integer called the *offset* of the mask and $s = \mathrm{s}(m)$ is an ordered set of integers called the *structure* of the mask. Each integer in a mask structure is called an *interval*. If m is a mask and i is an integer, then the function $\mathrm{m}(m, i)$ (see Figure 1) returns the *masked value* of i for the mask m; and the function $\mathrm{u}(m, i)$ (see Figure 2) returns the *unmasked value* of i for the mask m.

Figure 3 illustrates how a mask is used to map a subset of the integers onto the complete set of integers. In the upper part of Figure 3, the mask $\langle 3, \langle 2, 2, 1, 2, 2, 2, 1\rangle\rangle$ is applied. The mask defines a periodic repeated pattern of intervals on the number line such that successive elements in the pattern are mapped onto successive integers. For example, the mask offset, 3, is mapped onto 0. The next integer that does not map onto nil is 5, which therefore maps onto 1, and so on. This particular mask, $\langle 3, \langle 2, 2, 1, 2, 2, 2, 1\rangle\rangle$, can be used to represent the E♭ major scale; and the structure of this mask, $\langle 2, 2, 1, 2, 2, 2, 1\rangle$, can be used to represent the class of all major scales.

Figure 3 also illustrates that the output of one mask can be given as input to another. Thus we can take the range of the mask $\langle 3, \langle 2, 2, 1, 2, 2, 2, 1\rangle\rangle$ and apply the function in Figure 1 to these values with the mask $\langle 4, \langle 2, 2, 3\rangle\rangle$ to give the result shown in the lower part of Figure 3. In

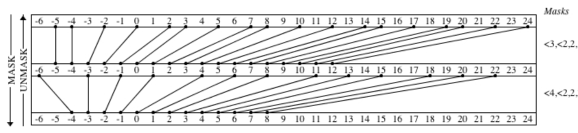


Figure 3. Applying the mask sequence, $\langle\langle 3, \langle 2,2,1,2,2,2,1\rangle\rangle, \langle 4, \langle 2,2,3\rangle\rangle\rangle$.

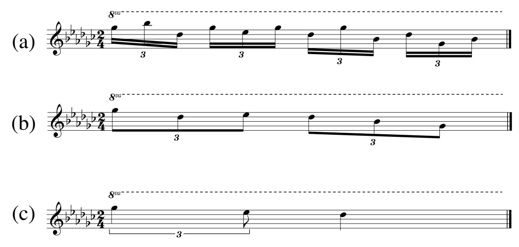


Figure 4. (a) The right-hand part of the first bar of Chopin's *Étude*, Op. 10, No. 5. (b) A plausible rhythmic reduction of (a). (c) A plausible rhythmic reduction of (b).

this particular case, the mask $\langle 4, \langle 2,2,3\rangle\rangle$ can be used to represent a dominant triad in a seven-note scale. Applying this to the output of the mask $\langle 3, \langle 2,2,1,2,2,2,1\rangle\rangle$ therefore gives a representation of the dominant triad in E♭ major—i.e., the B♭ major triad. If the topmost number line in Figure 3 represents MIDI note number, then MIDI pitch 10 maps onto 4 in the middle number line, representing the fact that 10 is the fifth scale degree in E♭ major. The number 4 in the middle line is then mapped onto 0 in the lowest number line, representing the fact that this scale degree is now the root of the dominant triad. Thus, the dominant triad in E♭ major can be represented by the mask sequence, $\langle\langle 3, \langle 2,2,1,2,2,2,1\rangle\rangle, \langle 4, \langle 2,2,3\rangle\rangle\rangle$. This example demonstrates that a mask sequence can be used to represent the notion of hierarchically related pitch alphabets proposed by Deutsch and Feroe [1]. For example, the mask sequence $\langle\langle 3, \langle 2,2,1,2,2,2,1\rangle\rangle, \langle 4, \langle 2,2,3\rangle\rangle\rangle$ corresponds to Deutsch and Feroe's alphabet,

$$\{\{(*, 2n^2, n^3); \{\{(*, 2n^2, n, 3n^2, n); \mathrm{Cr}\}; e\flat\}\}; 5\} .$$

However, unlike Deutsch and Feroe's pitch alphabets, mask sequences can also be used to represent rhythmic and metric structures. For example, the crotchet metric level in a 4/4 bar in which the tatum is a semiquaver can be represented by the mask sequence $\langle\langle 0, \langle 2\rangle\rangle, \langle 0, \langle 2\rangle\rangle\rangle$. Similarly, the dotted crotchet metric level in a 6/8 bar where the tatum is a semiquaver can be represented by $\langle\langle 0, \langle 2\rangle\rangle, \langle 0, \langle 3\rangle\rangle\rangle$.

Figure 4 (a) shows the right-hand part of the first bar of Chopin's *Étude*, Op. 10, No. 5. Figure 4 (b) and (c) show plausible rhythmic reductions of the surface in Figure 4 (a). If we use the triplet semiquaver as the tatum duration, then the rhythm of Figure 4 (b) can be represented by the mask sequence $\langle\langle 0, \langle 2\rangle\rangle\rangle$; and the rhythm of Figure 4 (c) could be represented by the mask sequence $\langle\langle 0, \langle 2\rangle\rangle, \langle 0, \langle 2,1,3\rangle\rangle\rangle$ (or $\langle\langle 0, \langle 4,2,6\rangle\rangle\rangle$).

If $\mathbf{M}$ is a mask sequence and i is an integer, then the function $\mathrm{m}(\mathbf{M}, i)$ (see Figure 5) returns the masked value of i for the mask sequence $\mathbf{M}$ (i.e., the result of applying each of the masks in $\mathbf{M}$, in turn, with an initial input value of i). Conversely, the function $\mathrm{u}(\mathbf{M}, i)$ in Figure 6 returns the unmasked value of i for the mask sequence $\mathbf{M}$.

If v is a vector, then $\mathbf{M}_{\mathrm{t}}(v)$ and $\mathbf{M}_{\mathrm{p}}(v)$ together define the *space*, $\mathcal{M}(v) = \langle \mathbf{M}_{\mathrm{t}}(v), \mathbf{M}_{\mathrm{p}}(v)\rangle$, in which v is defined. If a vector, v, is in note space, then the vector can be written as an ordered pair, $\langle \mathrm{t}(v), \mathrm{p}(v)\rangle$, without specifying the time and pitch mask sequences (e.g., $\langle 2, 3\rangle = \langle 2, 3, \langle\langle 0, \langle 1\rangle\rangle\rangle, \langle\langle 0, \langle 1\rangle\rangle\rangle\rangle$). Given a note n_1 and a vector v, then we can *translate* n_1 by v to give a new note, n_2. In order to do this, n_1 must first be mapped onto a point, q_1, in the space $\mathcal{M}(v)$. q_1 is then translated by v in the usual geometric way to another point, q_2, in $\mathcal{M}(v)$. Finally, q_2 is mapped back onto a note, n_2, in note space. If n is a note and v is a vector, then this process of translation can be accomplished using the algorithm in Figure 7.

Note that, if v is a vector in any space other than note space, then there will not, in general, be a unique vector in note space to which v is equivalent. For example, if $v = \langle 1, 1, \langle\langle 0, \langle 2,3\rangle\rangle\rangle, \langle\langle 0, \langle 3,2\rangle\rangle\rangle\rangle$, then Figure 8 shows that there are 4 distinct vectors in note space to which v might be equivalent, depending on the note that is being translated. A consequence of this is that, in general, there is no unique vector in any space that is equivalent to the sum of two or more vectors that are in different spaces. The

```
m(M, i)
    $k \leftarrow i, j \leftarrow 0$
    **while** $j < |\mathbf{M}|$
        $k \leftarrow \mathrm{m}(\mathbf{M}[j], k)$
        $j \leftarrow j + 1$
    **return** $k$
```

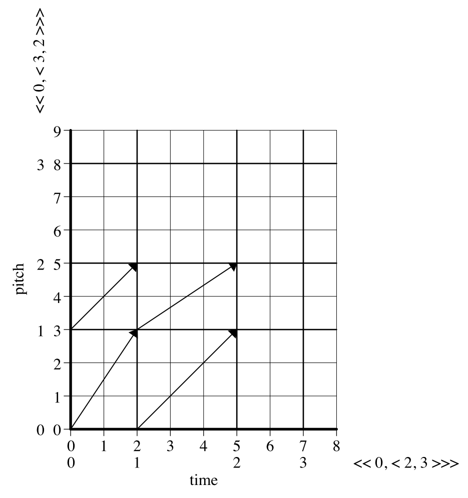
Figure 5. The function $\mathrm{m}(\mathbf{M}, i)$ where $\mathbf{M}$ is a mask sequence and i is an integer.

```
u(M, i)
    $k \leftarrow i, j \leftarrow |\mathbf{M}| - 1$
    **while** $j \geq 0$
        $k \leftarrow \mathrm{u}(\mathbf{M}[j], k)$
        $j \leftarrow j - 1$
    **return** $k$
```

Figure 6. The function $\mathrm{u}(\mathbf{M}, i)$ where $\mathbf{M}$ is a mask sequence and i is an integer.

```
T(n, v)
    $x_1 \leftarrow \mathrm{m}(\mathbf{M}_{\mathrm{t}}(v), \mathrm{t}(n))$
    $y_1 \leftarrow \mathrm{m}(\mathbf{M}_{\mathrm{p}}(v), \mathrm{p}(n))$
    $x_2 \leftarrow x_1 + \mathrm{t}(v)$
    $y_2 \leftarrow y_1 + \mathrm{p}(v)$
    $t_2 \leftarrow \mathrm{u}(\mathbf{M}_{\mathrm{t}}(v), x_2)$
    $p_2 \leftarrow \mathrm{u}(\mathbf{M}_{\mathrm{p}}(v), y_2)$
    **return** $\langle t_2, p_2\rangle$
```

Figure 7. The function $\mathrm{T}(n, v)$ where n is a note and v is a vector.


$\langle\langle 0, \langle 3, 2 \rangle\rangle\rangle$
pitch
9
3 8
7
6
2 5
4
1 3
2
1
0 0
0 1 2 3 4 5 6 7 8
0 1 2 3
$\langle\langle 0, \langle 2, 3 \rangle\rangle\rangle$
time

```
v = <1,1,<<0,<2,3>>>,<<0,<3,2>>>>
T(<0,0>,v) = <2,3>    (v is <2,3> in note space)
T(<2,0>,v) = <5,3>    (v is <3,3> in note space)
T(<0,3>,v) = <2,5>    (v is <2,2> in note space)
T(<2,3>,v) = <5,5>    (v is <3,2> in note space)
```

Figure 8. Equivalent vectors in note space for the vector, $v = \langle 1, 1, \langle\langle 0, \langle 2, 3\rangle\rangle\rangle, \langle\langle 0, \langle 3, 2\rangle\rangle\rangle$.

resultant vector in note space of applying two vectors in succession to a note depends on the note itself. This means that vectors in the sense defined here cannot be added together in the normal way. Instead, if a note is to be translated by the sum of two or more vectors, the vector sum has to be explicitly stated.

Since the sum of two or more vectors is not necessarily equal to a unique vector in any space, it must be considered a different type of object from a vector. We therefore define a special type of object called a *vector sum* to represent a sum of vectors. A vector sum is an *ordered set* of vectors, since vector addition in the sense defined here is not commutative. If we want to denote the sum of the vectors $v_1, v_2, \ldots v_k$, applied *in that order*, then we simply write $v_1 + v_2 + \ldots + v_k$. If w is the vector sum $v_1 + v_2 + \ldots + v_k$, then $|w| = k$, $w[j] = v_{j+1}$ and $w = \sum_{i=1}^{k} v_i$. If w_1 and w_2 are vector sums such that $w_1 = v_{1,1} + v_{1,2} + \ldots + v_{1,m}$ and $w_2 = v_{2,1} + v_{2,2} + \ldots + v_{2,n}$, then $w_1 + w_2 = v_{1,1} + v_{1,2} + \ldots + v_{1,m} + v_{2,1} + v_{2,2} + \ldots + v_{2,n}$. If v is a vector, then $\mathrm{w}(v)$ returns the vector sum that contains just v. In other words, $\mathrm{w}(v)$ typecasts a vector to a vector sum. If w is a vector sum, then we define $\mathrm{w}(w) = w$. To simplify the notation, we allow for vector sums to be added to vectors without explicit typecasting. Thus if v is a vector and w is a vector sum, then $v + w = \mathrm{w}(v) + w$ and $w + v = w + \mathrm{w}(v)$.

As an example, suppose we have three mask sequences and two vectors as follows

$$
\begin{aligned}
\mathbf{M}_0 &= \langle\langle 0, \langle 1 \rangle\rangle\rangle \ , \\
\mathbf{M}_1 &= \langle\langle 0, \langle 2, 2, 1, 2, 2, 2, 1 \rangle\rangle\rangle \ , \\
\mathbf{M}_2 &= \langle\langle 0, \langle 2, 2, 1, 2, 2, 2, 1 \rangle\rangle, \langle 0, \langle 2, 2, 3 \rangle\rangle\rangle \ , \\
v_1 &= \langle 1, 1, \mathbf{M}_0, \mathbf{M}_1 \rangle \ \text{and} \\
v_2 &= \langle 1, 1, \mathbf{M}_0, \mathbf{M}_2 \rangle \ .
\end{aligned}
$$

```
T($n, w$)
1   $n_2 \leftarrow n$
2   for $i \leftarrow 0$ to $|w| - 1$
3       $n_2 \leftarrow \mathrm{T}(n_2, w[i])$
4   return $n_2$
```

Figure 9. The function $\mathrm{T}(n, w)$ where n is a note and w is a vector sum.

```
S($\mathcal{V}$)
1   if $\mathcal{V} = \langle\rangle$ return $\{\}$
2   $w \leftarrow \mathrm{w}(\mathcal{V}[0])$
3   $W \leftarrow \{w\}$
4   for $i \leftarrow 1$ to $|\mathcal{V}| - 1$
5       $w \leftarrow w + \mathcal{V}[i]$
6       $W \leftarrow W \cup \{w\}$
7   return $W$
```

Figure 10. Function for computing the equivalent vector sum set, W, for $\mathcal{V}$, where $\mathcal{V}$ is either a vector sequence or a vector sum sequence.

If we now translate the note $\langle 0, 0 \rangle$ by v_1, then we get the note $\langle 1, 2 \rangle$, that is $\mathrm{T}(\langle 0, 0 \rangle, v_1) = \langle 1, 2 \rangle$. However, we cannot then translate $\langle 1, 2 \rangle$ by v_2 because $\langle 1, 2 \rangle$ is not in the space in which v_2 is defined. If n is a note and w is a vector sum, then the function $\mathrm{T}(n, w)$ in Figure 9 can be used to compute the note that results when n is translated by w. $\mathrm{T}(\langle 0, 0 \rangle, v_1 + v_2)$ is therefore undefined, whereas $\mathrm{T}(\langle 0, 0 \rangle, v_2 + v_1) = \langle 2, 5 \rangle$, illustrating the fact that vector addition is not commutative in this context.

A *vector sequence* is an ordered set of vectors and a *vector sum sequence* is an ordered set of vector sums. For any given vector sequence or vector sum sequence there exists an equivalent *vector sum set* which can be computed using the function in Figure 10. Any run of k identical vectors, v, in a vector sequence can be encoded as kv. For example, $\langle v_1, 3v_2, v_3 \rangle = \langle v_1, v_2, v_2, v_2, v_3 \rangle$. A run of k identical vector sums, w, in a vector sum sequence can similarly be encoded as kw.

If $\mathcal{V}$ is a vector set, vector sum set, vector sequence or vector sum sequence, then the function $\mathrm{W}(\mathcal{V})$, defined in Figure 11 returns the vector sum set that is equivalent to $\mathcal{V}$.

```
W($\mathcal{V}$)
1   if $\mathcal{V}$ is a vector sequence or vector sum sequence
2       $W \leftarrow \mathrm{S}(\mathcal{V})$
3   else if $\mathcal{V}$ is a vector set
4       $W \leftarrow \emptyset$
5       for each $v \in \mathcal{V}$
6           $W \leftarrow W \cup \{\mathrm{w}(v)\}$
7   else
8       $W \leftarrow \mathcal{V}$
9   return $W$
```

Figure 11. Function for computing the equivalent vector sum set, W, for $\mathcal{V}$, where $\mathcal{V}$ is a vector sequence, a vector sum sequence, a vector set or a vector sum set.

```
T(n, V)
1   $W \leftarrow \mathrm{W}(\mathcal{V})$
2   $N \leftarrow \emptyset$
3   for each $w \in W$
4       $N \leftarrow N \cup \{\mathrm{T}(n, w)\}$
5   return $N$
```

Figure 12. Function for translating a note, n, by a vector set, vector sequence, vector sum set or vector sum sequence, $\mathcal{V}$.

```
T(N, V)
1   $N_2 \leftarrow \emptyset$
2   for each $n \in N$
3       $N_2 \leftarrow N_2 \cup \mathrm{T}(n, \mathcal{V})$
4   return $N_2$
```

Figure 13. Function for translating a note set, N, by a vector set, vector sequence, vector sum set or vector sum sequence, $\mathcal{V}$.

A single note, n, or a set of notes, N, can be used to generate a set of notes by translating it by a vector set, a vector sum set, a vector sequence or a vector sum sequence. Figures 12 and 13 show functions for carrying out these types of translation. A vector sum set therefore acts as a generalisation of Deutsch and Feroe's concept of a "structure". Moreover, the combination of a note and a vector sum set to generate a set of notes is a generalisation of Deutsch and Feroe's notion of combining a structure with a reference pitch to generate a sequence of pitches.

The function $\mathrm{P}(\mathbf{X})$, defined in Figure 14 is a generalisation of Deutsch and Feroe's "prime" sequence operator, "pr". The single argument, $\mathbf{X}$, is an ordered set in which each element is a vector set, vector sum set, vector sequence or vector sum sequence. The first step in $\mathrm{P}(\mathbf{X})$ is to convert each element $\mathbf{X}[i]$ into its equivalent vector sum set (lines 1–3). The zero vector sum is then included in each vector sum set, $\mathbf{Y}[i]$ (lines 4–6). $\mathrm{P}(\mathbf{X})$ then returns a set containing a vector sum for each element of the n-ary Cartesian product of the vector sum sets in $\mathbf{Y}$ (lines 7–14). Note that if $\mathbf{A}$ and $\mathbf{B}$ are sequences such that $\mathbf{A} = \langle a_1, a_2, \ldots, a_m\rangle$ and $\mathbf{B} = \langle b_1, b_2, \ldots, b_n\rangle$ then $\mathbf{A} \oplus \mathbf{B} = \langle a_1, a_2, \ldots, a_m, b_1, b_2, \ldots, b_n\rangle$.

Deutsch and Feroe's "ret" and "inv" sequence operators correspond to reflection in the geometric language proposed here: "ret" corresponds to reflection in an axis parallel to the pitch axis, while "inv" corresponds to reflection in an axis parallel to the time axis. The functions,

$$\mathrm{R_p}(v) = \langle \mathrm{t}(v), -\mathrm{p}(v), \mathbf{M}_\mathrm{t}(v), \mathbf{M}_\mathrm{p}(v)\rangle \text{ and}$$
$$\mathrm{R_t}(v) = \langle -\mathrm{t}(v), \mathrm{p}(v), \mathbf{M}_\mathrm{t}(v), \mathbf{M}_\mathrm{p}(v)\rangle$$

reflect vectors in the time axis and pitch axis, respectively. The functions in Figure 15 can be used to reflect vector sums and the functions in Figure 16 can be used to reflect sequences or sets of vectors or vector sums. Note that there is no necessity for a function analogous to Deutsch and Feroe's "alt", because the music is represented as a

```
P(X)
1   $\mathbf{Y} \leftarrow \langle\rangle$
2   for $i \leftarrow 0$ to $|\mathbf{X}| - 1$
3       $\mathbf{Y} \leftarrow \mathbf{Y} \oplus \langle \mathrm{W}(\mathbf{X}[i])\rangle$
4   $v_0 \leftarrow \langle 0, 0\rangle$
5   for $i \leftarrow 0$ to $|\mathbf{Y}| - 1$
6       $\mathbf{Y}[i] \leftarrow \mathbf{Y}[i] \cup \{\mathrm{w}(v_0)\}$
7   $W_1 \leftarrow \mathbf{Y}[0]$
8   for $i \leftarrow 1$ to $|\mathbf{Y}| - 1$
9       $W_2 \leftarrow \emptyset$
10      for each $w_1 \in W_1$
11          for each $w_2 \in \mathbf{Y}[i]$
12              $W_2 \leftarrow W_2 \cup \{w_1 + w_2\}$
13      $W_1 \leftarrow W_2$
14  return $W_1$
```

Figure 14. The function $\mathrm{P}(\mathbf{X})$ where $\mathbf{X}$ is an ordered set in which each element is a vector set, vector sum set, vector sequence or vector sum sequence.

```
R_p(w)
1   $w_2 \leftarrow \mathrm{w}(\mathrm{R_p}(w[0]))$
2   for $i \leftarrow 1$ to $|w| - 1$
3       $w_2 \leftarrow w_2 + \mathrm{R_p}(w[i])$
4   return $w_2$

R_t(w)
1   $w_2 \leftarrow \mathrm{w}(\mathrm{R_t}(w[0]))$
2   for $i \leftarrow 1$ to $|w| - 1$
3       $w_2 \leftarrow w_2 + \mathrm{R_t}(w[i])$
4   return $w_2$
```

Figure 15. Functions for reflection. w is a vector sum.

set of geometrical *points* rather than a *sequence* of *symbols*. There is also no need for a scaling function to represent augmentation or diminution, since this can be accomplished using appropriate time mask sequences.

4. EXAMPLES

Figure 17 shows one of the examples used by Deutsch and Feroe [1, p. 504] to illustrate their model. This pattern can be encoded as $\mathrm{T}(n_1, \mathrm{P}(\langle 3v_1\rangle, \langle v_2\rangle))$ where

$$\begin{aligned}
n_1 &= \langle 1, 60\rangle\ , \\
v_1 &= \langle 1, 1, \mathbf{M}_1, \mathbf{M}_2\rangle\ , \\
v_2 &= \langle -1, -1\rangle\ , \\
\mathbf{M}_1 &= \langle\langle 1, \langle 2\rangle\rangle, \langle 0, \langle 3\rangle\rangle\rangle \text{ and} \\
\mathbf{M}_2 &= \langle\langle 0, \langle 2, 2, 1, 2, 2, 2, 1\rangle\rangle, \langle 0, \langle 2, 2, 3\rangle\rangle\rangle\ .
\end{aligned}$$

Given that the function, $\mathrm{U}(S_1, S_2, \ldots S_k)$ returns the union of sets $S_1, S_2, \ldots S_k$, then the pattern in Figure 4 can be encoded as follows:

$$\mathrm{U}(\mathrm{T}(n_1, \mathrm{P}(\langle v_1\rangle, \langle v_2\rangle)), \mathrm{T}(n_2, \mathrm{P}(\langle v_3\rangle)), \mathrm{T}(n_3, \mathrm{P}(\langle 2v_1\rangle, \langle v_2\rangle)))$$

```
R_p(V)
1  W_1 ← W(V)
2  W_2 ← ∅
3  for each w ∈ W_1
4      W_2 ← W_2 ∪ {R_p(w)}
5  return W_2

R_t(V)
1  W_1 ← W(V)
2  W_2 ← ∅
3  for each w ∈ W_1
4      W_2 ← W_2 ∪ {R_t(w)}
5  return W_2
```

Figure 16. Functions for reflection. $\mathcal{V}$ is a vector set, vector sum set, vector sequence or vector sum sequence.

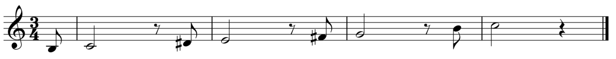


Figure 17. Example pattern used by Deutsch and Feroe [1, p. 504].

where

$$\begin{aligned}
n_1 &= \langle 0, 90\rangle \ , \\
n_2 &= \langle 4, 87\rangle \ , \\
n_3 &= \langle 6, 85\rangle \ , \\
v_1 &= \langle 1, -1, \mathbf{M}_1, \mathbf{M}_2\rangle \ , \\
v_2 &= \langle 1, 1, \mathbf{M}_3, \mathbf{M}_2\rangle \ , \\
v_3 &= \langle 1, 1, \mathbf{M}_3, \mathbf{M}_4\rangle \ , \\
\mathbf{M}_1 &= \langle\langle 0, \langle 2\rangle\rangle\rangle \ , \\
\mathbf{M}_2 &= \langle m_1, \langle 0, s_1\rangle\rangle \ , \\
\mathbf{M}_3 &= \langle\langle 0, \langle 1\rangle\rangle\rangle \ , \\
\mathbf{M}_4 &= \langle m_1, \langle 3, s_1\rangle\rangle \ , \\
m_1 &= \langle 6, \langle 2, 2, 1, 2, 2, 2, 1\rangle\rangle \text{ and} \\
s_1 &= \langle 2, 2, 3\rangle \ .
\end{aligned}$$

Figure 18 shows bars 320–322 from the first movement of Beethoven's Sonata in E♭, Op. 7. This passage can be encoded as $\mathrm{U}(A, B)$ where

$$\begin{aligned}
A &= \mathrm{T}(n_1, \mathrm{P}(\langle 2v_1\rangle, \langle 5v_2\rangle)) \ , \\
B &= \mathrm{T}(n_2, \mathrm{P}(\langle 17\langle 1, 0\rangle\rangle)) \ , \\
n_1 &= \langle 0, 65\rangle \ , \\
n_2 &= \langle 0, 58\rangle \ , \\
v_1 &= \langle 0, 1, \mathbf{M}_1, \mathbf{M}_2\rangle \ , \\
v_2 &= \langle 1, -1, \mathbf{M}_1, \mathbf{M}_2\rangle \ , \\
\mathbf{M}_1 &= \langle 0, \langle 3\rangle\rangle \text{ and} \\
\mathbf{M}_2 &= \langle\langle 3, \langle 2, 2, 1, 2, 2, 2, 1\rangle\rangle, \langle 6, \langle 2, 2, 3\rangle\rangle\rangle \ .
\end{aligned}$$

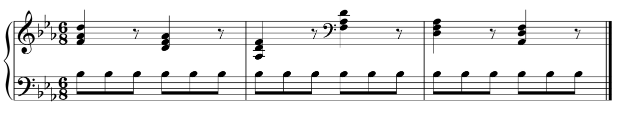


Figure 18. Bars 320–322 of Beethoven's Sonata in E♭, Op. 7, 1st. mvt.

5. CONCLUSIONS AND FUTURE WORK

This paper has introduced a geometric coding language for describing musical structure that extends Deutsch and Feroe's [1] model and recasts it in geometrical terms, allowing rhythmic and pitch structure in polyphonic music to be expressed as transformations on sets of points. The language introduces the concepts of masks, mask sequences and masked spaces which generalise Deutsch and Feroe's notion of hierarchical alphabets to the time dimension, allowing rhythms and pitch collections to be represented parsimoniously in a uniform way. A Java implementation of the language and some extended encoding examples are freely available online. [1]

The primary design goal of the language described here is that it should allow for the formulation of minimal-length descriptions of musical works. There are many ways in which the language could be developed further. For example, decoupling vector co-ordinate values from spaces could permit repetitions of vector co-ordinate value patterns in different spaces to be represented more parsimoniously. There are also cases where structure might be expressed more compactly if pitch class were decoupled from pitch height. Such decoupling of information types and other potentially useful modifications will be explored in the near future. Longer-term goals include

- to develop algorithms for automatically inferring encodings from note sets,
- to develop appropriate measures for description length and
- to explore the relationship between such an encoding language and the way that musical structure is represented cognitively and neurologically.

6. REFERENCES

[1] D. Deutsch and J. Feroe. The internal representation of pitch sequences in tonal music. *Psychol. Rev.*, 88(6):503–522, 1981.

[2] K. Koffka. *Principles of Gestalt Psychology*. Harcourt Brace, New York, 1935.

[3] E. L. J. Leeuwenberg. A perceptual coding language for visual and auditory patterns. *Am. J. Psychol.*, 84(3):307–349, 1971.

[4] D. Meredith, K. Lemström, and G. A. Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *J. New Music Res.*, 31(4):321–345, 2002.

[5] F. Restle. Theory of serial pattern learning: Structural trees. *Psychol. Rev.*, 77(6):481–495, 1970.

[6] H. A. Simon. Complexity and the representation of patterned sequences of symbols. *Psychol. Rev.*, 79(5):369–382, 1972.

[7] H. A. Simon and R. K. Sumner. Pattern in music. In B. Kleinmuntz, editor, *Formal representation of human judgment*. Wiley, New York, 1968.

[8] D. Temperley. *Music and Probability*. MIT Press, Cambridge, MA., 2007.

[9] P. A. van der Helm and E. L. J. Leeuwenberg. Accessibility: A criterion for regularity and hierarchy in visual pattern codes. *J. Math. Psychol.*, 35:151–213, 1991.

[1] Java code available at `http://tinyurl.com/bl7rfgd`, longer examples available at `http://tinyurl.com/blqkgmq`.

UNSUPERVISED LEARNING OF LOCAL FEATURES FOR MUSIC CLASSIFICATION

Jan Wülfing and Martin Riedmiller
University of Freiburg
{wuelfj,riedmiller}@tf.uni-freiburg.de

ABSTRACT

In this work we investigate the applicability of unsupervised feature learning methods to the task of automatic genre prediction of music pieces. More specifically we evaluate a framework that recently has been successfully used to recognize objects in images. We first extract local patches from the time-frequency transformed audio signal, which are then pre-processed and used for unsupervised learning of an overcomplete dictionary of local features. For learning we either use a bootstrapped k-means clustering approach or select features randomly. We further extract feature responses in a convolutional manner and train a linear SVM for classification. We extensively evaluate the approach on the GTZAN dataset, emphasizing the influence of important design choices such as dimensionality reduction, pooling and patch dimension on the classification accuracy. We show that convolutional extraction of local feature responses is crucial to reach high performance. Furthermore we find that using this approach, simple and fast learning techniques such as k-means or randomly selected features are competitive with previously published results which also learn features from audio signals.

1. INTRODUCTION

Automatic categorization of music pieces into categories such as mood, artist or genre is a widely studied topic in music information retrieval. Those categorization tasks basically consist of two steps: feature selection/extraction and classification. Designing and selecting good features for a certain task is demanding and requires expert knowledge about the domain at hand. Nonetheless, a wide range of those hand designed features have been proposed in the past. More recently there has been a growing interest in methods that automatically learn features from data in an unsupervised fashion. Those methods have been very successful on a range of recognition benchmarks for images as well as audio data (see Section 2).

In this work, we investigate the applicability of a k-means based unsupervised feature learning framework that has been used successfully for object recognition in RGB-D images [1] to the problem of genre classification of music pieces.

This framework allows us to fast and flexibly learn local features of different sizes and shapes and to show whether features that span the whole frequency range, only parts of it or even features that cover several consecutive points in time perform best for the task of genre prediction.

To do so, we need to transform the raw audio signal into the time-frequency domain, for which researches have used varying transformations in the past. We determine the influence of this choice by evaluating the feature learning on two different transformations.

After learning an overcomplete dictionary of local features, we extract feature responses in a convolutional manner. Although computationally more expensive, we show that convolutional extraction, together with the right pooling scheme, improves recognition performance significantly.

The paper is structured as follows: In Section 2 we give a short review of approaches that also learn features from audio data, followed by a description of the learning framework in Section 3 and 4. We extensively evaluate the parameters of the learning framework and show its potential on the GTZAN dataset [14] by reporting competitive results in Section 5.

2. RELATED WORK

There is a range of feature learning methods that have been used to tackle music information retrieval tasks e.g. sparse coding [6], principal component analysis [5], deep belief networks [4,8], or a mean-covariance restricted Boltzmann machine [11].

All those feature learning frameworks rely on a transformation of the raw audio signal to the spectral domain. Frequently used is the short time Fourier transformation (with varying window lengths), which can be mel-frequency scaled [5, 11]. Henaff et al. [6] apply the constant-Q transform [12].

Features learned by those approaches differ in size and shape, e.g. some approaches rely on features that cover single time frames [4–6], only parts of the frequency range [6] or even learn time-frequency features that span several consecutive points in time [11].

Learning feature codebooks using simple k-means has a long tradition and has also been applied to audio tasks in [11, 13].

However, Coates et al. [3] just recently found that good image features can be learned using k-means if state-of-the-art image pre-processing and feature encoding is used. This finding could be confirmed and extended for RGB-D images by Blum et al. [1] who used a convolutional bootstrapped k-means procedure to successfully learn RGB-D features for object recognition in "3D" images.

3. LEARNING FEATURE RESPONSES

Our goal is to learn a set of feature responses $D \in \mathbb{R}^{N \times k}$ given a set of input vectors $X = \{x^{(1)}, \ldots, x^{(m)}\}$ with $x^{(i)} \in \mathbb{R}^N$. The input vectors are patches of size $v \times w$ extracted from a training set represented as column vectors. Each value is represented using d channels (e.g. with RGB images $d = 3$, with spectrograms $d = 1$) and hence $N = v \cdot w \cdot d$. Random patches of size $v \times w$ are extracted to build the training set X. Once X is known we apply a pre-processing step followed by the unsupervised learning algorithm.

3.1 Pre-processing

As a pre-processing step we first normalize all patches contained in X by subtracting their mean and dividing by the standard deviation. Afterwards a whitening transformation [7] is applied to the patches. The purpose of the whitening transformation is to ensure that values are decorrelated and have unit variance. This step is crucial to ensure a good quality of the learned feature responses as shown in [3]. We use PCA whitening, which allows us to drop insignificant dimensions from the input data. This results in increased feature extraction speed and can improve feature quality as shown in [1]. If dimensionality reduction is used, we chose to keep the first n components thereby projecting each extracted patch $x \in \mathbb{R}^N$ to a lower dimensional vector $x' \in \mathbb{R}^n$.

3.2 Unsupervised learning

We use a k-means approach to learn k centroids building the feature response dictionary D by clustering the extracted patches X. Although k-means is a very simple unsupervised learning algorithm that is easy to implement, it has recently been shown that it is competitive to other unsupervised learning algorithms when learning local, low-level features from pre-processed image data [3]. Apart from its simplicity the main advantage of using k-means over other algorithms is that it is very fast and scales well to a large amount of centroids. It can therefore be trivially parallelized on current computer hardware in a map-reduce manner and allows us to learn large, over-complete feature dictionaries that can be expensive to learn using other unsupervised learning approaches.

3.2.1 Bootstrapping

To further improve upon the feature quality that can be achieved using standard k-means, as well as the required run time until convergence, we use a bootstrapping learning scheme as proposed in [1] to train the k centroids.

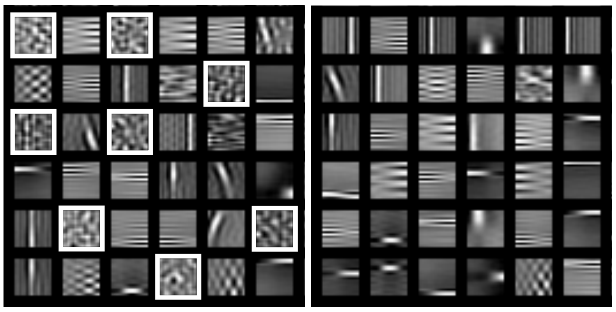

(a) without bootstrapping (b) with bootstrapping

Figure 1: Comparison of 16×16 features learned on the GTZAN dataset using the CQT transform without and with bootstrapping. (a) Without bootstrapping several cluster centers, marked in white, do not represent good feature responses due to the high dimensional space in which k-means clustering is performed. (b) When bootstrapping is enabled all learned centroids correspond to nicely localized features.

We first cluster in the subspace spanned by the first p principal components and fill the learned centroids with zeros for all other $n - p$ dimensions. These centroids are then used to *warm start* the clustering procedure in the n dimensional PCA whitened space.

Without bootstrapping some features are badly localized, which is an artifact of clustering in a high dimensional space (e.g. 256 dimensions if patches of size 16×16 are used). This effect is visible in Fig. 1 where affected features are marked white. When the bootstrapping procedure is used the features are well distributed over the whole feature space by pre-training the centroids on the major principal components. The consecutive clustering procedure in the complete feature space is thus simplified and the badly localized features disappear.

4. FEATURE EXTRACTION

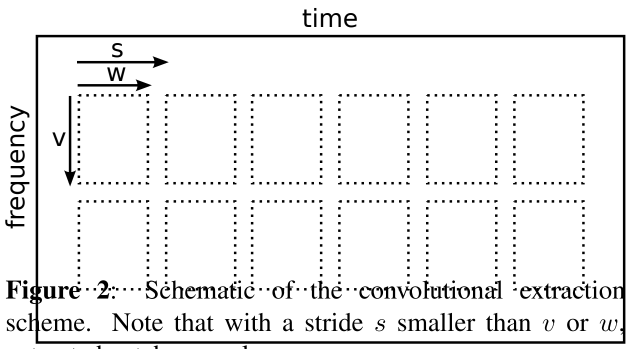


Figure 2: Schematic of the convolutional extraction scheme. Note that with a stride s smaller than v or w, extracted patches overlap.

After learning the dictionary, feature responses are extracted from the input data. Employing a convolutional extraction scheme (see Fig. 2), we traverse the input data with stride s and extract patches at all possible positions. Instead of using standard hard k-means where the feature response $f(x)$ is a sparse vector indicating the closest centroid

$$f_i(x) = \begin{cases} 1 & \text{if } c^i = \arg\min_{c^i \in D} \|c^i - x\| \\ 0 & \text{else} \end{cases}, \quad (1)$$

we compute the triangular response to maximize the information content of each feature response. It keeps the information about the distance of the current patch to all centroids $c^i \in D$ that are closer than the average distance $\mu(z) = \frac{1}{k}\sum_{i=1}^{k} z_i$ where $z \in \mathbb{R}^k$ with $z_i = \|c^i - x\|$. In this case $f(x)$ can be defined as

$$f_i(x) = \max(0, \mu(z) - z_i). \quad (2)$$

4.1 Pooling

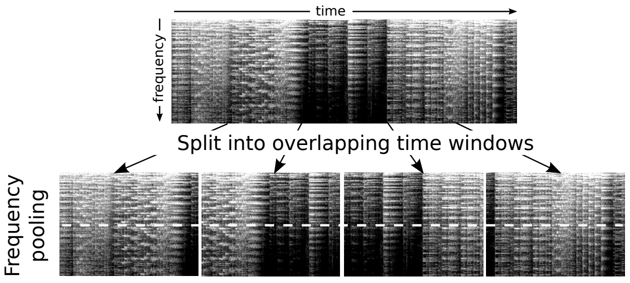


Figure 3: Illustration of the pooling scheme. In the first step, the time-frequency transformed audio signal is split up into overlapping windows. For each window features are extracted and pooled.

Since using all feature responses for classification is computationally expensive - we get a response of size k for each patch that we extract convolutionally - it is common practice to use a pooling scheme to reduce the dimensionality of the feature vector. The term pooling here refers to placing a grid with c cells on the input data and computing a function (e.g. maximum or average) over all feature responses that fall into a grid cell. The dimension of the resulting feature vector is reduced to $c \times k$. For object recognition in images a frequently used grid structure is 2×2. This is a reasonable choice for object recognition tasks where objects are positioned at the center of the image. Here the grid helps to roughly encode the spatial properties of the presented objects in the resulting feature vector. Analogously for audio data, using more than one pool on the time axis results in an encoding of the temporal properties of the feature response. This however is not desired for the task of genre prediction, since characteristic patterns for a certain genre might not always occur at exactly the same time. Additionally we might have to predict the genre based only on a fragment of the song (as it is the case for the GTZAN dataset), underlining the problem that encoding timing may not help, but in fact impair the quality of our prediction. Invariance to timing can be achieved by pooling only once on the time axis (e.g. 2×1). This however may reduce the information content of the feature vector too drastically. To overcome this problem, we split the input data into overlapping time windows of a certain length (similar to [6] and [5]), compute feature responses and pool on each window separately. Each window then serves as input to the classifier and the final result is determined by voting over the classification results of all windows. An illustration of this scheme is depicted in Fig. 3.

5. EVALUATION

We evaluate the performance of the learned features on the GTZAN dataset [14].

5.1 Experimental Setup

5.1.1 Dataset

The GTZAN dataset is organized into 10 distinct genres: Blues, Classical, Country, Disco, Hip hop, Jazz, Metal, Pop, Reggae and Rock. Each genre is represented by 100 song fragments of 30 seconds length.

5.1.2 Pre-processing of audio data

There are several transformations used in the literature to transform the raw audio signal into the time-frequency domain (see Section 2). To determine the influence of this pre-processing choice, we evaluate the feature learning for two different transformations. We apply a short time Fourier transform, calculated on 1024 samples with 512 samples overlap (STFT) and, in a second setting, use the Constant Q-Transform (CQT) [12] spanning 8 octaves, using 64 bins per octave, to create spectrograms of the audio signal. Both spectrograms have exactly the same number of values on the frequency axis (512 values). We also sub sample the CQT to have exactly the same time resolution as the STFT (1292 time frames). This way any possible advantage due to a larger representation can be ruled out.

5.1.3 Classification

For classification we use a linear SVM in a 10-fold cross validation setting.

5.2 Patch dimension and learning techniques

In a first experiment, we show the influence of varying patch sizes and learning techniques on classification accuracy. We learned features using k-means with bootstrapped and randomly initialized cluster centers (chosen at random from the input data). We ran k-means until convergence.

Table 1: Influence of varying patch dimensions and learning methods on classification accuracy. Results are averaged over ten runs of 10-fold cross validation to minimize the influence of random partitioning. The standard deviations are all well below 1%.

Patch size	k-means (random)		k-means (boot.)	
(freq. × time)	STFT	CQT	STFT	CQT
64×1	64.81	70.91	64.72	70.57
128×1	59.59	67.14	68.48	72.19
256×1	65.79	62.12	67.86	70.8
512×1	62.37	66.54	67.69	67.26
16×16	75.2	74.2	75.11	77.7

Features span parts of the frequency axis (64×1, 128×1, 256×1), the whole frequency range (512×1) or frequency and time (16×16). Note that features are extracted convolutionally (with stride 1) if possible which excludes features of size 512×1. In contrast to the smaller patches those features cannot benefit from introducing several pools on the frequency axis, they already span the whole frequency range. That is why we only use one pool in this experimental condition. If not mentioned otherwise, we learn dictionaries of size 800 using PCA whitening and keeping as many components necessary to explain 95% of the variance. Table 1 shows the results of this experiment.

For a small patch size of 64×1 the performance of k-means and bootstrapped k-means is almost equal. Here bootstrapping k-means is unnecessary, since in low dimensions random initialization of the cluster centers suffices. With growing feature size however (e.g. 256×1), random k-means suffers from the effect depicted in Fig. 1, where parts of the dictionary are wasted on ill localized features. Bootstrapping k-means reduces the impact of this problem and affects classification accuracy significantly (e.g. 5.05% improvement with a feature size of 128×1).

Comparing the performance of varying feature sizes, we find that learning features on the whole frequency range (without convolution) has the lowest accuracy, compared to smaller frequency patches. The 16×16 time-frequency features outperform any other setting.

In all settings the STFT accuracies are worse than the results on the CQT transformed audio signal. In addition to the advantages of the Constant-Q transform over the discrete Fourier transform described in [12], we suspect that this is due to the fact that the Constant-Q transform is much sparser and less noisy than the STFT and thereby facilitates learning of good features.

5.3 Pooling and time windows

In this experiment we evaluate the parameters of the pooling scheme described in Section 4.1 used for feature extraction. We employ average pooling in all experiments and vary the number of pools on the frequency axis. We perform experiments on the CQT transformed data. The results for features of size 256×1 and 16×16 (note that both settings share the same number of components) learned with bootstrapped k-means are are shown in Fig. 4a). In all tested settings, increasing the number of frequency pools helps to improve the classification accuracy. Best results are achieved using two to four pools.

In Fig. 4b) the results of varying the length of the time windows are depicted. Accuracy increases with shorter time windows. Depending on the patch size, the optimum is reached with a window length of 1 second (16×16) or 2 seconds (256×1). This finding is in agreement with [5].

5.4 PCA dimensionality reduction and dictionary size

We show the effect of varying the number of principal components kept in Fig. 4c). In the previous experiments, we used exactly as many principal components needed to explain 95 % of the variance, which translates to keeping 88 principal components in case of the 16×16 features and 133 for the 256×1 features. We find that for both feature sizes the highest accuracy can be achieved by setting the number of principal components to 100.

Finally, we evaluate the effect of varying the size of the dictionary learned. In Fig. 4d) the results of varying this number are depicted. Increasing the size of the dictionary steadily improves recognition performance.

Classifier	Features	Accuracy (%)
Linear SVM	Convolutional K-means (16×16) (our)	$\mathbf{85.25 \pm 3.5}$
RBF SVM	DBN [4]	84.3
Linear SVM	PSD on octaves [6]	83.4 ± 3.1
Linear SVM	Convolutional K-means (128×1) (our)	$\mathbf{83.37 \pm 2.54}$
Linear SVM	PSD on frames [6]	79.4 ± 2.8

Table 2: Our results (in bold) compared to previously published results that learn features on the GTZAN dataset. We report the averaged accuracy and standard deviation after one run of 10-fold cross validation.

5.5 Overall performance

To compare our results with previously published results on the GTZAN dataset we learned 1600 features, used 4 frequency pools, time windows of 2 seconds length and kept the first 100 (16×16) and 72 (128×1) principal components. Results are shown in Table 2. With features that span time and frequency, we reach the best result on the GTZAN dataset compared with other approaches that learn features from audio data. There are however approaches that do not learn features in an unsupervised fashion, but focus on sophisticated classifiers and significantly outperform our results (92.7% [2] , 92.4% [9]).

5.6 Additional experiment using random features

Recently randomly selected features were found to perform well on object recognition benchmarks. Saxe et al. [10] attribute the success of random features to the convolutional pooling architecture they are used in. To investigate the role of convolutional feature extraction for audio data, we performed a similar, additional experiment. We chose the same parameters as described in Section 5.5, but instead of learning features, we randomly selected PCA whitened patches without any further clustering and used these as features. Indeed, we found that classification accuracy did not suffer significantly ($85.09\% \pm 3.56$), but the interpretability of the features is lost. This result underlines the importance of convolutional feature extraction.

6. DISCUSSION

Our experiments indicate that convolutional extraction of local feature responses is a viable approach to increase recognition accuracy for the task of genre prediction.

With convolutional extraction there is a trade off between computational complexity and accuracy. Extracting

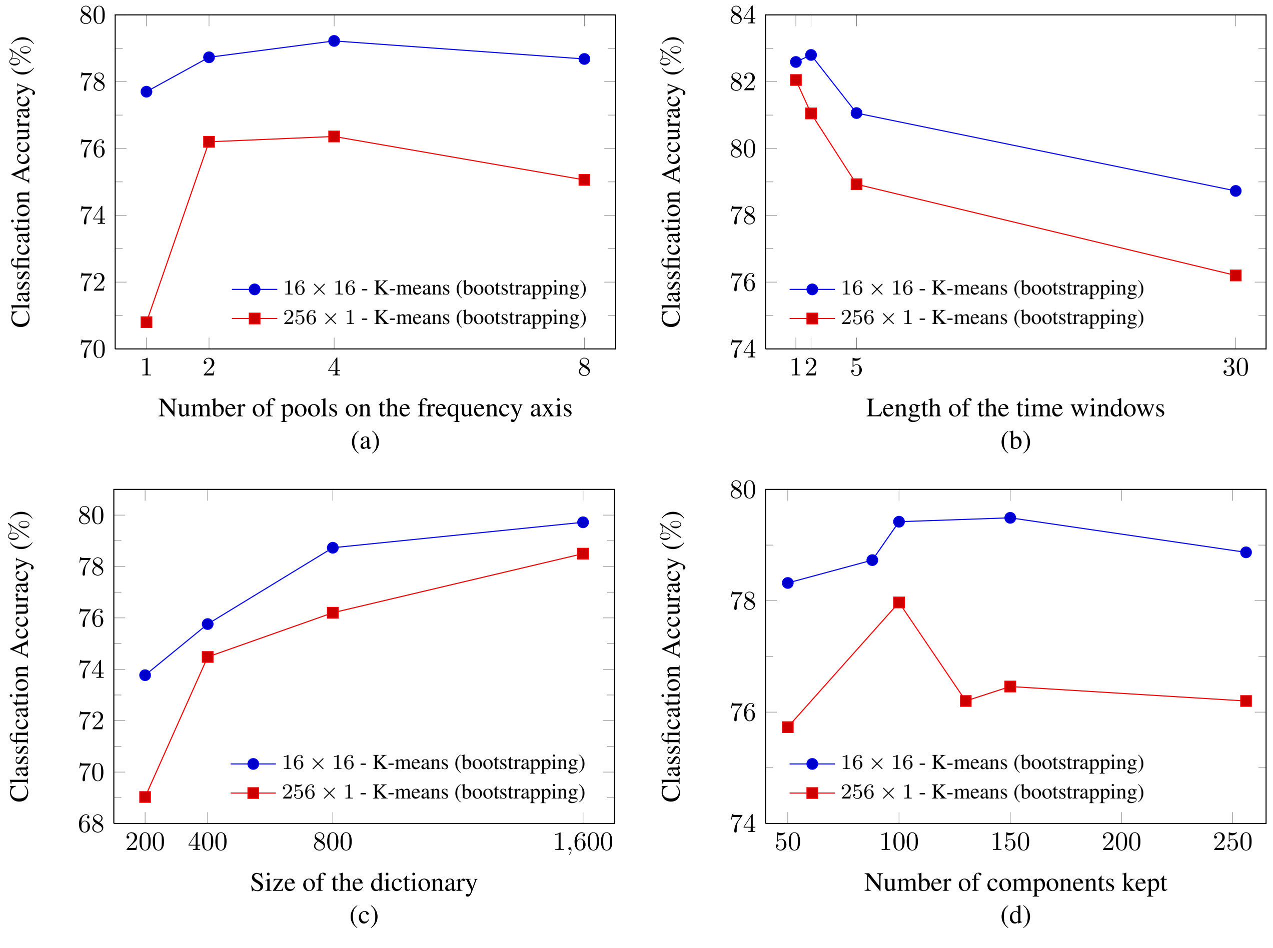


Figure 4: Classification accuracies averaged over ten runs of 10-fold cross validation with (a) varying number of frequency pools, (b) varying time window lengths, (c) varying dictionary size and (d) varying number of components kept.

features convolutionally with a stride of 1 is computationally more expensive than extracting non-overlapping feature responses. To speed up the feature extraction we tried to reduce the size of the spectrograms to a quarter of their original size (128 × 323), which helps twofold. For one, the number of patches that need to be extracted decreases and we are able to learn smaller patch dimensions, which speeds up finding the closest centroids. We found that accuracy did only decrease marginally to $84.77\% \pm 2.6$, when learning patches of size 8×8 on the smaller input. Another way of speeding up the extraction is to increase the stride s. This however has a stronger effect on accuracy, which reduces to $83.45\% \pm 3.3$ (16×16, same setting as in Section 5.5) with a stride of 4.

Another important finding is that time-frequency features perform better in terms of accuracy than frame level features. Nonetheless, features that span the whole frequency range have the advantage of being easily interpretable in musical terms (see Fig. 5 for exemplary features learned only on the frequency axis). This is not the case for local time-frequency patches since the exact frequency is not encoded in those patches. They do however represent patterns of energy distribution over time that can occur at any frequency, e.g. energy remaining constant at one frequency

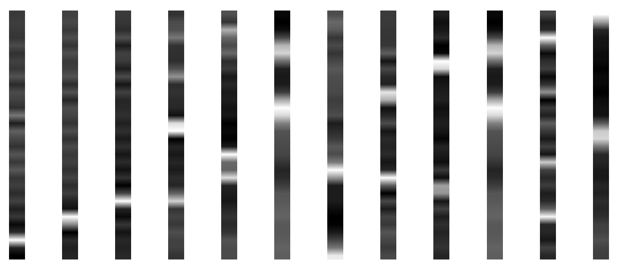


Figure 5: Example features learned on frequency patches 32 × 1 (enlarged for better visualization).

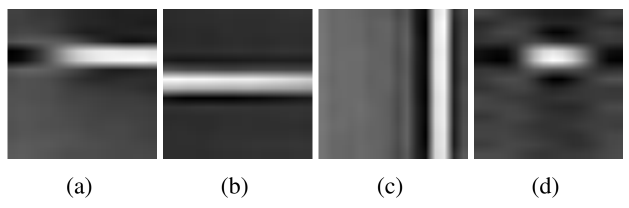


Figure 6: Examples of learned time-frequency features (enlarged for better visualization).

(Fig. 6b), energy spreading across frequencies (Fig. 6c) and note onsets (Fig. 6a and d).

Finally, we show the confusion matrix of the result that was achieved with our best performing features in Fig. 3. Genres that have a low confusion rate include classic, jazz and metal, problematic are rock and pop songs. We believe that the confusion patterns that occur are plausible, e.g. confusing metal with rock songs is a reasonable mistake, since both genres are closely related.

	Bl	Ro	Di	Hi	Ja	Re	Po	Co	Cl	Me
Bl	827	61	1	9	0	20	12	48	0	4
Ro	12	650	55	25	7	25	51	49	0	32
Di	31	36	853	11	2	36	45	12	0	9
Hi	8	5	21	866	0	32	33	0	0	0
Ja	19	6	0	4	961	10	0	12	9	0
Re	48	16	10	22	0	824	15	20	0	0
Po	0	36	33	41	0	18	775	24	0	0
Co	45	110	18	1	0	35	16	830	0	7
Cl	0	5	9	0	30	0	10	1	991	0
Me	10	75	0	21	0	0	43	4	0	948

Table 3: The confusion matrix for ten runs of 10-fold cross validation using features of size 16×16. Genres that have a low confusion rate include classic, jazz and metal, problematic are rock and pop songs.

7. CONCLUSION

In this work, we have presented an approach that predicts the genre of a music piece. We have shown that learning local features using simple and fast techniques like k-means or even randomly selected features is competitive with other more complex learning approaches, if features are extracted convolutionally. We found that time-frequency patches perform better than one dimensional frequency patches and that they reach the highest accuracy to date compared with other learned features on the GTZAN dataset. Furthermore, we have shown that features learned on the CQT transformed audio signal perform better than those learned on the STFT spectrogram. We consider as interesting future work to apply the feature learning to different tasks in the domain of music information retrieval, e.g. auto tagging and also to investigate the possibility of learning a deeper representation on top of the low level features learned so far.

8. REFERENCES

[1] Manuel Blum, Jost Tobias Springenberg, Jan Wülfing, and Martin Riedmiller. A Learned Feature Descriptor for Object Recognition in RGB-D Data. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[2] Kaichun K. Chang, Jyh-Shing Roger Jang, and Costas S. Iliopoulos. Music genre classification via compressive sampling. In *ISMIR*, pages 387–392. International Society for Music Information Retrieval, 2010.

[3] Adam Coates, Honglak Lee, and Andrew Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[4] Philippe Hamel and Douglas Eck. Learning features from music audio with deep belief networks. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, pages 339–344, August 2010.

[5] Philippe Hamel, Simon Lemieux, Yoshua Bengio, and Douglas Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *In Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR11)*, 2011.

[6] Mikael Henaff, Kevin Jarrett, Koray Kavukcuoglu, and Yann LeCun. Unsupervised Learning of Sparse Features for Scalable Audio Classification. In *Proceedings of the International Society for Music Information Retrieval*, 2011.

[7] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–30, 2000.

[8] Honglak Lee, Yan Largman, Peter Pham, and Andrew Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems 22*, pages 1096–1104, 2009.

[9] Yannis Panagakis, Constantine Kotropoulos, and Gonzalo R. Arce. Music genre classification using locality preserving non-negative tensor factorization and sparse representations. In *ISMIR*, pages 249–254. International Society for Music Information Retrieval, 2009.

[10] Andrew Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Ng. On random weights and unsupervised feature learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 1089–1096, June 2011.

[11] Jan Schlüter and Christian Osendorfer. Music Similarity Estimation with the Mean-Covariance Restricted Boltzmann Machine. In *Proceedings of the 10th International Conference on Machine Learning and Applications (ICMLA 2011)*, 2011.

[12] A. Schoerkhuber, C. and Klapuri. Constant-Q transform toolbox for music processing. In *7th Sound and Music Computing*, 2010.

[13] Klaus Seyerlehner, Gerhard Widmer, and Peter Knees. Frame level audio similarity - a codebook approach. In *Proc. of the 11th International Conference on Digital Audio Effects (DAFx-08)*, 2008.

[14] George Tzanetakis and Perry Cook. Musical Genre Classification of Audio Signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

MODELING PIANO INTERPRETATION USING SWITCHING KALMAN FILTER

Yupeng Gu
Indiana University
School of Informatics and Computing

Christopher Raphael
Indiana University
School of Informatics and Computing

ABSTRACT

An approach of parsing piano music interpretation is presented. We focus mainly on quantifying expressive timing activities. A small number of different expressive timing behaviors (constant, slowing down, speeding up, accent) are defined in order to explain the tempo discretely. Given a MIDI performance of a piano music, we simultaneously estimate both discrete variables that corresponds to the behaviors and continuous variables that describe tempo. A graphical model is introduced to represent the evolution of the discrete behaviors and tempo progression. We demonstrate a computational method that acquires the approximate most likely configuration of the discrete behaviors and the hidden continuous variable tempo. This configuration represent a "smoothed" version of the performance which greatly reduces parametrization while retaining most of its musicality. Experiments are presented on several MIDI piano music performed on a digital piano. An user study is performed to evaluate our method.

1. INTRODUCTION

The score of Western classical music is a notation form that contains information such as pitches, durations, and words or symbols that give an abstract reference of how music should be played. It is rather a cartoon like description that misses much detail compared to actual performances. Classical musicians are trained to fill the differences. It is fair to say that the music we hear from CD or concerts have much more information than its corresponding notation. Often people use words such as intention, emotion, expression, interpretation, gesture, phrasing and articulation to describe this extra information. However, from a scientific point of view, these descriptions are vague, subjective and hard to quantify.

In this work, we propose a mathematical approach that aims to create a representation for interpretation. We think of interpretation as having a categorical component to it. This will be the discrete component of our model. We consider interpretation to be performance strategies for different groups of music notes, where the fine details of notes such as inter-onset intervals (IOI) in each group should be strongly correlated and explicitly constrained, instead of modeled independently. We believe this is similar to how musicians think of and communicate about music. Thus our representation will consist of discrete states that describe different performance behaviors and continuous variables that describe tempo and timing in detail. Where the detail will follow the characteristic of the discrete behaviors.

In order to circumvent the difficulty of audio recognition, we chose to use MIDI data for interpretation parsing in this work. The note by note detail and continuous controllers enable using MIDI to create and preserve expressive performances. For instance, we can find many MIDI piano performances from internet that demonstrate expressive interpretation. But relating MIDI data directly to interpretation is still not straightforward. Because the observable aspects of the performance are consequences of hidden interpretive notions. There is a missing layer of ideas that one needs in order to interpret the numbers in MIDI performance. This hidden layer of interpretive constructs guide the timing and volume data. We believe this hidden layer has a close relationship to interpretation and attempt to model it in this work.

This approach has many potential applications. In the area of creating expressive digital music in symbolic form, we have a long standing interest of trying to systematically change a performance meaningfully. It is often that someone had a decent performance recorded where some part of the performance is not fulfilling. If one is not willing or able to repeat the performance until getting better results, the only thing we can do is to modify parameters at the individual note level and hope some combinations might work. This is clearly an unnatural and unmusical way to modify an expressive performance. It would be better to operate on a higher-lever representation of the interpretation that understands notion of gestures and phrases. For instance, when we modify the parameter of a single note, some other parameters will compensate to retain a musical sense.

A musically meaningful representation of interpretation can also be used as a visualization tool. It is often an interesting experience for musicians to listen to a recording of themselves. As a listener, one has a different perspective and judges the performance more objectively. However, listening to a recording is time consuming, and we can only access a small amount of information at one time. Our representation can be used to visualize tempo changes in a discrete way, so musicians can take advantages of their eyes to see and explore an entire performance at once. Furthermore, such visualization can also be used to compare different performances, so it will be easier for musicians to discover how they differ from professionals.

Such representation could also be applied to the expressive rendering problem. With the development of the computer technology, there is a growing interest in generating performance that can match the level of professional musicians. The existing rendering systems are mostly rule-based or case-based. Such systems often include extracting and applying rules with parameters [1] [2] [3]. The advantage of our representation is that it is much lower dimensional than the usual MIDI performance. Hence it is easier to estimate the parameters rather than to estimate all the details for every note. Our representation also has the potential to reduce the unintentional activities from performers which could cause troubles in applying machine learning techniques to performances.

Another possible application of such representation is in creating accompaniment system. A traditional accompaniment system seeks to create a flexible accompaniment to a live soloist that follows the player [4] [5]. For most existing systems, the main focus is to keep up with the soloist as much as possible. Which could inevitably result in overfitting the soloists performance and failing to understand what the player's real intentions are. Good following requires a deeper understanding of the performers intention, thus separating signal from noise. Our representation has the potential to provide a performance model that maintains a certain level of musicality as well as offer enough flexibility. Also, a more advanced accompaniment system may be able to function like a music partner and even teach the player in the future. It is hard to imagine constructing such system without having a layer that can represent the interpretation.

We present a mathematical model in section 2. There is a literature on models that combine discrete state variables with Gaussian variables in fields such as economics, medical science and control engineering [6] [7] [8] [9]. These models are known alternately as Markov jump process, hybrid models, state-space models with switching and switching Kalman filter. We think this type of model suits our purpose of parsing the interpretation of a piano performance. A computation method is introduced in section 3 in order to compute the approximate most likely configuration of the variables in our model. Experiments are presented in section 4 as well as a brief user study that evaluates our model.

2. THE MODEL

We consider only expressive timing in this section. Suppose we have a music performance that contains a sequence of note onset times $o_0, o_1, ...o_N$. Let the score positions associated with the notes be $p_0, p_1, ...p_N$ which are measured in beats. We define four possible types of different behaviors regarding tempo activities. Every event will be labeled as one of the following four behaviors.

α_1. constant speed

In much notated music, especially Western classical music, often tempo marks or beats per measure(BPM) are used to indicate how fast the music should be played. It is clearly impossible for a human being to strictly execute them, but for most of the time these indications are still expected to be respected. There are words such as "rush" and "unstable" that sometimes are used by musicians to describe unintentional tempo change. In our analysis, we want to recognize and fix these unintentional actions.

α_2. slowing down

Although for many sections of music performance constant speed is intended, it is still very unlikely that such speed will be carried consistently though a music piece. An always strict in-tempo performance is often referred to as "mechanical", which is often undesirable and uncommon for Western classical music. Intentional tempo variation within a short time period is a technique that is often used to show expressiveness. Even though certain varying process could be very complicated, it can always be seen as a sequence of basic behaviors. We consider slowing down to be one of them.

α_3. speeding up

We consider speeding up to be the other basic behavior. Combined with α_1 and α_2, these three "devices" can theoretically represent any kind of tempo behaviors.

α_4. Accent (single note behavior)

A common technique to make an accent of a certain note is to take a little extra time before playing that note. Although it can also be seen as a slowing down followed by an immediate speeding up, in this discussion, we would like to model this as an individual behavior for two reasons: 1) Such behavior occurs often; 2) the tempi before and after accents are usually the same. We believe this needs to be modeled explicitly.

So, the possible discrete states for every event are described by the set $\Sigma = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$. Our goal is to label each event o_n with a behavior S_n from Σ. Let $S_1, S_2, ..., S_N$ be the discrete behavior process, $S_n \in \Sigma, n = 1, ..., N$.

We model the sequence of the discrete states as a Markov chain. Figure 1 shows The Markov model. The assumptions are: 1) The states can stay in either constant speed state, slowing down state or speeding up state; 2) Before speeding up, there must be a slowing down process; 3) before slowing down, the performance must be in constant speed; 4) Accent can only happen during constant speed mode and will only last for one note. These assumptions are not necessarily true, we only make our model this way for simplicity.

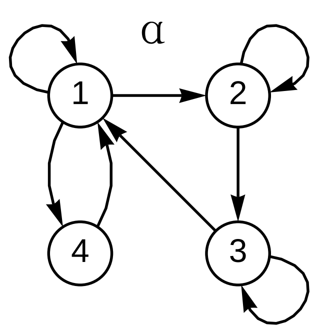


Figure 1. A Markov model showing possible transitions between the discrete states.

This Markov chain is modeled with initial probabilities:

$$I(s_1) = P(S_1 = s_1)$$

s_1 could only be α_1 or α_3 in our model – meaning we only start a performance with a constant tempo or speeding up to a constant tempo.

The transition probability matrix is defined as:

$$R(s_{n+1}, s_n) = P(S_{n+1} = s_{n+1} | S_n = s_n)$$

Now we model the tempo behavior in different states under a switching Kalman filter framework. Let $t_1, t_2, ..., t_N$ and $a_1, a_2, ..., a_N$ be the continuous variables that represent the tempo and acceleration associated with $o_1, o_2, ..., o_N$ repectively, measured in seconds per quarter note. Denote

$$\begin{aligned} X_{n,t} &= t_n \\ X_{n,a} &= a_n \\ X_n &= (t_n, a_n)^T \\ l_n &= p_n - p_{n-1} \end{aligned}$$

Where l_n is the IOI in beats for two consecutive events. We have the initial distribution

$$X_1 \sim N(\begin{pmatrix}\mu_t \\ 0\end{pmatrix}, \begin{pmatrix}\sigma_t^2 & 0 \\ 0 & 0\end{pmatrix}) \qquad |_{S_1=\alpha_1}$$

$$X_1 \sim N(\begin{pmatrix}\mu_t \\ \mu_a\end{pmatrix}, \begin{pmatrix}\sigma_t^2 & 0 \\ 0 & \sigma_a^2\end{pmatrix}) \qquad |_{S_1=\alpha_3}$$

Then we define the different behaviors

$$X_n = X_{n-1} \qquad |_{S_n=\alpha_1, S_{n-1}=\alpha_1} \tag{1}$$

$$X_n \sim N(\begin{pmatrix}\mu_t \\ 0\end{pmatrix}, \begin{pmatrix}\sigma_t^2 & 0 \\ 0 & 0\end{pmatrix}) \qquad |_{S_n=\alpha_1, S_{n-1}=\alpha_3} \tag{2}$$

$$X_n = X_{n-1} \qquad |_{S_n=\alpha_1, S_{n-1}=\alpha_4} \tag{3}$$

$$\begin{aligned} X_{n,a} \sim &\quad N(\mu_a, \sigma_a^2) \\ X_{n,t} = &\quad X_{n-1,t} + l_n X_{n,a} \end{aligned} \qquad |_{S_n=\alpha_2, S_{n-1}=\alpha_1} \tag{4}$$

$$X_n = \begin{pmatrix}1 & l_n \\ 0 & 1\end{pmatrix} X_{n-1} \qquad |_{S_n=\alpha_2, S_{n-1}=\alpha_2} \tag{5}$$

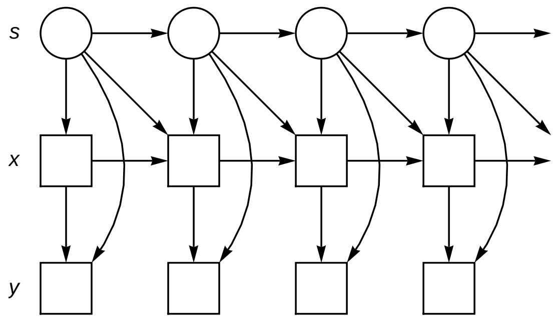


Figure 2. The DAG describing the dependency structure of the variables of our model. Circles represent discrete variables while squares represent continuous variables.

$$\begin{aligned} X_{n,a} \sim &\quad N(-\mu_a, \sigma_a^2) \\ X_{n,t} = &\quad X_{n-1,t} + l_n X_{n,a} \end{aligned} \qquad |_{S_n=\alpha_3, S_{n-1}=\alpha_2} \tag{6}$$

$$X_n = \begin{pmatrix}1 & l_n \\ 0 & 1\end{pmatrix} X_{n-1} \qquad |_{S_n=\alpha_3, S_{n-1}=\alpha_3} \tag{7}$$

$$X_n = X_{n-1} \qquad |_{S_n=\alpha_4} \tag{8}$$

This model forces the tempo to be a constant (but unknown) in each section where the discrete states stay in α_1 or α_4. It also forces an unknown constant acceleration when discrete states in α_2 and α_3. Equation (2) means when the performance comes back from speeding up, the performer will start a new unknown tempo. We denote all the unknown tempi $\{X_{n,t}$ s.t. $S_n = \alpha_1, S_{n-1} = \alpha_3\}$ as $\{\tau_1, ..., \tau_K\}$. Equation (4) means every time when the performance gets into slowing down states, a new unknown acceleration with a positive mean is introduced. Equation (6) means every time when the performance gets into speeding up states, a new unknown acceleration with a negative mean is introduced. We denote all the unknown accelerations $\{X_{n,a}$ s.t. $S_n = \alpha_2, S_{n-1} = \alpha_1$ OR $S_n = \alpha_3, S_{n-1} = \alpha_2\}$ as $\{\gamma_1, ..., \gamma_L\}$.

Now we relate this tempo and acceleration to the observables. Let the IOI $y_n = o_n - o_{n-1}$ for $n = 1, 2, ..., N$ Then the data model is:

$$y_n = l_n X_{n,t} + c_n + \epsilon_n \tag{9}$$

where

$$c_n = 0 \qquad |_{S_n \neq \alpha_4} \tag{10}$$

$$c_n \sim N(\mu_c, \sigma_c^2) \qquad |_{S_n=\alpha_4} \tag{11}$$

$$\epsilon_n \sim N(\mu_\epsilon, \sigma_\epsilon^2) \tag{12}$$

Equation (11) means when the performance comes to an accent, the performer will stretch the IOI with a random length. We denote all the unknown variables $\{C_n$ $s.t.S_n = \alpha_4\}$ as $\{\kappa_1, ..., \kappa_M\}$. Equation (12) represent the observation errors. All the other variables of the model depend deterministically on these variables

$$\tau_1, ..., \tau_K, \gamma_1, ..., \gamma_L, \kappa_1, ..., \kappa_M, \epsilon_1, ..., \epsilon_N$$

The directed acyclic graph(DAG) of the graphical model is represented in figure 2. The model has both discrete and continuous variables. For every configuration of the discrete variables, the continuous variable have a multivariate Gaussian distribution and is a Kalman filter. Thus, the $S_1, .., S_N, X_1, ..., X_N,\ y_1, ..., y_N$ collectively have a conditional Gaussian distribution.

3. COMPUTING THE INTERPRETATION PARSE

We want to simultaneously estimate the discrete state variable $S_1, S_2, ..., S_N$ and the continuous variable tempo $X_1, X_2, ..., X_N$ given the observed IOI data $y_1, y_2, ..., y_n$.

The joint likelihood function can be expressed as

$$L(y, s, x) = I(s_1)P(y_1|x_1, s_1)$$
$$\times \prod_{n=2}^{N} (P(s_n|s_{n-1})P(x_n|x_{n-1}, s_n, s_{n-1})P(y_n|x_n, s_n))$$

We are interested in finding the best configuration of hidden variable S and X that has the greatest probability of giving the observation y.

$$(\hat{s}, \hat{x}) = \arg \max_{s \in S, x \in X} L(y, s, x)$$

Since our model has a linear graph structure described in Figure 2, the maximization problem can be solved using dynamic programming. Using the notation $a_i^j = \{a_i, a_{i+1}, ..., a_j\}$. Let $L_n(y_1^n, s_1^n, x_1^n)$ be the joint likelihood function for variables until observation n, y_1^n, S_1^n, X_1^n for $n = 1, 2, ..., N$. Then we define the density of the optimal configuration for variables until observation n

$$H_n(s_n, x_n) = \max_{s_1^{n-1}, x_1^{n-1}} L_n(y_1^n, s_1^n, x_1^n) \tag{13}$$

Then $H_n(s_n, x_n)$ can be computed recursively

$$H_1(s_1, x_1) = \max_{s_1} I(s_1)P(y_1|x_1, s_1)$$

$$\begin{aligned} H_n(s_n, x_n) = \max_{s_{n-1}, x_{n-1}} & H_{n-1}(s_{n-1}, x_{n-1}) \\ & \times P(s_n|s_{n-1}) \\ & \times P(x_n|x_{n-1}, s_n, s_{n-1}) \\ & \times P(y_n|x_n, s_n)), n = 2, ..., N \end{aligned}$$

We can see that

$$\max_{s_N, x_N} H_N(s_N, x_N) = \max_{s_1^N, x_1^N} L_n(y_1^N, s_1^N, x_1^N))$$

A more detailed description of this method can be found in [10].

It is obvious that the possible state sequences grow exponentially with the number of event N. In order to make the computation tractable, we need to approximate. In this experiment, we use a simple approach that is to sort the current hypotheses on probability densities and leave out the small ones in (13). This method is also known as "beam search".

Once we compute the approximately optimal configuration of discrete states $\hat{s}$. We can recover $\hat{x}$ from the Kalman filter defined by $\hat{s}$.

4. EXPERIMENTS

MIDI is our data format. We use the time stamps directly from MIDI files as the onset times of the notes. All data are collected from a high quality digital piano made by YAMAHA. The reason we choose such an instrument is to ensure that we can hear exactly the same thing as originally recorded when the music is being reproduced. Also when we evaluate our model by modifying the performances and compare them to the original ones, using this instrument can minimize the effect introduced by difference in sound characteristic. The piano keyboard is weighed to simulate the feeling of the real piano keys. According to the 5 pianists who helped creating the data, although it is still not the same as playing a real piano, they can adapt to it and play expressively.

3 sets of experiments are performed:

4.1 Smoothing a Performance

The first set of experiment demonstrates that our model parsimoniously and faithfully represents the original performance.

The data set contains 12 piano excerpts played by graduate level piano major students from the Indiana University Jacobs School of Music. In order to show the generality of our model, the excerpts are selected from composers from different time periods, including Bach, Haydn, Mozart, Beethoven, Schumann, Brahms and Barber. The notated tempi for the excerpts also differ (i.e. there are fast pieces and slow pieces.).

For each excerpt, we have a corresponding MIDI score created from music notation software. Using the method described in [11], we can acquire the music times $\{p_k\}$ in beats for the performance. For each note, we use the time stamp of the MIDI onset as our observation o_n. If several notes are struck at the same time (i.e. a chord), we use the onset time of the first note. μ_t is always set to be the notated tempo. $\sigma_t, \mu_a, \sigma_a, \mu_c, \sigma_c, \epsilon_n$ are manually set to some appropriate value.

Using the method described in section 2 and 3, we can compute the approximate optimal state configuration $\hat{s}$ and corresponding tempo process $\hat{x}_{\bullet,t}$. By reconstructing t, and hence y, from our estimated variables $\hat{x}_{n,t}$, we created a simplified or "smoothed" interpretation. Figure 3 4 5 shows some examples of observed IOIs and "smoothed" IOIs. In each figure, the top plot represents the tempo of the original performance where the bottom plot represents the "smoothed" version. There are many sections of the "smoothed" version that are horizontal lines, which is the behavior of α_1 constant tempo. The "peaks" in the bottom plot show the α_2 slowing down and α_3 speeding up expressive gestures as well as the α_4 accents. We want to see that if the "smoothed" version is approximating the original one with greatly reduced parametrization as well as capturing some of the "important events" and eliminates "unintentional variation".

We use the "smoothed" version to render a MIDI performance with everything else unchanged. Which includes MIDI velocities/note length for each notes and pedaling.

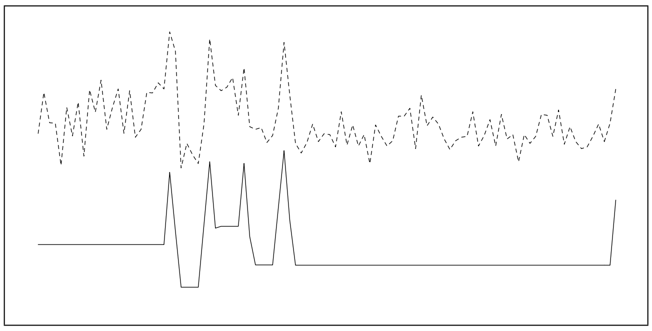


Figure 3. The original tempi and "smoothed" tempi of a performance of Schubert Piano Sonata D959, 1st movement excerpt. (The x-axis represent the music time as in beats. The dotted line represents the original performance. The normal line represents the "smoothed" version. Lines with slope $= 0$ represent state α_1; lines with slope > 0 represent state α_2; lines with slope < 0 represent state α_3; lines with slope $= +\infty$ represent state α_4)

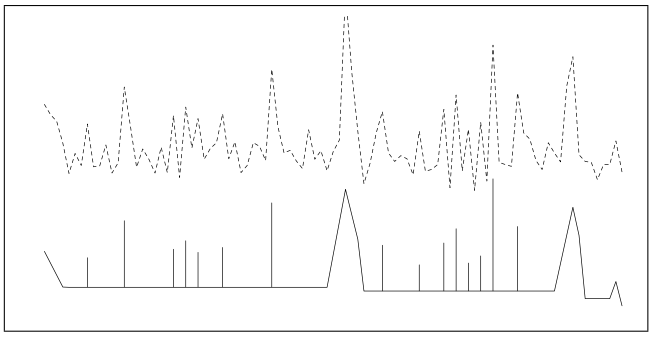


Figure 4. The original tempi and "smoothed" tempi of a performance of Beethoven Piano Sonata Op.31 No.3, 1st movement excerpt. (The lines have the same meaning as in Figure 3)

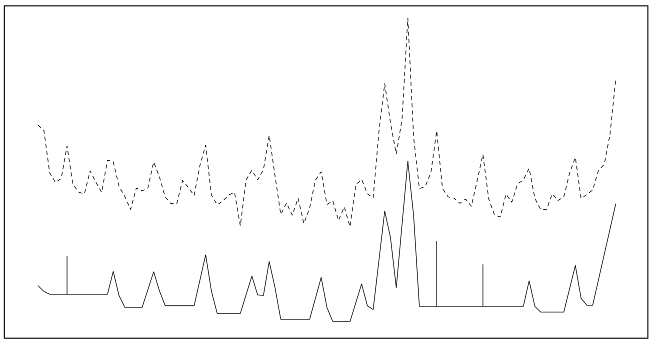


Figure 5. The original tempi and "smoothed" tempi of a performance of Chopin Etude Op.10 No.3 excerpt. (The lines have the same meaning as in Figure 3)

played by	worse	similar	better
professional pianists (4.1)	23	54	31
other musicians (4.2)	4	6	17
dynamic experiment (4.3)	1	7	1

Table 1. Results from the survey of asking 9 subjects about their opinions on "smoothed" version. The total 16 excerpts of the 3 sets of experiments are presented in random order to avoid bias.

Then we perform a simple user study. We present both the original version and the "smoothed" version to 9 participants who were graduate level piano major students. The subjects were presented with random ordering of the two versions of every excerpt. They are asked to choose one from the following options: 1) version 1 is better; 2) version 2 is better; 3) they are about the same. Although what we are really interested in is whether the "smoothed" version is similar to the original performance, we design the questionnaire this way to avoid putting bias towards choosing similar in our subjects' mind.

From the $9 \times 12 = 108$ results that evaluated in this way. The results are shown in table 1 Which shows many of the cases subjects think our "smoothed" version is at least on par with the original version.

4.2 Improving a Performance

The second set of experiment demonstrates that our model can provide a performance standard. If someone has a sense of musicality but lacks piano skills, our model may be able to improve their performance. This experiment differ from the previous one because the amateur piano players play less professionally. So we are testing the "correcting" and "improving" abilities of our model rather than "smoothing". This data set contains several excerpts played by students majoring string performance who knew music well but didn't have serious training in piano performance. We run the exact same procedure as in the first set of experiment and ask participants the same questions. From the results in table 1 we can see subjects think the "smoothed" version is better more often than the excerpts in 4.1, though we do not make inference on the larger population.

4.3 On Dynamics

The third set of experiment demonstrates that dynamics can also be modeled with the conditional Gaussian framework.

We choose Beethoven sonata op.101 1st movement as our material. First we manually partition our music into 3 monophonic voices. For each voice we have a series of MIDI velocities $v_1, v_2, ..., v_n$. Our model for dynamic has two types of behaviors β_1, β_2, in which dynamic can change to a new value or starting a new second order smooth progression.

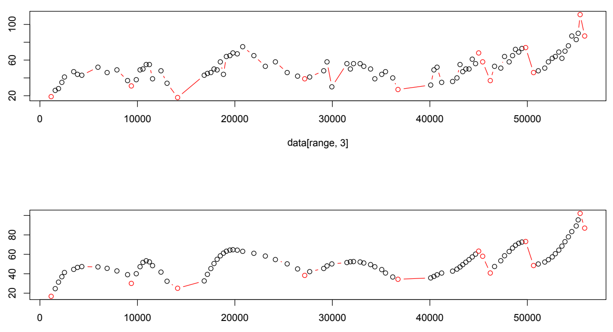


Figure 6. The original dynamics and "smoothed" dynamics of one voice of a performance of Beethoven Sonata Op.101, 1st movement excerpt.(Red dots represent state β_1; Black dots represent state β_2)

Denote

$$Z_n = (d_n, e_n, f_n)^T$$

We define the dynamic behaviors

$$Z_n = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} Z_{n-1}|_{R_{n-1}=\beta_2}$$

$$Z_n \sim N(\begin{pmatrix} \mu_d \\ \mu_e \\ \mu_f \end{pmatrix} \begin{pmatrix} \sigma_d^2 & 0 & 0 \\ 0 & \sigma_e^2 & 0 \\ 0 & 0 & \sigma_f^2 \end{pmatrix})|_{R_n=\beta_2, R_{n-1}=\beta_1}$$

$$Z_n \sim N(\begin{pmatrix} \mu_d \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \sigma_d^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix})|_{R_n=R_{n-1}=\beta_1}$$

Then we relate the model to observations

$$v_n = Z_{n,d} + \delta_n$$
$$\delta_n \sim N(\mu_\delta, \sigma_\delta^2)$$

Using the similar method described in section 3, we compute the "smoothed" dynamics. Figure 6 shows an example of the dynamics before and after the "smoothing".

We use both "smoothed" onset times and dynamics to render a new MIDI performance with everything else unchanged. Then we ask the subjects the same questions. Again, the majority think the "smoothed" version is at least on par with the original performance.

5. DISCUSSION

Although there is no clear evidence showing "smoothed" performances computed from our model are better than original ones, it is still interesting to see that people think they are comparable. It suggests that our model understands the interpretation in a reasonable way.

The direct follow-up of this work is applying the model in accompaniment system. It is challenging to deal with soloist's unintentional activities [11]. We can see from the experiments that our model can reduce the "performance noise". Also, learning the discrete parameters from our model may help with the score following problem since we can model following strategies separately in different parts of music.

In visualization scenario, our current model is only a "toy version" since it only has a limited number of behaviors. For future work, we will explore more "devices" (i.e. more discrete states) that match musicians' intuitive ideas.

Eventually, we also want to use such models for expressive rendering problem. For fine detail of performance, the "devices" may need to be more sophisticated than simple linear models.

We look forward to see more generally useful applications of this model framework as it develops.

6. REFERENCES

[1] Hiraga, R.; Bresin, R.; Hirata, K. & Katayose, H.: Rencon 2004: "Turing Test for Musical Expression" *Proceedings of International Conference on New Interfaces for Musical Expression, in Proc. of NIME 2004*, pp. 120-123, 2004

[2] G. Widmer, S. Flossmann, and M. Grachten, "YQX plays chopin", *AI Magazine*, vol. 30, no. 3, pp. 35-48, 2009.

[3] T. Suzuki, The Second Phase Development of Case Based Performance Rendering System Kagurame, *In Proc. of the IJCAI-03 Rencon Workshop*, 2003, pp. 17-25.

[4] C. Raphael, "Music Plus One and Machine Learning Machine Learning", *Proceedings of the Twenty-Seventh International Conference ICML* 2010.

[5] R. Dannenberg, "An on-line algorithm for real-time accompaniment," in *Proceedings of the International Computer Music Conference, 1984*. Int. Computer Music Assoc., 1984, pp. 193 - 198.

[6] J.D. Hamilton, "A new approach to the economic analysis of nonstationary time series and the business cycle". *Econometrica*, 57:357-384, 1989

[7] R.H. Shumway, D.S. Stoffer, "Dynamic linear models with switching". *J. Amer. Stat. Assoc.*, 86:763-769, 1991

[8] C.J. Kim, "Dynamic linear models with Markov-switching." *J.Econometrics*, 60:1-22, 1994

[9] Z. Ghahramani and G. Hinton. "Variational learning for switching state-space models". *Neural Computation*, 12(4):963-996, 1998.

[10] C. Raphael, "A Mixed Graphical Model for Rhythmic Parsing", *Proceed. of 17th Conf. on Uncertainty in Artif.* Int.462–471, Morgan Kaufmann, 2001.

[11] Y. Gu, C. Raphael, "Orchestral Accompaniment for a Reproducing Piano", *Proceedings of the ICMC09*, Montreal, Canada 501-504, 2009.

INTERPRETING RHYTHM IN OPTICAL MUSIC RECOGNITION

Rong Jin
School of Informatics and Computing
Indiana University, Bloomington
rongjin@imail.iu.edu

Christopher Raphael
School of Informatics and Computing
Indiana University, Bloomington
craphael@indiana.edu

ABSTRACT

We present a method for understanding the rhythmic content of a collection of identified symbols in optical music recognition, designed for polyphonic music. Our object of study is a measure of music symbols. Our model explains the symbols as a collection of voices, while the number of voices is variable throughout a measure. We introduce a dynamic programming framework that identifies the best-scoring interpretation subject to the constraint that each voice accounts for the musical time indicated by the known time signature. Our approach applies as well to the situation in which their are multiple possible hypotheses for each symbol, and thus combines interpretation with recognition in a top-down manner. We present experiments demonstrating a nearly 4-fold decrease in the number of false positive symbols with monophonic music, identify missing tuplets, and show preliminary results with polyphonic music.

1. INTRODUCTION

Throughout the history of the ISMIR community symbolically-represented music has figured prominently in a wide variety of applications, analysis techniques, as well as search and retrieval schemes. In spite of this demonstrated need, symbolic music data are still in short supply, greatly limiting the scale and variety of scientific music research. For music in machine-generated common Western notation, optical music recognition (OMR) provides, in principle, a direct path to create rich and extensive symbolic databases, thus OMR is among the most important problems for the classically-oriented music scientist. Significant advances in core OMR technology would lead to large scale symbolic music libraries, digital music stands, content-based search, as well as many specific applications well-known in this community.

OMR is a deeply challenging problem, well-known to ISMIR stalwarts [1–3], though less well-represented over recent years in published research. Blostein and Baird [4] present a 1992 OMR overview that is not so different from a more current description [5]. One does not work long in this domain without encountering longstanding themes and conflicts of recognition science.

Our strong bias is for *top-down* recognition: approaches that begin by clearly articulating the world of possible hypotheses or answers, then scoring these hypotheses according to their *a priori* plausibility and their agreement with the data. The HMM approach to speech recognition is one of the most famous and successful examples of this kind, combining top-down modeling with computationally powerful dynamic programming (DP) techniques for search and training. However, the real-world recognition problems admitting feasible top-down approaches appear to constitute a small minority. All OMR approaches we know, except [3,6], instead proceed *bottom-up* — beginning with the image data, gradually trying to piece together progressively higher-level constructions, ultimately concluding with the overall interpretation of the data. The preference for bottom-up strategies by nearly all OMR researchers (including ourselves) stems from their computational feasibility. The hallmark of a bottom-up approach is a series of *intermediate* and *irreversible* decisions as one climbs the ladder connecting the image data and its interpretation. The Achilles' heel of the bottom-up paradigm is the inevitable incorrect intermediate decision constituting a blind alley that cannot be retraced. In OMR, the most obvious example would be an incorrect segmentation of the data leading to unrecognizable symbols, though many others exist.

The greatest virtue of the top-down recognition approach is its simultaneous focus on recognition and interpretation — primitive hypotheses such as "note head here" are only considered when they fit into a meaningful interpretation of the scene (say measure) at hand. While it seems too optimistic to hope to formulate OMR in an entirely top-down manner, there are many sub-problems where one can employ this philosophy. We do this whenever possible. While we begin bottom-up, seeking various self-contained objects without regard for their overall organization, each individual search procedure is itself top-down. For example, we find isolated chords (including single notes) by exploring candidate stem locations through grammar-induced DP strategies that consider *every* meaningful chord presentation and result in a globally optimal interpretation. Similarly, we recognize a beamed group by building a grammar of the possible presentations and computing the globally optimal meaningful structure. Further details can be found in [7].

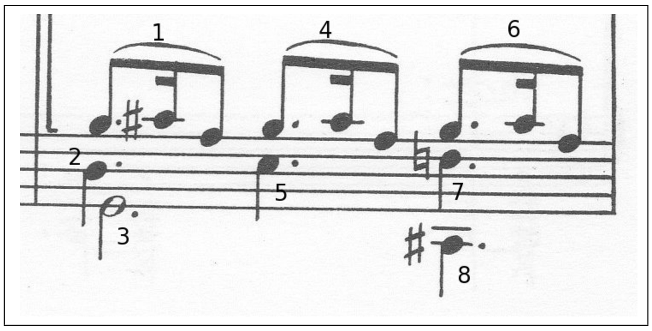


Figure 1. Numbering of symbols for polyphonic rhythm decoding.

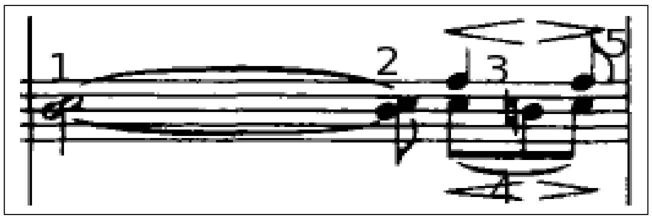


Figure 2. Example of a voice split.

The result of this process is a collection of mutually inconsistent and overlapping hypotheses that share "body parts" in impossible ways — this is a typical pitfall of a bottom-up approach where it is difficult to formulate the concept of a hypothesis' unique "claim" to a particular image pixel. As described in [7], we resolve these conflicts by a phase seeking non-overlapping variants of the identified objects, completely discarding some of them, resulting in a collection of separate symbols that best explain the image data. This is where our present discussion begins.

An OMR system may attempt many different levels of music understanding. The most superficial approach would only record the primitive symbols (note head, stem, ledger line, etc.) found on the page, "punting" on any deeper interpretation of their meaning. Many levels of increasing depth could conceivably be added to this baseline. While we remain uncertain about the right depth of interpretation for our OMR system, it is hard to imagine a useful approach that does not understand rhythm and pitch. Without such interpretation, we cannot even play back the music, significantly limiting the value of the resulting symbolic data. Of these problems rhythm is, by far, the more challenging one.

In the simplest case — single voice music with no unmeasured notes — rhythm interpretation is rather straightforward: symbols can be clearly ordered from left to right, with the onset time of each symbol within a measure given as the sum of all preceding symbol durations. This situation quickly breaks down when the music uses multiple voices on a single staff as in Figure 1. Identifying the onset times here requires that we partition the symbols into three simultaneous voices, thus allowing the application of our monophonic strategy to each voice. Unfortunately, the number of voices is not known *a priori* and frequently changes throughout the duration of the measure as in Figure 2. In this work we propose a method of rhythmic interpretation that understands the music in terms of voices, allowing voices to be added or dropped anywhere in the bar.

Of course, this rhythmic understanding is an essential part of the symbolic data we seek to create, thus valuable in its own right. However, the process of understanding rhythm can be *combined* with the recognition process to improve the accuracy of our recognized results. The simplest example of this idea would leverage the "time signature constraint" — the note values in a voice must add up to the time signature when viewed as a rational number. For example, [8] has used this idea as a post-process to identify potential recognition errors. In this context we consider *multiple* hypotheses for each recognized symbol, choosing the best scoring *overall* measure interpretation obeying this constraint. More generally, we present a model for the possible polyphonic presentations of a measure, seeking the best scoring interpretation generated by the model, given our recognized symbols. The most significant contribution of our present work is a top-down approach for rhythm interpretation that *integrates* recognition with higher-level understanding.

2. RHYTHM DECODING

2.1 Monophonic Rhythm Decoding

The basic processing unit of our system is the measure, whose identification is discussed in [7]. In order to recognize the contents of a measure, we must both correctly segment the measure into meaningful pieces while interpreting the meaning of each piece. We first treat the case of *monophonic* music, here meaning that the notes and rests form a single stream of events. The most typical example would be music played by an instrument that produces a single note at a time, though our approach also applies to sequences of chords, as long as the notes of each chord share a stem.

Suppose we have partitioned the monophonic measure into a sequence of K symbols that can be unambiguously ordered from left to right. These objects could be rests, isolated notes, beamed groups, as well as objects without associated musical time such as clefs. If extraneous symbols, not corresponding to actual document symbols, have been (mis)recognized, it won't matter how these symbols figure in this ordering.

We let S_k be a collection of possible *interpretations* for the kth object. For instance, for a rigid isolated symbol such as a rest, we consider all possible position and label hypotheses, retaining the best scoring position for each label (quarter rest, eighth rest, etc). In the case of an isolated note, our recognition result may involve a closed note head, though an open note head may match nearly as well. Or perhaps we were uncertain about the number of augmentation dots or flags attached to the note. We revisit the DP analysis of the note, modifying the trace-back phase to create the "N-best" interpretations of the object [9]. Thus our isolated note analysis produces a list of possible interpretations along with scores measuring the quality of fit to the image data.

Similarly we construct an N-best list for the interpretations of a beamed group. These hypotheses may differ in the number of beams that connect any pair of adjacent notes, existence of partial beams, or number of augmentation dots attached to a note. In summary, the input to our rhythm recognizer is an ordered list of K objects, each with collection of possible hypotheses, S_k. For each $s_k \in S_k$ we let $D(s_k)$ denote the musical time consumed by the hypothesis, with recognition score $H(s_k)$. Our convention for musical time gives a quarter note duration $\frac{1}{4}$, and eighth note $\frac{1}{8}$, with similar rational numbers for other notes, rests, or beamed groups. In each collection, S_k we include the "null" interpretation with duration and score 0, corresponding to the case of a false positive recognition error.

In many cases we find that the best scoring hypotheses collectively make rhythmic sense. That is, we find that $\sum_k D(\hat{s}_k) = T$ where $\hat{s}_k = \arg\max_{s_k \in S_k} H(s_k)$ and T is the measure's time signature represented as a rational number (e.g. $T = \frac{3}{4}$ for 3/4 time). In such a case there would be no reason to consider any other possible interpretation of the symbols. However, it is common to encounter scenarios where the best scoring hypotheses do *not* "add up," while the correct interpretations of some symbols are found "further down" in the hypothesis list. In such a case it makes sense to look for the sequence $s_1^*, \ldots, s_K^*$ with $s_k^* \in S_k$ given by

$$s_1^*, \ldots, s_K^* = \arg \max_{\sum_k D(s_k) = T} \sum_k H(s_k)$$

where the maximum is taken over all sequences $s_1, \ldots, s_K$ with $s_k \in S_k$ for $k = 1, \ldots, K$.

The identification of this optimal configuration is a simple exercise in dynamic programming. To this end we let P_k denote the possible measure positions for the kth object:

$$P_k = \{\sum_{k'=1}^{k} D(s_{k'}) : s_{k'} \in S_{k'}\}$$

for $k = 1 \ldots, K$, with $P_0 = \{0\}$. These are the "states" of the DP calculation. Then we initialize $M_0(0) = 0$ and recursively define

$$M_k(p_k) = \max_{\substack{p_{k-1} \in P_{k-1}, s_k \in S_k \\ p_{k-1} + D(s_k) = p_k}} M_{k-1}(p_{k-1}) + H(s_k) \quad (1)$$

for $k = 1, \ldots K$ and $p_k \in P_k$. The optimal path, $s_1^*, \ldots, s_K^*$, has score $M_K(T)$ — it is a simple exercise to recover the path that generates the optimal score $M_K(T)$.

By enforcing the time signature constraint on our interpretation we guarantee that the result makes rhythmic sense and fix recognition errors in the process, analogous to the decoding of an error-correcting code.

2.2 Recognizing System Rhythm

A system groups together a collection of staves that are played simultaneously. Usually systems align symbols occurring at the same musical time to the same horizontal position. For instance, corresponding bar lines of a system generally occur at a common horizontal position — in fact, our system recognizer identifies systems by partitioning the staves into groups having common bar line positions. As always with music notation, there are exceptions to this general rule, such as when whole rests are centered rather than "left aligned," or when symbols must be offset from their idealized positions to avoid overlap, as with unison whole notes.

This alignment convention can be used to extend the idea of the preceding section by adding a term to the score function penalizing misalignment of simultaneous events. Suppose we begin with a system of L staves and write $s_1^l, \ldots, s_K^l$ with $s_k^l \in S_k^l$ for an interpretation of the lth staff. As a minor abuse of notation, we write $P(s_k^l) = \sum_{k'=1}^{k} D(s_{k'}^l)$ for the measure position of s_k^l, though clearly $P(s_k^l)$ depends on the entire history leading to s_k^l. For every pair of simultaneous rhythmic events in a system measure — that is, $s_k^l, s_{k'}^{l'}$ with $P(s_k^l) = P(s_{k'}^{l'})$, we penalize their misalignment by $Q(|X(k,l) - X(k',l')|)$, with some non-decreasing function, Q, where $X(k,l)$ gives the horizontal location of the kth event in the lth staff. For a rest, we would take this location to be the horizontal component of its center, while for an isolated note would would take the horizontal component of the note head center, since this is normally what the layout tries to align. For beamed groups we simply use the horizontal component of the first note head center.

We now optimize the criterion:

$$J = \sum_{l=1}^{L} \sum_k H(s_k^l) - \sum_{\substack{l,l'=1 \\ l \neq l' \\ P(s_k^l) = P(s_{k'}^{l'})}}^{L} Q(|X(k,l) - X(k',l')|) \quad (2)$$

subject to the usual time signature constraint on each measure. Due to the very large state space that would ensue, it may not be feasible to perform simultaneous optimization over all $\{s_k^l\}$ variables by DP. A computationally more tractable approach would be the familiar Gauss-Seidel or "coordinate-wise" optimization. That is, we first recognize each staff measure independently according to the technique of the previous section. Then we iteratively revisit each staff in turn, holding the interpretation of the other staff measures fixed while optimizing over the current staff measure. This calculation is possible since, when considering staff l', the measure positions, $P(s_k^l)$, $l \neq l'$, are known since the $s_1^l, \ldots, s_K^l$ are fixed, while for $l = l'$, $P(s_k^l)$ is the DP state.

2.3 Missing Tuplets and Symbol Overloading

Rhythmic notation allows for various abbreviations that may not literally make sense, but are clear in context. Often the correct interpretation is reinforced by the horizontal alignment of coincident symbols, as in the previous section. For instance, it is common to omit the '3' on

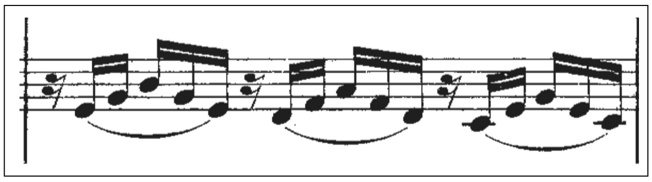


Figure 3. Example of implicit triplets. Our state model requires a triplet to begin on a beat and continue for entire beat before returning to duple rhythm or beginning another triplet.

a beamed group of three notes, when the triplet interpretation is obvious, though this convention also allows for mixing rests and notes implicitly grouped into 3's (or some other tuplet number) as in Figure 3. Another common abbreviated notation uses the half rest or whole rest to denote an empty measure even when the rest doesn't account for the correct bar length. While it may be literally correct to, for instance, write a dotted half rest for a blank measure of 3/4 time, there doesn't seem to be any possibility for misinterpreting the plain half rest, so the shorthand persists.

Examples such as the "overloaded" half rest are easy to treat with the preceding methodology. When the half rest appears as a possible interpretation of a symbol in 3/4 time, we simply add an identically-scored interpretation corresponding to the full length of the measure. The case of the missing triplet on a group of three beamed notes can be handled similarly, allowing both "straight" and triplet interpretations of the group (while in duple meter).

The same ideas can apply in the more complex missing triplets of Figure 3, where the implicit grouping involves several musical symbols. To do this we must multiply our state space by 2 allowing each state to occur in a "straight" and "triplet" version. When we are in a triplet state, all note values count for 2/3 their nominal length. We can leave the triplet state, reverting to the literal interpretation of rhythm, only when the measure position has no factor of 3 in the denominator (i.e. when the triplet is completed). We may also limit the places where triplets can begin (i.e. where we can transition from a non-triplet state to a triplet state) to quarter note or eighth note pulses.

2.4 Polyphonic Rhythm Decoding

As discussed in Section 1, the rhythmic intent of polyphonic notation is often ambiguous, deriving its meaning from implicit use of voices which may appear or disappear at any place within a measure. In this section we present an algorithm for the rhythmic decoding of a measure of polyphonic symbols. For clarity's sake we focus on the simplest statement of the problem, assuming correctly identified symbols, a single staff, and no missing tuplets. However, this technique can be extended using any of the ideas of the previous three subsections. For instance, the ideas of Section 2.1 can be included in an obvious way to cover the case where we have multiple rhythmic hypotheses for each symbol, as in Section 3, with analogous extensions for missing tuplets and staff measures.

We first consider the situation in which the number of voices, V, is known, while the voices persist throughout the entire measure. In such a case, the sum of rhythmic values over all symbols in the measure would be VT. Here the interpretation problem simply separates these symbols into voices, as is necessary for their rhythmic understanding. We begin by numbering the K symbols of the measure from left to right, breaking ties arbitrarily, as in Figure 1: we require only that the resulting sequence of the symbols' measure positions is non-decreasing. We represent a possible interpretation as a sequence of states, one state for each of the K symbols, where a state consists of three quantities for each active voice: the index of the voice's most recent symbol and two rational numbers giving the onset and offset times of the most recent symbol. For instance, the correct state sequence associated with Figure 1 would be:

	voice 1	voice 2	voice 3
1	$(\mathbf{1}, \frac{0}{1}, \frac{3}{8})$	—	—
2	$(1, \frac{0}{1}, \frac{3}{8})$	$(\mathbf{2}, \frac{0}{1}, \frac{3}{8})$	—
3	$(1, \frac{0}{1}, \frac{3}{8})$	$(2, \frac{0}{1}, \frac{3}{8})$	$(\mathbf{3}, \frac{0}{1}, \frac{6}{8})$
4	$(\mathbf{4}, \frac{3}{8}, \frac{6}{8})$	$(2, \frac{0}{1}, \frac{3}{8})$	$(3, \frac{0}{1}, \frac{6}{8})$
5	$(4, \frac{3}{8}, \frac{6}{8})$	$(\mathbf{5}, \frac{3}{8}, \frac{6}{8})$	$(3, \frac{0}{1}, \frac{6}{8})$
6	$(\mathbf{6}, \frac{6}{8}, \frac{9}{8})$	$(5, \frac{3}{8}, \frac{6}{8})$	$(3, \frac{0}{1}, \frac{6}{8})$
7	$(6, \frac{6}{8}, \frac{9}{8})$	$(\mathbf{7}, \frac{6}{8}, \frac{9}{8})$	$(3, \frac{0}{1}, \frac{6}{8})$
8	$(6, \frac{6}{8}, \frac{9}{8})$	$(7, \frac{6}{8}, \frac{9}{8})$	$(\mathbf{8}, \frac{6}{8}, \frac{9}{8})$

This sequence is "legal" since all voices account for the number of beats expressed by the time signature (9/8), as seen by the 3rd member of each voice in the last row of the table.

Of course, the true state sequence is not known, in practice. We proceed by considering *all* possible state sequences, scoring them according to the their plausibility in search of the best scoring candidate. In doing so we generate a search tree where the kth level of the tree treats the kth symbol in our list. At the kth level we expand each branch by adding the kth symbol to all possible voices, while scoring this extension according to several criteria. Perhaps the most important criterion is the degree to which musically coincident symbols align horizontally. When a new symbol enters a voice, we must first consult the state to see if it contains symbols sharing the new symbol's onset time. This is why the symbol's starting position is included as part of the state. For each such coincident symbol in the state, we compute the difference in horizontal position with that of the entering symbol. This is why the state also retains the index of the symbol. The state information can also be used to penalize the addition of a new symbol whose stem direction does not agree with that of the most recent symbol, etc.

The search proceeds over K iterations — one for each incoming symbol, generating a search tree in the process. Each iteration begins by expanding each surviving branch by adding the current symbol to one of the voices, or creating a new voice if available voices exist. These new hypotheses are then scored according to the criteria discussed above. At this point it is possible that we have generated multiple paths to the same state, and, if so, we only retain the best scoring state. That is, we perform DP cutoffs. In doing so, the particular voice numbering is not considered relevant, so two states that differ only by the labeling of voice numbers are considered identical. After performing DP cutoffs, we may still need to prune the tree further to render the search feasible, retaining only the best scoring B hypotheses after each iteration.

Of course, it is not reasonable to assume *a priori* that we *know* the number of voices. For that matter, the number of voices may change throughout the duration of the measure. The most common instance of this phenomena occurs when a multi-voice measure begins or ends with a rest, in which case it is common to use a single rest for all voices. More generally, it is common to allow voices to come in, or go out, of existence when the resulting notation uses less ink and still suggests the right idea to the reader. Figure 2 shows an example where a voice is added midway through the measure (we regard stems with multiple note heads as a single voice).

We address this problem by adding some flexibility to our state production rules. Regardless of the number of voices, we begin each measure with a single voice. At the beginning of each iteration, any voice is allowed to split into two identical voices, as long as some maximum number of voices has not yet been reached. The incoming symbol is then allowed to extend any currently active voice. Additionally, any two voices sharing the same ending time can merge into a single voice. Since we want to discourage gratuitous use of these kinds of productions, we add a penalty term when they are invoked. The correct state sequence associated with Figure 2 is as follows:

	voice 1	voice 2
1	$(\mathbf{1}, \frac{0}{1}, \frac{1}{2})$	—
2	$(\mathbf{2}, \frac{1}{2}, \frac{5}{8})$	—
3	$(2, \frac{1}{2}, \frac{5}{8})$	$(\mathbf{3}, \frac{5}{8}, \frac{7}{8})$
4	$(\mathbf{4}, \frac{5}{8}, \frac{4}{4})$	$(3, \frac{5}{8}, \frac{7}{8})$
5	$(4, \frac{5}{8}, \frac{4}{4})$	$(\mathbf{5}, \frac{7}{8}, \frac{4}{4})$

3. EXPERIMENTS

We tested the algorithm of Section 2.1 on the 2nd movement of the Mozart Quintet for Clarinet and Strings, K. 581. The original images of the four pages of this movement can be seen at

http://www.music.informatics.indiana.edu/papers/ismir12. As with all experiments presented here, we begin by finding our best representation of the image data in terms of non-overlapping isolated symbols, isolated chords, and beamed groups. This phase implicitly segments the image into distinct objects. Using the N-best techniques discussed above, we then identify a list of possible interpretations of each symbol or symbol group, thus forming the input to our rhythm decoder. The best scoring hypothesis for each symbol is superimposed in blue in the referenced images. As discussed above, the collection of best scoring individual hypotheses may not make rhythmic sense, thus we seek the best scoring *meaningful* interpretation through our rhythm decoder.

	Best Score		Rhythm Decoding	
symbol name	**False+**	**False-**	**False+**	**False-**
solid note head	6/898	14/908	5/891	18/908
open note head	1/34	4/37	1/32	6/37
note stem	36/921	10/927	32/913	14/927
1 beam	4/429	9/434	3/427	10/434
2 beam	1/77	4/80	0/76	4/80
3 beam	1/90	2/91	1/91	1/91
aug. dot	113/153	1/39	3/38	4/39
single flag down	0/7	0/7	7/14	0/7
single flag up	0/9	3/12	0/12	0/12
double flag up	0/0	1/1	0/0	1/1
whole rest	27/27	13/13	1/14	0/13
half rest	30/30	0/0	2/2	0/0
quarter rest	12/36	1/25	1/25	1/25
eighth rest	12/30	1/19	2/20	1/19
16th rest	0/1	3/4	0/2	2/4
32th rest	0/6	0/6	1/7	0/6
total	243/2748	66/2603	59/2544	62/2603
decimal	**.088**	.025	**.023**	.024

Table 1. False positives and false negatives for each primitive symbol with and without rhythm decoding. The table shows a nearly 4-fold decrease in false positives with essentially no change in false negatives.

Each of these images was hand-marked with ground truth by identifying bounding boxes of the primitive symbols of Table 1, as well as some rhythmically neutral symbols (clefs, accidentals, etc.) that don't appear in the table. As can be seen from the images and the table, the original recognition contained many small false positive symbols such as augmentation dots and whole/half rests. From a statistical point of view, almost any data model will be prone to such "small symbol" errors, due to the higher variability of small-sample estimates. However, many of these unwanted symbols have only marginal data scores and do not appear in the best scoring measure hypothesis subject to the time signature constraint. In fact, the Table 1 shows a nearly 4-fold decrease in false positives with virtually no change in false negatives. False negatives, for the most part, cannot be corrected by our rhythm decoder, since they stem mostly from errors in which the correct hypothesis does not appear anywhere in our input to the algorithm.

The last page of the Mozart Quintet 2nd movement, visible at the website reference above, contains a number of unmarked triplets, as well as several marked ones, as in Figure 3. We tested the algorithm of Section 2.3 which includes unmarked triplets among the hypotheses that are considered. Since a number of the triplets on our page involve two symbols, a rest and two beamed notes, we must modify our state space in the manner described in Section 2.3, giving two versions of each rhythmic position: "triplet" and "straight." We recognized the page using the rhythm decoder of Section 2.1, both with and without accounting for unmarked triplets. When allowing for triplets we correctly recognized all of the triplets on the page, while the larger associated state space caused no additional errors on the measures that did not contain triplets. This is, of course, a small "proof of concept" experiment, rather than a large scale validation.

A final experiment treats the first page of the Rachmaninov *Etudes Tableaux*, op. 33 for piano, also displayed at the aforementioned web page. Piano music is particularly difficult for OMR, due to the higher symbol density, implicit uses of voices, as well as other idiosyncrasies of keyboard notation. However, the frequent use of implicit voices poses an appropriate challenge for our polyphonic rhythm decoder of Section 2.4 — most measures in the right hand of this page contain two voices.

Our goal now is *simultaneously* to choose from the available hypotheses for each object, and to explain the symbols' rhythm in terms of several possible voices. In this way we *integrate* recognition and interpretation, as is consistent with our philosophy, rather than treating them as distinct phases of OMR. While the page uses voices in a consistent manner (two for the right hand and one in the left), we do not assume this knowledge. Rather we assume a maximum of two voices that are allowed to come and go in each measure, as described in Section 2.4. Since we do not yet recognize time signatures, we assume the time signature is known for each measure.

Evaluating OMR in terms of symbol primitives, as in Table 1, is relatively straightforward and common in the OMR literature. We can imagine various useful notation applications based only on such primitive information, justifying a limited place of this kind of evaluation. However, we expect that most uses of OMR will require a higher level of music understanding than that expressed by symbol primitives. One possible approach to OMR evaluation represents each measure as a list of notes (or notes and rests), with each note having several attributes such as position within measure, length, pitch, coordinates of note head, etc. When both ground truth and recognized results are represented in this manner, a false negative can be identified as any note in the ground truth that cannot be "matched" with a note in the recognized results. Here a match requires agreement of *all* attributes of the ground truth note with one of the recognized notes. False positives are computed by reversing the roles of ground truth and recognized results. Since our current emphasis is on rhythm, we evaluate our approach in this manner describing each note in terms of its note head coordinates and rhythmic onset position within the measure. While not explored here, we believe this general evaluation paradigm (with suitable modifications) is serviceable in a wide range of OMR scenarios.

Using this procedure we achieved false negative and false positive rates of 30/402 and 8/380 on the Rachmaninov page. While evaluations in terms of musical quantities such as pitch and rhythm may better measure the usefulness of the OCR results, they don't clearly convey what actually goes wrong in recognition — in contrast, primitive evaluation is quite specific in this regard. On this one-page test, all errors were due to one of three things. One measure simply had misrecognized rhythm, however, the rhythmic result was quite syncopated, suggesting we may be able to further improve by penalizing unusual rhythm. Most of the false negatives were due to the second kind of error — missing note heads on chords that were otherwise correctly recognized. Our approach cannot possibly recover from such errors. The last type of error results from the unusual figure in the left hand of measures 5-9, in which an eighth and sixteenth are beamed together with a sixteenth rest written in the *interior* of the beamed group. This situation violates our assumption that notes in beamed group are executed in sequence without the possibility of intervening notes/rests from other symbols. We believe this type of error could be corrected with simple modifications of our approach. As before, numerous false positives from recognition are corrected by this procedure.

4. REFERENCES

[1] 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.

[2] D. Bainbridge and T. Bell: "The Challenge of Optical Music Recognition," *Computers and the Humanities* 35: pp. 95-121, 2001.

[3] L. Pugin, J. A. Burgoyne, I. Fujinaga: "MAP Adaptation to Improve Optical Music Recognition of Early Music Documents Using Hidden Markov Models": in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007)*, pp. 513–16, Vienna, Austria, 2007.

[4] D. Blostein and H. S. Baird: "A Critical Survey of Music Image Analysis," In *Structured Document Image Analysis*, ed. H. S. Baird, H. Bunke and K. Yamamoto, pp. 405-34. Berlin: Springer-Verlag, 1992.

[5] A. Rebelo and G. Capela and J. S. Cardoso: "Optical Recognition of Music Symbols," in *International Journal on Document Analysis and Recognition* 13:19-31, 2009.

[6] G. Kopec, P. Chou, D. Maltz: "Markov Source Model for Printed Music Decoding," *Journal of Electronic Imaging*, 5(1), 7-14, 1996.

[7] C. Raphael and J. Wang: "New Approaches to Optical Music Recognition" in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, Miami, USA, pp. 305-310, 2011.

[8] F. Rossant and I. Bloch: "Robust and Adaptive OMR System Including Fuzzy Modeling, Fusion of Musical Rules, and Possible Error Detection," *EURASIP Journal on Applied Signal Processing*, vol: 2007.

[9] R. Schwartz, Y.-L. Chow: "The N-Best Algorithm: An Efficient and Exact Procedure for Finding the Most Likely Sentence Hypotheses," *Proc. of ICASSP-90*, pp. 81–84, Albuquerque, NM, 1990.

ASSIGNING A CONFIDENCE THRESHOLD ON AUTOMATIC BEAT ANNOTATION IN LARGE DATASETS

José R. Zapata[1], **André Holzapfel**[1], **Matthew E.P. Davies**[2], **João L.Oliveira**[2,3] **and Fabien Gouyon**[2,3]
[1]Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
[2]Sound and Music Computing Group, INESC TEC, Porto, Portugal
[3]Faculty of Engineering of the University of Porto, Porto, Portugal
joser.zapata@upf.edu, hannover@csd.uoc.gr, {mdavies, jmso, fgouyon}@inescporto.pt

ABSTRACT

In this paper we establish a threshold for perceptually acceptable beat tracking based on the mutual agreement of a committee of beat trackers. In the first step we use an existing annotated dataset to show that mutual agreement can be used to select one committee member as the most reliable beat tracker for a song. Then we conduct a listening test using a subset of the Million Song Dataset to establish a threshold which results in acceptable quality of the chosen beat output. For both datasets, we obtain a percentage of trackable music of about 73%, and we investigate which data tags are related to acceptable and problematic beat tracking. The results indicate that current datasets are biased towards genres which tend to be easy for beat tracking. The proposed methods provide a means to automatically obtain a confidence value for beat tracking in non-annotated data and to choose between a number of beat tracker outputs.

1. INTRODUCTION

Beat tracking can be considered one of the fundamental problems in music information retrieval (MIR) research. There have been numerous algorithms presented (*e.g.*, [5, 6, 10]) whose common aim is to "tap along" with musical signals. Furthermore the inclusion of beat trackers within other music analysis tasks (such as harmony analysis [8], structural segmentation [11]) has become common-place. However despite the somewhat automatic inclusion of beat trackers as temporal processing components, beat tracking itself is not considered a solved problem. Recent comparative studies of beat trackers suggest there is often little to choose between the best performing state of the art methods [4, 12]. Indeed the viewpoint could be taken that beat tracking performance is approaching a glass ceiling [9] with the current algorithms stagnating at around the 80% mark when evaluated using the least stringent metrics on common datasets [4].

In previous work [9] we proposed that the presence of this apparent glass ceiling was not the result of beat tracking algorithms having reached their full potential, but rather the datasets on which beat trackers are evaluated not containing a sufficient proportion of challenging examples; and that current beat trackers have over-learned the musical properties of the "easier" songs within these datasets. Towards the future advancement of beat tracking we presented a technique to automatically identify challenging examples for beat tracking without the need for ground truth annotations [9]. Our technique was based on measuring the mean mutual agreement (MMA) between a committee of state of the art beat tracking algorithms, where low mutual agreement (or put another way, high disagreement) between beat outputs was shown to be a good indicator of low performance against the ground truth. To this end we empirically determined an MMA "failure" threshold below which beat tracking performance was shown to be very poor, and created a new database comprised of challenging songs with MMA below this threshold.

In this paper we address the opposite issue, where, instead of trying to find where beat tracking algorithms fail, we wish to identify when beat tracking has been successful. When ground truth annotations are available this question can be easily answered, however the problem is non-trivial when no ground truth exists, *i.e.*, on the vast majority of music. The current implicit means for doing so is simply to extrapolate the performance on the limited dataset, for which a precise evaluation can be conducted, and assume this is representative of beat tracking performance on all music.

In light of our previous concerns about the make-up of these annotated databases, we believe that extrapolating performance in this way will be overly optimistic. Therefore when seeking to determine an unbiased measure of performance we can either manually annotate more and more music examples for evaluation, or instead attempt to estimate beat tracking performance without ground truth. Due to the impractical nature of the first option, we pursue the second. Furthermore, if no ground truth is required, then performance can be estimated on very large (effectively unlimited) collections of music.

We extend our previous work to attempt to determine an MMA "success" threshold above which we can have high confidence in the beat tracking output of a committee of state of the art algorithms. We determine the success threshold by means of a subjective listening test, where listeners are asked to rate the quality of the beat output given by the committee across a range of songs for which the MMA has been calculated. In each case the beat tracker output chosen to represent the committee is selected automatically as the one which most agrees with the remainder of the committee, *i.e.*, the beat tracker output with the maximal mutual agreement (MaxMA). We demonstrate that selecting between beat tracker outputs using MaxMA leads to improved performance over consistently picking any individual algorithm from the committee.

Through the calculation of both MMA and MaxMA we present a technique by which we can estimate the level of successful beat tracking on any dataset without ground truth, and, for those songs with MMA above the threshold, automatically annotate the beats in a way that exceeds the performance of the state of the art. In light of the recently presented Million Song Dataset [1] we consider this work to be particularly timely.

The remainder of the paper is structured as follows: Section 2 gives an overview of the proposed method based on mutual agreement and describes the chosen committee. Section 3 demonstrates the improvement in performance when selecting a beat tracker based on the MaxMA approach on a manually annotated dataset. Section 4 applies the technique to non-annotated data and describes the procedure followed in the listening test and the main results. Section 5 concludes the paper with discussion of the results and areas for future work.

2. MEASURING MUTUAL AGREEMENT

The measurement of Mean Mutual Agreement (MMA) is inspired by the Query by Committee concept [14] which selects the most informative set of samples from a database based on the mutual (dis-)agreement between a designated committee of learners. In beat tracking, the MMA is computed using the beat outputs (or beat sequences) of a committee of N beat trackers on a musical piece, by measuring the mutual agreement $MA_{i,j}$ between every pair of estimated beat tracker outputs i and j, and retrieving the mean of all $N(N-1)/2$ mutual agreements. A graphical example is shown in Figure 1.

In addition to calculating the MMA as a summary statistic, we can easily identify the mutual agreement, MA_i, of the beat tracker output i which most agrees with the remainder of the committee: MaxMA, and the beat tracker output i which agrees the least: MinMA. In order to measure the mutual agreement $MA_{i,j}$ between each pair $\{i, j\}$ of beat tracker outputs, a beat tracking evaluation method must be chosen. In [9] we reviewed the properties of existing evaluation methods [2] and selected the Information Gain approach [3] (InfGain) as the only one with a true zero value, able to match low MMA (measured in bits) with unrelated beat tracker outputs:

$$MA_{i,j} = InfGain(i, j), \qquad i, j = 1, \ldots, N \wedge i \neq j. \quad (1)$$

The Information Gain measure is determined by forming a

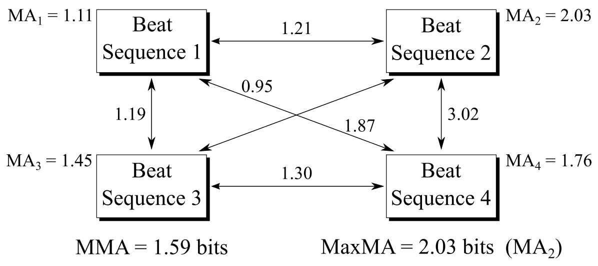


Figure 1: Example calculation of the MMA and MaxMA for a song with the beats estimated from a committee of four beat trackers.

beat error histogram representing the timing error between beat sequences. A numerical score is calculated as a function of the entropy of the histogram. The range of values for the Information Gain is 0 bits to approximately 5.3 bits, where the upper limit is $\log_2(K)$ for K=40 histogram bins. For further details see [3].

To form our committee we select five state of the art and publicly available beat trackers: Dixon (Dix.) [5], Degara (Deg.) [4], Ellis (Ell.) [6], IBT [13], and Klapuri (Kla.) [10]. These convey the performance and diversity necessary to compute a reliable MMA [9].

3. MUTUAL AGREEMENT ON EXISTING ANNOTATED DATA

In order to assess if the mutual agreement among our committee of beat trackers can reliably inform us about the best estimated beat tracker output we computed and compared the outputs of this committee on a manually annotated dataset containing 1360 song excerpts [5,7] (referred to as **Dataset1360**) which covers the following genres: Acoustic; Afro-American; Jazz/Blues; Classical; Choral; Electronic; Rock/Pop; Balkan/Greek; and Samba.

Since we have shown in previous work that disagreement among the committee indicates poor beat tracking performance [9], we consider the potential positive effect of agreement within the committee. Our hypothesis is that the beat tracker that best agrees with the rest of the committee (the one with MaxMA) will be the most reliable algorithm for a specific musical piece. On this basis, we compare the mean ground truth performance of the best overall beat tracker, Best Mean, (which was shown to be Klapuri [10] (Kla.) for Dataset1360 [9]) against the mean scores of the algorithms with the MaxMA and MinMA for each excerpt. To illustrate the upper limit on performance for our committee we also compute the Oracle as the mean score given by the best beat tracker per excerpt.

Figure 2 compares the results of the described performance variants on Dataset1360. As described in Section 2, the MaxMA and MinMA were computed using the InfGain [1]. In order to compare MinMA and MaxMA against

[1] the InfGain and AMLt measures were computed using the the beat tracking evaluation toolbox, available at `http://code.soundsoftware.ac.uk/projects/beat-evaluation`

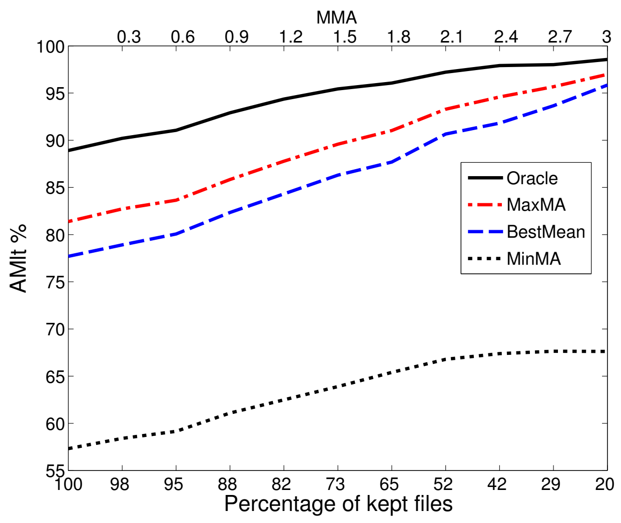


Figure 2: AMLt scores of the beat tracker output with maximum (MaxMA) and minimum (MinMA) agreement per song, compared with the single best beat tracker choice (BestMean), and the oracle score (Oracle) for various thresholds of MMA applied to Dataset1360.

the Best Mean and Oracle performances of the committee on the same data, we used the least stringent continuity-based measure, AMLt [1] (Allowed Metrical Level with no continuity required) [3], where beats are accurate when consecutive falling within tempo-dependent tolerance windows around successive annotations. Beat tracker outputs are also considered accurate if beats occur on the off-beat, or are estimated at double or half the annotated tempo. This performance measure provides a more intuitive scale of 0 to 100% than Information Gain and allows some ambiguity in the choice of metrical level at which the beats are estimated.

Performance across these conditions was computed for different amounts of data confined by incremental values of MMA, in the range of [0-3] bits and varying in steps of 0.3 bits. These MMA values act as a threshold for the selection of excerpts from the dataset (*e.g.*, for an MMA of 2.1 bits we retain 52.1% of the song in the dataset).

As expected, the overall performance of the committee increases with the MMA threshold. This confirms the hypothesis that the MMA is able to reliably detect difficult songs for beat tracking, and therefore can confine the data to easier songs by removing those with low MMA. Across all MMA thresholds we can observe that the performance of MinMA is significantly lower than all other configurations tested. Although lower than the Oracle, MaxMA outperforms the BestMean algorithm, and the difference between the two, around 3.3%, is statistically significantly ($p<0.01$) for all songs with an MMA below 2.4 bits. Above 2.4 bits this difference is no longer significant however the performance of the Oracle, BestMean and MaxMA are all very high. This suggests that for very high MMA thresholds, where beat tracker outputs are highly consistent with one another, any attempt to choose between the members of the committee offers little scope for improvement.

4. AUTOMATICALLY BEAT-ANNOTATING A LARGE DATASET

Having illustrated the validity of using the MaxMA method to select a beat tracker output among a committee of algorithms on a manually annotated dataset, we now turn our attention to applying it to a large collection of non-annotated data. For very large collections it is impractical to expect there to be ground truth annotations on which to base the performance evaluation. Towards understanding how well the state of the art in beat tracking can automatically annotate beats in large collections we employ our MMA and MaxMA methods and attempt to determine the proportion of songs for which the beat estimates are acceptable via a subjective listening test. We want to establish a threshold on MMA above which the beat tracker outputs are perceptually acceptable. For each file, the beat tracker output will be chosen using the MaxMA method.

4.1 Million Song Subset

The large collection we aim to automatically annotate is the **MillionSongSubset** from the Million Song Dataset [1]. The subset is comprised of 10,000 songs without ground truth for which audio previews were obtained. The majority of audio previews were either 30 s or 60 s in duration, however to provide sufficiently long song excerpts for beat tracking we discarded any shorter than 20 s. This left a set of 9940 songs on which to automatically annotate beats. To complement the audio data, we obtained 31696 Last.fm [2] tags which covered a subset of 4638 songs.

Once all of the audio and meta data was collected we ran the committee of beat tracking algorithms recording the MMA value per excerpt and saving the MaxMA beat tracker output.

4.2 Subjective Listening Test

The aim of our listening test was to determine an MMA threshold above which the beat tracker output given by the MaxMA method was deemed acceptable to human listeners. By subsequent inspection of the number of songs in the dataset above this MMA threshold we could then estimate the proportion for which beat tracking can be considered successful.

Just as it is not possible to hand annotate beats in nearly 10,000 songs, it is equally impractical to ask participants to listen and rate this large number. As alternative to the exhaustive rating of all audio songs, we selected 8 levels of MMA = [0.5, 1.0, 1.5, …, 4.0] bits and chose the 6 closest songs from the MillionSongSubset to each MMA level, giving a total of 48 songs to summarize the dataset. To create the musical stimuli for the listening test we constructed stereo audio files containing a mixture of source audio and the MaxMA beat output synthesized as short click sounds. To mitigate the effect of errors in beat tracking at the start of songs, which might bias the listener ratings, each musical stimulus was formed out of the middle 15 s of each

[2] `http://labrosa.ee.columbia.edu/millionsong/lastfm`

song. To allow listeners to hear the audio with and without click sounds, we panned the source audio on its own on the left channel, and on the right channel we mixed the click sounds conveying the beats with a quiet version of the source audio. Through informal listening tests prior to the main experiment, this was deemed an acceptable method for creating the stimuli.

To take the listening test we recruited 25 participants (21 male, 4 female) with an age range of 23 to 41 (mean = 31 years, std = 4.7 years). The participants' level of music training ranged from 0 to 20 years (mean = 8.7 years, std = 7.7 years). Each participant was instructed to perform the test in a quiet environment with good quality headphones. Prior to starting the main test, the participants were given three training examples (not in the main set of 48). The training phase was used for three reasons: *i)* to familiarise participants with the type of musical stimuli in the test, *ii)* for the participants to understand the panning of the beats in the stimuli and *iii)* so the participants could set the playback volume to a comfortable level. To prevent order effects in the stimuli, each participant was given an individual playlist of songs in a different random order.

In taking the test, the participants were asked to answer the following question: "*How do you rate the overall quality of the given click as a beat annotation of the piece?*" The options for rating were: 1 - Bad, 2 - Poor, 3 - Fair, 4 - Good, 5 - Excellent.

4.3 Results

4.3.1 Listening Test

Figure 3 presents a comparison between the human ratings and the MMA of our committee of beat trackers for the selected 48 pieces of the MillionSongSubset. The plot shows that for an MMA equal to 1.5 bits the mean rating was 3.7 (Good) with a standard deviation of 0.93. However, for MMA equal to 1 bit, the mean rating was much lower, at around 2.4 (Poor). Performing a t-test, we found the difference between the mean ratings at these MMA values to be highly significant ($p < 0.0001$). On this basis we can easily identify an MMA threshold of 1.5 bits which separates perceptually acceptable beat tracking from inaccurate beat tracking.

4.3.2 MMA Threshold

By selecting an MMA of 1.5 bits as a threshold of perceptual confidence for beat tracking we find 996 songs (73%) in Dataset1360 and 7252 songs (coincidentally also 73%), in the MillionSongSubset above this limit (see Figure 4). Table 1 shows the AMLt scores for the Oracle, MaxMA, Best Mean, and MinMA for the two subsets of Dataset1360 separated by MMA = 1.5 bits, evaluated against the ground truth. The beat tracking performance is consistently high for songs with MMA >1.5 bits, with a mean MaxMA performance of ≈90%, which must be considered very accurate, and hence hints at a meaningful relationship between subjective judgement of beat tracking and the AMLt scores obtained from the objective evaluation. While beat tracking performance is lower for MMA < 1.5 bits this does not

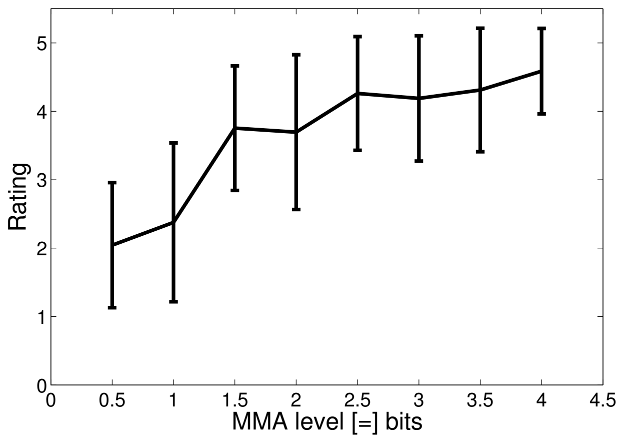


Figure 3: Listening test ratings *vs* MMA for the selected 48 music excerpts, from the MillionSongSubset.

Name	AMLt (%)	MMA
Oracle	95.4	MMA>1.5
MaxMA	89.9	
Best Mean	86.3	
MinMA	63.9	
Oracle	70.9	MMA<1.5
MaxMA	58.8	
Best Mean	54	
MinMA	50.1	

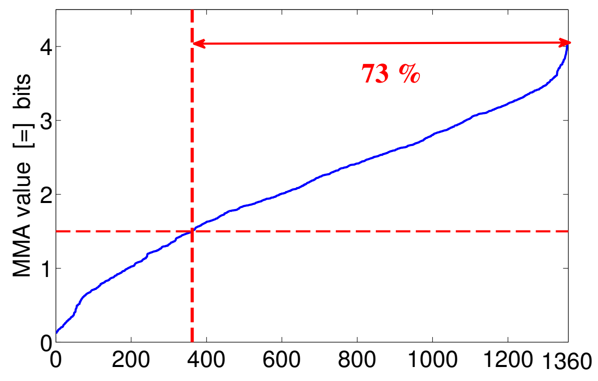
Table 1: Mean AMLt score of Oracle, MaxMA, Best_Mean, and MinMA for the two subsets of Dataset1360 divided by an MMA threshold of 1.5 bits.

mean the MaxMA beat estimations cannot be perceptually accurate, merely that we do not have high confidence in them.

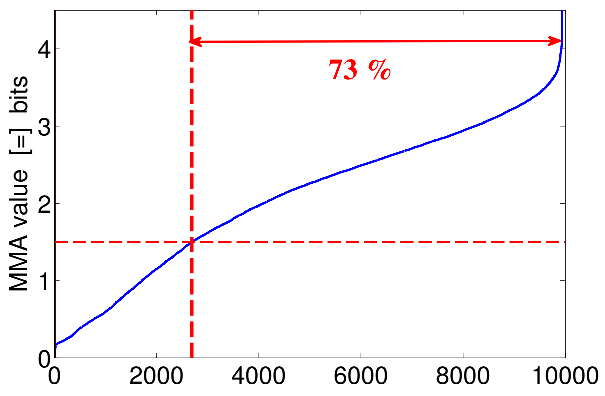
4.3.3 Last.fm Tag Analysis

Given the MMA threshold and collected Last.fm metadata, we now look at the genre-related tags of the songs that appear significantly more often (with $p < 0.0001$) in the MillionSongSubset with MMA above and below 1.5 bits. These are shown in Table 2. From inspection of the table we can see that the genres above the MMA threshold are those which we would typically associate with being "easier" for beat tracking where as those below the threshold appear more challenging. Seeing all genre labels related to metal music below the threshold was a surprising result since this music is strongly percussive and is not characterised by wide tempo changes. The fact that metal music consistently falls below the threshold indicates it might be the "noisy" element of the music which causes it to be difficult. To the best of our knowledge we are unaware of many metal examples in existing beat tracking databases. This suggests it is something of a forgotten genre for beat tracking.

Another important observation relates to the tag frequency for genre labels above and below the threshold. There is a far higher proportion of songs tagged "Rock" and "Pop" compared to all the others, and in general the



(a) Dataset 1360



(b) MillionSongSubset

Figure 4: Datasets sorted by MMA and the perceptual threshold of 1.5 bits.

tags used above the threshold appear much more frequently than those below it. From this we can infer that, just as Dataset1360 is biased towards easier cases for beat tracking [9], the same could be said of the MillionSongSubset. Evidence for this conclusion can be found in the description of the MillionSongDataset itself [1] where the lack of diversity is mentioned; in particular the small amount of classical and world music.

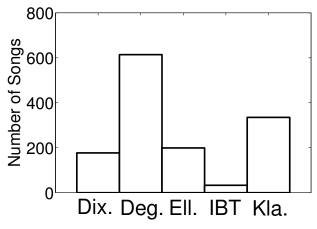
Given the disproportionate number of easier songs for beat tracking in this dataset, our estimate of 73% of songs for which beat tracking is acceptable may still be an optimistic estimate of the true level of beat tracking performance across all music.

4.3.4 MaxMA Choice of Beat Tracker

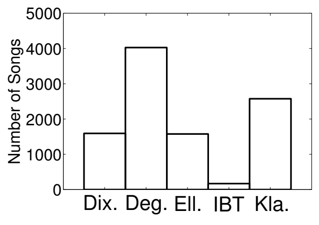
Having investigated the main results of applying MaxMA to automatically annotate beat locations, we now address the properties of the committee. Figure 5 presents histograms for both evaluated datasets depicting the proportion of songs where each beat tracking algorithm is selected as the MaxMA beat output. Both histograms show similar shapes, indicating that there may be some similar properties between the musical content of both datasets. The two most chosen algorithms are those of Degara [4] and Klapuri [10]; both of which perform most accurately against the ground truth, and can be considered the best among the state of the art methods. As to why the Degara algorithm is chosen more frequently than that of Klapuri,

Tag	Frequency	MMA
Rock	1080	MMA>1.5
Pop	680	
Dance	320	
Hip-hop	271	
Rap	193	
Pop rock	154	
Reggae	149	
Jazz	227	MMA<1.5
Instrumental	199	
Death metal	80	
Black metal	74	
Progressive metal	59	
Classical	36	
Grindcore	28	

Table 2: Frequency of the genre-based occurrence of tags for the two subsets of MillionSongSubset divided by an MMA threshold of 1.5 bits.



(a) Dataset 1360



(b) MillionSongSubset

Figure 5: Histograms with the number of times each algorithm is chosen with the MaxMA approach.

results in [4] indicate that the inter-quartile range of the Degara algorithm is smaller than that of Klapuri (for a similar median), implying it is "wrong" in a lower proportion of songs.

5. DISCUSSION AND CONCLUSIONS

To estimate the confidence of beat tracking without ground truth annotations we have proposed the use of two methods based on the mutual agreement between a committee of beat tracking algorithms. The first, the Mean Mutual Agreement, was used to estimate the level of consensus between the beat outputs of the committee. The second, the Maximum Mutual Agreement, was used for selecting the best beat tracking output from the committee of beat trackers.

Through a subjective listening test we determined an MMA threshold between this committee of beat trackers of 1.5 bits above which we believe automatic beat tracking can be applied with high confidence. Based on this perceptual confidence, we demonstrate that around 73% of the MillionSongSubset could be automatically annotated using our committee of beat trackers. This proportion of songs for which we can be confident in an automatic beat annotation was also verified in a second dataset with manually annotated ground truth. Given the apparent bias in these datasets towards easier genres for beat tracking, we consider this value of 73% to be somewhat optimistic. We plan to verify this hypothesis in future work by measuring MMA in more diverse datasets.

Regarding the types of music which formed the remaining 27% of the MillionSongSubset (*i.e.*, those below the threshold) we found a high proportion of tags related to metal and similar "noisy" styles of music. Beyond classical music and jazz, which are known to be challenging for beat tracking systems, we consider the difficulty of beat tracking in metal to be a new and unexpected result, and furthermore an interesting area for the future development of beat tracking algorithms.

In addition to using MMA to determine successful beat tracking, we also presented a related technique, MaxMA, to select beat estimations among a committee of beat trackers. The fact that a simple approach of this kind was able to demonstrate a significant improvement over using individual state of the art algorithms is encouraging. Yet, as our results indicate, performance of MaxMA falls some way below that of the Oracle system using our committee. This suggests that there is still room for making a more accurate selection among existing algorithms, and exploring new selection methods will form a further area for future work.

One limitation of our approach may have been the use of short song excerpts for the listening test. This was done to make the listening test as manageable as possible for a wide range of participants. However, to obtain a greater understanding of subjective ratings for longer musical excerpts and a better understanding of perceptual difficulty in beat perception we plan to conduct more sophisticated subjective listening experiments.

While all the directions for future work have so far been related to beat tracking, we strongly believe that, given suitable evaluation metrics, our framework based on MMA and MaxMA could be readily applied to other areas of MIR. We therefore encourage researchers to explore its usage in problems such as onset detection, chord detection, structural segmentation, and music transcription.

6. ACKNOWLEDGEMENTS

This research received support from the Portuguese Foundation for Science and Technology through the "ShakeIt" project (grants UTAustin/CD/ 0052/2008 and PTDC/ EAT-MMU/ 112255/ 2009) and through grants SFRH/BD/ 43704/ 2008 and SFRH/ BPD/ 51348/ 2011, and by Universidad Pontificia Bolivariana (Colombia) and Colciencias, and by the EU-funded project MIReS.

7. REFERENCES

[1] T. Bertin-Mahieux, D. P.W. Ellis, B. Whitman and P. Lamere, "The Million Song Dataset," in *Proc. of the 12th ISMIR conference*, pp. 591–596, 2011.

[2] M. E. P. Davies, N. Degara, and M. D. Plumbley, "Evaluation methods for musical audio beat tracking algorithms," Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06, 2009.

[3] M. E. P. Davies, N. Degara and M. D. Plumbley, "Measuring the performance of beat tracking algorithms using a beat error histogram," *IEEE Signal Processing Letters*, vol. 18, no. 3, pp. 157–160, 2011.

[4] N. Degara, E. Argones, A. Pena, S. Torres-Guijarro, M. E. P. Davies and M. D. Plumbley, "Reliability-Informed Beat Tracking of Musical Signals," *IEEE Transactions on Audio, Speech and Language Processing,* Vol. 20, pp. 290–301, 2012.

[5] S. Dixon, "Evaluation of the audio beat tracking system BeatRoot," *Journal of New Music Research*, Vol. 36, pp. 39–50, 2007.

[6] D. P. W. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[7] F. Gouyon, *A Computational Approach to Rhythm Description — Audio Features for the Computation of Rhythm Periodicity Functions and their use in Tempo Induction and Music Content Processing,* PhD. Thesis. MTG, Universitat Pompeu Fabra, 2005.

[8] A. Holzapfel and Y. Stylianou "Parataxis: Morphological similarity in traditional music," *Proc. of the 11th ISMIR Conference,* pp. 453–458, 2010.

[9] A. Holzapfel, M. E. P. Davies, J.R. Zapata, J.L. Oliveira and F. Gouyon, "Selective sampling for beat tracking evaluation," *IEEE Transactions on Audio, Speech and Language Processing,* In press, 2012.

[10] A. P. Klapuri, A. J. Eronen and J. T. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Trans. on Audio,Speech, and Language Processing,* Vol. 14, No. 1, pp. 342–355, 2006

[11] M. Levy and M. Sandler, "Structural segmentation of musical audio by constrained clustering," *IEEE Transactions on Audio, Speech and Language Processing,* vol. 16, no. 2, pp. 318–326, 2008.

[12] M. F. McKinney, D. Moelants, M. E. P. Davies, A. Klapuri, "Evaluation of Audio Beat Tracking and Music Tempo Extraction Algorithms," *Journal of New Music Research,* Vol. 36, pp. 1–16, 2007.

[13] J. Oliveira, F. Gouyon, L. Martin, and L. Reis: "IBT: A realtime tempo and beat tracking system.," in *Proc. of the 11th ISMIR conference*, pp. 291–296, 2010.

[14] H. S. Seung, M. Opper and H. Sompolinsky "Query by committee," in *Proc. of the 5th Annual Workshop on Computational learning theory,* pp. 287–294, 1992.

A CORPUS-BASED STUDY OF RHYTHM PATTERNS

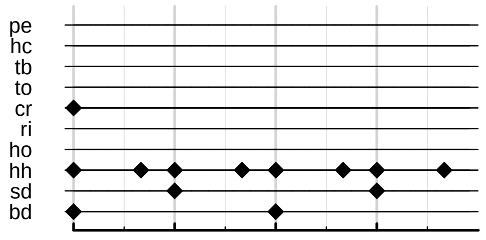
Matthias Mauch **Simon Dixon**
Centre for Digital Music, Queen Mary University of London
{matthias.mauch, simon.dixon}@eecs.qmul.ac.uk

ABSTRACT

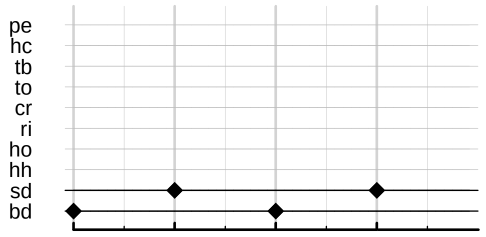
We present a corpus-based study of musical rhythm, based on a collection of 4.8 million bar-length drum patterns extracted from 48,176 pieces of symbolic music. Approaches to the analysis of rhythm in music information retrieval to date have focussed on low-level features for retrieval or on the detection of tempo, beats and drums in audio recordings. Musicological approaches are usually concerned with the description or implementation of man-made music theories. In this paper, we present a quantitative bottom-up approach to the study of rhythm that relies upon well-understood statistical methods from natural language processing. We adapt these methods to our corpus of music, based on the realisation that—unlike words—bar-length drum patterns can be systematically decomposed into sub-patterns both in time and by instrument. We show that, in some respects, our rhythm corpus behaves like natural language corpora, particularly in the sparsity of vocabulary. The same methods that detect word collocations allow us to quantify and rank idiomatic combinations of drum patterns. In other respects, our corpus has properties absent from language corpora, in particular, the high amount of repetition and strong mutual information rates between drum instruments. Our findings may be of direct interest to musicians and musicologists, and can inform the design of ground truth corpora and computational models of musical rhythm.

1. INTRODUCTION

In Western popular music and jazz, the main percussive instrument is the drum kit, consisting of a collection of drums and cymbals arranged around the drummer. Drum kits can contain a large range of different instruments. The *bass drum* (or *kick drum*) is usually the drum with the lowest frequency and is operated via a foot pedal. The *snare drum*, the dominant back-beat instrument, has a higher-pitched sound with additional noise components from the *snares* spanned across its lower skin. The *hi-hat* is made from two cymbals facing each other, which the drummer can open and close via a foot-pedal. The closed hi-hat has a short, high-pitched sound, whereas the open hi-hat has a longer sustain. *Ride cymbals* have a sustained high-pitched ring, while the *crash cymbals'* sound is usually more noise-like. *Tom-toms* are drums of intermediate sizes between the bass and snare drums. *Hand-claps* and *tambourine* are often used to provide additional colour, along with further varied percussion instruments, which we do not discuss in this study. The history and makeup of the modern drum kit is covered comprehensively elsewhere, e.g. [3].



(a) a pattern in $\frac{4}{4}$ meter



(b) pattern projected to the bass drum/snare drum space

Figure 1: Example drum patterns in drum tab notation, with musical time on the x axis; strong lines are beats. Drums from bottom to top: bd – bass drums, sd – snare drums, hh – hi-hat closed/pedal, ho – hi-hat open, ri – ride cymbals, cr – crash cymbals, to – toms, tb – tambourine, hc – hand clap, pe – other percussion.

In music information retrieval (MIR), most research on rhythmic features has concentrated on beats, meter and tempo. The tracking of beats establishes a temporal grid on a piece of music, which is useful to anchor other descriptors of a musical piece in time. The timing of beats also determines the tempo, which correlates with the perceived speed of the music [10]. The automatic identification of meter [9] is also based on beats, and provides additional rhythmic information. However, neither meter nor tempo, nor their combination, capture the temporal sequence of rhythmic events. Several audio features have been developed to include this temporal information [5, 17, 19] with considerable success, especially for clear-cut cases such as the classification of ballroom dances [4]. Being concerned with good performance in retrieval tasks, these methods are deliberately agnostic to how the rhythmic signal was originally created. On the other end of the spectrum lies the automatic transcription of music, and that of drums in particular. Drum transcription algorithms [7, 22, 23] usually neglect musical higher-level context such as meter and simultaneous rhythm patterns. One possible exception is Paulus's n-gram model of drum sequences [18] that informs transcription. The n-gram model demonstrates that context models can be useful to suppress errors, but it is also quite obvious that modelling rhythm as sequences of half-beat length symbols is a strong simplification that cannot capture interactions of concurrent rhythms played on multiple drums. A further simplification present in most drum transcription papers is the very small set of different drums considered: bass drum, snare drum and hi-hat. In this paper, we consider a much larger rhythm space, both in terms of temporal context and drum instrumentation.

Comparatively little work in MIR has quantitatively examined rhythms in symbolic data. While Muramaki's work on drum fill detection [15] is concerned with analysis, most work is focussed on improving music production, for example by combination drum loops of suitable complexity [21]. The study of rhythm has a long tradition in musicology, but only in recent decades has empirical music analysis found its way into the musicological tradition. Notable tools include the Humdrum Toolkit [8], jSymbolic [14] and music21 [2], which facilitate the processing of symbolic music, but do not directly examine the statistical properties of the corpus itself, nor provide tools as sophisticated as those available for natural language processing. In the domain of harmony, some attempts have been made to analyse chord progressions with language models [13, 20].

In this paper, language models are employed to analyse the statistical properties of a large corpus of drum parts, to reveal the degree of variety within and between pieces, and to discover interdependencies between different parts of the drum kit. In the next section we describe our representation of rhythm patterns, while in section 3 an overview of the data set, consisting of 48,176 MIDI files, is given. Section 4 provides the results of our analyses, and the final two sections contain a brief discussion and conclusions.

2. DRUM PATTERN DEFINITION

In order to build a corpus of drum patterns, we need to segment the music into short chunks whose lengths corresponds to meaningful metrical units. Since we are dealing with a symbolic representation which provides unambiguous onset times, the main effort required is to parse the events according to the metrical structure, suppressing performance-related information such as fluctuations in tempo, timing, and dynamics, which—for the purposes of this study—we are not interested in. Instead, similar to linguists building text corpora from *stemmed* words with grammatical endings removed, we build reduced drum pattern models by applying five levels of abstraction.

Bar segmentation. The tracks are segmented into bars as encoded in the MIDI files. Each bar is a *token*, the fundamental unit, similar to word tokens in language.

Drum categorisation. We summarise the General MIDI standard drums into 10 known drum categories (see Figure 1) and one *unknown* category.

	count	portion
4/4	4,305,516	90.3%
3/4	188,297	3.9%
2/4	114,068	2.3%
6/8	53,681	1.1%
12/8	19,575	0.4%
other	84,830	1.7%

Table 1: Distribution of time signatures in the corpus.

Tempo abstraction. We discard tempo information (but not metrical structure).

Intensity abstraction. We discard sound intensity information, i.e. MIDI velocity.

Quantisation. We quantise the drum notes relative to the beat, reducing the granularity to a grid of 12 equally spaced divisions per beat span, and retain only their onset time.

The resulting representation contains approximately the same information that would be found in traditional score notation. After this "stemming" procedure, we characterise a drum pattern via the presence (or absence) of drum onsets for each beat, position within the beat, and drum category, as visualised in the example *drum tab* representation shown in Figure 1a. Hence, a bar with N_b beats can be represented as a binary sequence of $N_b \times 12 \times 11$ bits. For the most frequent time signature, $\frac{4}{4}$, the number of beats is $N_b = 4$, and so the space of possible $\frac{4}{4}$ patterns allows $2^{4\times12\times11} \approx 10^{159}$ different patterns. Thus, despite five abstraction steps, we have retained an extremely large pattern space. Since the space is much larger than any data set, it is clear that large parts of the space will never appear in actual music. We show later that we can not only quantify the size of the space used in a given corpus, but also make predictions about how much of the space will be used as the corpus grows.

We define drum pattern sub-spaces by discarding some drums or metric positions. For example, if we restrict our attention to sub-patterns made of only bass and snare drums, a large number of different full patterns with, say, different use of the hi-hat would be mapped to the sub-pattern shown in Figure 1b.

3. DATA

We collected 72,283 unique MIDI files from the Internet. In order to understand the nature of the resulting collection, we drew a random sample of 100 songs and manually classified them. The sample mainly contains pop/rock music (62 songs), film music (10), jazz (9), classical (7) and country/folk music (6). Of the six remaining songs, five are of various genres and one was not decodable. A large proportion of the songs are good-quality renditions of popular recordings.

A study of the within-track interonset intervals (IOIs) on the whole dataset reveals that many songs are already quantised; about a third of the songs (34%) contain $> 99\%$

(a) All metrical positions

drums	# types	predicted # types at 20M tokens	$\mathcal{P}$ in %	R_{sw} in %	R_{loc} in %
all	**656798**	**1688906**	**6.23**	**73.5**	**33.4**
bd	46243	101230	0.45	91.1	64.2
sd	62647	143525	0.62	90.5	67.5
bd/sd	186688	454218	1.90	85.5	52.6
hh/ho	76351	174590	0.79	91.3	69.4
cmb	170344	415935	1.76	85.8	54.3
to	29394	70500	0.30	95.3	85.9
hc/tb/pe	84417	191712	0.88	94.4	81.9
bd/sd/cmb	466962	1176552	4.76	77.6	38.5

(b) Beats 1 and 2 only

drums	# types	predicted # types at 20M tokens	$\mathcal{P}$ in %	R_{sw} in %	R_{loc} in %
all	342453	786850	3.30	82.1	48.0
bd	7602	14788	0.07	94.9	76.8
sd	14272	30612	0.14	94.7	79.5
bd/sd	48493	106701	0.47	91.3	68.5
hh/ho	21460	44131	0.20	94.5	79.1
cmb	57287	124465	0.54	90.8	64.8
to	7523	16782	0.07	97.2	92.1
hc/tb/pe	32014	67845	0.31	96.1	86.9
bd/sd/cmb	198699	454309	1.94	85.1	52.9

Table 2: Sub-pattern statistics. $\mathcal{P}$ is productivity (see Section 4.2), R are repetition indices (Section 4.3).

IOI-quantised events, while 60% still contain $> 75\%$ IOI-quantised events. Our impression that the songs are usually carefully crafted for authentic playback is reflected in the fact that 71% of the songs have varied velocities (less than half of the notes uses the most popular velocity), i.e. it is likely that only few songs are MIDI exports from music typesetting programs.

In order to limit the influence of abnormally long songs only notes less than 20 minutes into any song are considered. Very soft drum notes (velocity < 20) are removed. We exclude songs with empty drum tracks, and those whose musical beat is likely to be out of sync with the MIDI beat (i.e. where the frequency of on-beat drum notes is $< 50\%$ that of the most frequent quantisation).

After decoding, the collection contains 4,765,947 bar tokens in 48,176 files, which corresponds to a mean of around 99 bars per song. The overwhelming majority, 90% of bars, is in $\frac{4}{4}$ time, with only a few other time signatures exceeding 1% of the corpus (see Table 1).

The terms *type* and *token* are borrowed from natural language processing and will be used here as follows:

type: unique drum pattern ($\approx$ unique word in language),

token: drum pattern type instance.

The overall number of bar types in our database is 656,798. The sub-pattern spaces retain the same number of tokens, but can have dramatically fewer distinct pattern types, as Table 2 (column 2) shows.

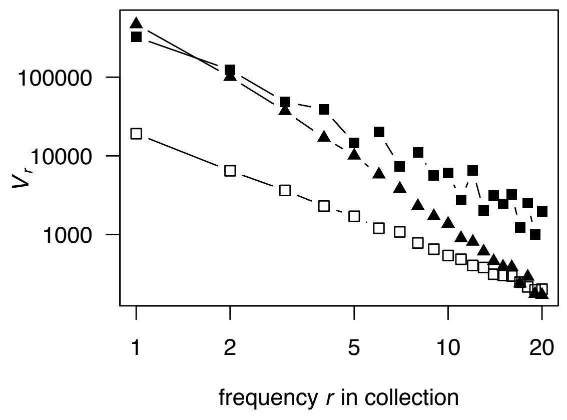


Figure 2: Type frequencies V_r by token count r for drum patterns (filled squares), drum pattern by song (filled triangles), the Brown language corpus (blank squares).

4. RESULTS

This section provides an overview and discussion of some insights that can be gained from our corpus of drum patterns. More exhaustive information is available at *http://isophonics.net/ndrum.*

4.1 Large Number of Rare Events

As with many natural language corpora, the distribution of type counts and the frequencies of these type counts are extremely skewed. Figure 2 shows a plot called *frequency-of-frequencies* plot, in which the number of type occurrences r in the database is plotted against the number of types V_r that occur r times. The figure shows three graphs: drum pattern counts, song-wise drum pattern counts (one per song in which the type occurs), and word type counts from a corpus of American English, the *Brown Corpus*. The number V_1 of types that occur only once in the whole collection is greatest; tokens occurring twice already account for much smaller fractions of the corpora, a phenomenon often referred to as *large number of rare events* (LNRE).

The log-log scale plot in Figure 2 illustrates an additional property of the data: all distributions can be approximated by a straight line, a characteristic of "scale-free" distributions. For a discussion of this phenomenon, see, for example, [16]. While the full drum pattern count (filled squares) resembles the word distribution in the Brown corpus in slope (the absolute height reflects that the Brown corpus has only 1M tokens), it is not as smooth as the Brown corpus's. However, the unstable nature of the graph is not random; rather, the higher values at multiples of 2 reflect the usual organisation of music in units of even multiples of bars. As we would expect, then, counting the number of *songs* in which a type appears leads to a much smoother graph (filled triangles) that is unaffected by repetitions. We will return to song-wise counts in Section 4.3.

4.2 Vocabulary Growth

For LNRE distributions we can estimate how fast the number of types (in our case: distinct drum patterns) is growing with vocabulary size. A popular measure for that is productivity $\mathcal{P}$ [1]. For a corpus of size N with V_1 types that occur only once, productivity is calculated as

$$\mathcal{P} = V_1/N. \tag{1}$$

This measure is an indicator of the potential to generate new patterns. The productivity of large pattern spaces is generally much higher than that of smaller sub-spaces. For example, all productivity values in Table 2b, where the sub-patterns are constrained to the first two beats of a bar, are far smaller than the respective ones in table 2a. More interestingly, however, there are also large differences to be found between single drums. For example, the productivity of the snare drum as shown in table 2a is far greater than that of the bass drum in the same table, suggesting that snare drum patterns are used more creatively (most probably due to the bass drum usually being operated by one foot). In fact, assuming a Zipf-Mandelbrot model [6], we can predict the vocabulary size as a function of corpus size; Table 2 displays productivity values and the predicted number of tokens for a vocabulary size of 20 million.

4.3 Repetition and Different Ranking Types

Simply using the relative frequencies p_{rf} of pattern types is the standard way to measure word probabilities, but it is less informative in music because of the high amount of repetition present. In our paper on chord progressions [12] we suggest to use the proportion of songs a (chord) pattern occurs in, which we call p_{sw} here. For example, counting a token only once per song reduces the overall token count from $N = 4,765,967$ to $N_{\text{sw}} = 1,264,139$ for the full pattern spaces. A softer way of reducing the influence of repetition is motivated by the observation that drum patterns in consecutive bars are often identical: one can eliminate tokens that are exact repetitions of the immediately preceding token. This locally non-repeating set has $N_{\text{loc}} = 3,176,153$ tokens, with relative frequencies denoted by p_{loc}. We use the reduced token counts to define local and song-wise repetition indices:

$$R_{\text{loc}} = 1 - \frac{N_{\text{loc}}}{N} \text{ and } R_{\text{sw}} = 1 - \frac{N_{\text{sw}}}{N}. \tag{2}$$

Table 2 lists the repetition indices (in %) for different sub-pattern corpora. Even the full patterns have a song-wise repetition index R_{sw} of 74%, meaning that only just more than a quarter of the drum patterns per song are unique. A similar picture emerges when looking at local repetition, which accounts for $R_{\text{loc}} \approx 33\%$ of all tokens. Repetition is even more dominant in smaller sub-patterns: $R_{\text{sw}} = 90\%$ of snare drum pattern tokens are repeated within a song, and $R_{\text{sw}} = 68\%$ are repetitions of the preceding bar.

In Figure 3, we show the 10 most common bar-length patterns in the corpus. Empty patterns with different time signatures occupy the 1st, 4th and 5th rank, while standard rock patterns using only bass, snare and closed high-hat occupy the remaining ranks. A variation with a swing high-hat pattern appears at rank 9.

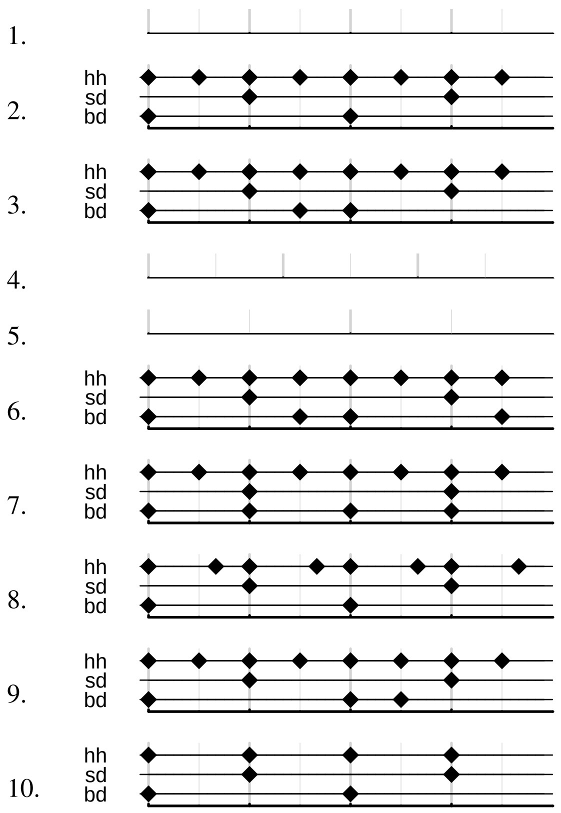


Figure 3: Ranking by p_{rf} score (Section 4.3).

The song-wise results are shown in Figure 4, with the empty patterns removed. Although some of the same patterns appear, the non-empty pattern which occurs in the most songs is a crash cymbal and bass drum on the first beat of the bar, presumably at the end of a piece or section. Rank 3 and 7 have quarter note patterns often used for "counting in" a song, while ranks 4 and 8 are the two sub-patterns of the rank 2 pattern—a single crash cymbal and a single bass drum respectively. Ranks 5 and 6 contain standard rock drum beats seen previously. The results with local repetition removed (i.e. ranked by p_{loc}) are similar and are not shown here. Comprehensive rankings can be found at *http://isophonics.net/ndrum.*

4.4 Collocations and Typical Drum Patterns

Linguists have long realised that interesting, idiomatic word combinations do not usually appear in the top ranks when sorted by frequency. *Collocations*—combinations of two words that occur more often than would be expected from their individual frequencies—are usually more interesting and meaningful. One strategy to discover collocations is to consider two hypothetical models: H_1, by which the likelihood of one of the tokens to occur depends on the other, and H_2, by which their occurrences are independent. One can then calculate the likelihood ratio

$$\log \lambda = \log \frac{L(H_1)}{L(H_2)} \tag{3}$$

of the two hypotheses for any pair of word types—or indeed drum pattern types. We follow Manning and

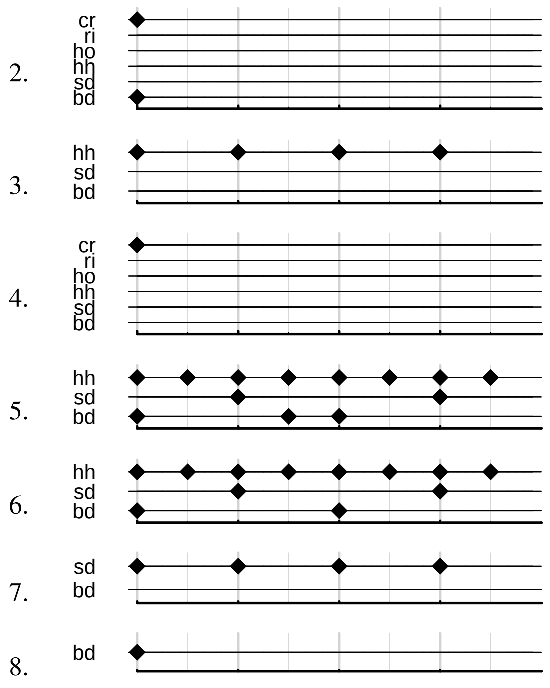


Figure 4: Ranking by p_{sw} score (Section 4.3).

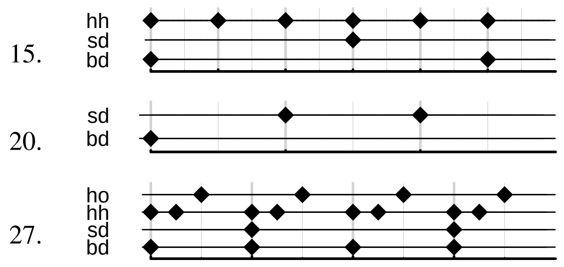


Figure 5: Ranking by collocation score (Section 4.4).

Schütze's approach [11, Chapter 5], assuming binomial type count distributions, and calculate $\log \lambda$ scores for combinations of bass drum/snare drum patterns on the one hand and hi-hat (open and closed) patterns on the other.

Ranking by the collocation score (3) results in a list of *typical* drum patterns that need not necessarily be frequent. Figure 5 shows some example of rarer patterns that nevertheless rank much higher than in the frequency rankings discussed in Section 4.3. For example, the typical $\frac{6}{8}$ pattern at rank 15 appears only at rank 99 in the raw frequency ranking (and at ranks 59 and 389 when ranked by r_{loc} and r_{sw}, respectively). The $\frac{3}{4}$ pattern at rank 20, too, is much further down the frequency rankings (48, 115, 264), as is the disco-style pattern at collocation rank 27 (35, 67, 171).

4.5 Mutual Information

That the decomposition of drum patterns is meaningful can be illustrated by the fact that the information flow between the sub-patterns across the corpus models musical relationships between them. The entropy (in bits) of a discrete

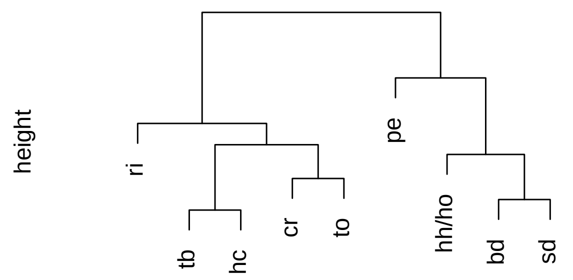


Figure 6: Hierarchical clustering of nine drum types by mutual information. The distance matrix is based on the inverted, normalised mutual information values (see text).

distribution X in with probabilities p_i is defined as

$$H(X) = -\sum_{i=1}^{N} p_i \log_2 p_i \tag{4}$$

with the convention that if $p_i = 0$, then $p_i \log_2 p_i = 0$. It expresses how much information is needed in order to represent the distribution. While this is interesting in itself, we are interested in how much information two drum pattern sub-spaces X and Y share. This is what mutual information expresses. It is defined as

$$I(X;Y) = H(X) + H(Y) - H(X,Y). \tag{5}$$

To normalise the measure we divide by the sum of the individual entropies, and to turn the similarity into a measure of divergence we take the exponential of its negation:

$$d(X,Y) = \exp\left\{-\frac{I(X;Y)}{H(X)+H(Y)}\right\}. \tag{6}$$

This allows us to calculate pair-wise divergence values between all drum types. The result is visualised in Figure 6 as a binary tree obtained by hierarchical clustering with the complete-linkage algorithm. The more information is shared between drums according to d, the closer they will appear on the tree. The algorithm has indeed recovered aspects of the usage of the drum kit, with the drums that form the core of most rhythms in popular music—bass drum, snare drum and hi-hat—grouped together on the right, loosely associated with the percussion instruments. Within the remaining drums on the left hand side, the ride cymbals have their own branch, whereas the tambourine is grouped with hand claps, and crash cymbals with tomtoms, with each grouping suggesting high mutual information.

5. DISCUSSION AND FUTURE WORK

We must be aware that findings made in the MIDI domain may only partially be applicable to other music. Furthermore, the size of the database prohibits the manual verification of every song. In addition to the measures described in Section 3, automatic sanity checks could further reduce the noise in the data.

We expect that the outcomes of the present study will be valuable to musicians and researchers, so we are interested in a rigorous evaluation of its usefulness. Several scenarios are conceivable. For example, our system can easily be extended to return a ranked list of pattern synonyms, i.e. patterns that are used in similar contexts as the query pattern—a creative tool for drummers. A useful music informatics application could be to extend the promising n-gram technique for audio drum transcription proposed by Paulus, especially with models that "back off" [11, Chapter 6] not only in time, but also in the sub-pattern instrument spaces presented in this paper.

6. CONCLUSIONS

We have introduced a novel method of empirical research on musical rhythm by considering bar-length drum patterns and treating them analogously to words in natural language processing. This paper has shown that the approach yields useful and interesting results because the palette of tools available from natural language processing can—to a large extent—be used in the musical domain, too.

We have found that the distributions of drum patterns resemble those of and English words, and have used this fact to predict different vocabulary growth behaviours in our musical corpus. Vocabulary growth predictions can be useful to inform decisions on how much ground truth is needed to cover a given proportion of unseen data.

We have discovered some properties that clearly distinguish our data from language corpora, most prominently the extremely high degree of repetition. A second, more subtle, difference is that drum patterns can be decomposed in time and by instrument, yielding distributions with different characteristics.

We have proposed three simple ways of ranking drum patterns by raw frequency, repetition-reduced frequency, and song frequency. In order to identify not only frequent, but *interesting* drum pattern combinations, we have applied collocation ranking to our drum corpus. For musicians, the pattern rankings, which can be found at *http://isophonics.net/ndrum*, may be the most interesting aspect of this paper.

Finally, by calculating the mutual information flow between sub-patterns pertaining to the individual drum categories, drum categories that are musically related cluster together.

We believe that the corpus-based study of rhythm as proposed in this paper is interesting not only to musicians. Musicologists and music informatics researchers might find it a valuable resource to obtain a quantitative view on rhythm and drum patterns.

7. REFERENCES

[1] R. H. Baayen. Quantitative aspects of morphological productivity. In Geert Booij and Jaap Van Marle, editors, *Yearbook of Morphology 1991*, pages 109–149. Kluwer Academic Publishers, 1992.

[2] M. S. Cuthbert. music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data. *Proc. of the 11th Int. Conf. on Music Information Retrieval (ISMIR 2010)*, pages 637–642, 2010.

[3] M. Dean. *The Drum: A History*. Scarecrow Press, 2011.

[4] S. Dixon, F. Gouyon, and G. Widmer. Towards characterisation of music via rhythmic patterns. *Proc. of the 5th Int. Conf. on Music Information Retrieval (ISMIR 2004)*, pages 509–516, 2004.

[5] D. P. W. Ellis and J. Arroyo. Eigenrhythms: Drum pattern basis sets for classification and generation. *Proc. of the 5th Int. Conf. on Music Information Retrieval (ISMIR 2004)*, pages 554–559, 2004.

[6] S. Evert and M. Baroni. Testing the extrapolation quality of word frequency models. *Proc. of Corpus Linguistics*, 2005.

[7] O. Gillet and G. Richard. Transcription and separation of drum signals from polyphonic music. *IEEE Trans. on Audio Speech and Language Processing*, 16(3):529–540, 2008.

[8] D. Huron. Music Information Processing Using the Humdrum Toolkit: Concepts, Examples, and Lessons. *Computer Music Journal*, 26(2):11–26, 2002.

[9] A. P. Klapuri, A. J. Eronen, and J. T. Astola. Analysis of the Meter of Acoustic Musical Signals. *IEEE Trans. on Audio, Speech, and Language Processing*, 14(1):342–355, 2006.

[10] M. Levy. Improving Perceptual Tempo Estimation with Crowd-Sourced Annotations. *Proc. of the 12th Int. Conf. on Music Information Retrieval (ISMIR 2011)*, pages 317–322, 2011.

[11] C. D. Manning and H Schütze. *Foundations of Natural Language Processing*. MIT Press, 1999.

[12] M. Mauch, S. Dixon, C. Harte, M. Casey, and B. Fields. Discovering Chord Idioms through Beatles and Real Book Songs. *Proc. of the 8th Int. Conf. on Music Information Retrieval (ISMIR 2007)*, 2007.

[13] M. Mauch, D. Müllensiefen, S. Dixon, and G. Wiggins. Can Statistical Language Models be Used for the Analysis of Harmonic Progressions? *Proc. of the 10th Int. Conf. on Music Perception and Cognition (ICMPC 2008)*, 2008.

[14] C. McKay. *Automatic Genre Classification of MIDI Recordings*. PhD thesis, McGill University, 2004.

[15] Y. Murakami and M. Miura. Automatic detection system for fill-in from drum patterns employed in popular music. *Proc. of the 10th Western Pacific Acoustics Conference*, 2009.

[16] M. E. J. Newman. Power laws, Pareto distributions and Zipf's law. *Contemporary physics*, 46(5):323–351, 2005.

[17] E. Pampalk. Audio-Based Music Similarity and Retrieval : Combining a Spectral Similarity Model with Information Extracted from Fluctuation Patterns. *Proc. of the 7th Int. Conf. on Music Information Retrieval (ISMIR 2006)*, 2006.

[18] J. K. Paulus and A. P. Klapuri. Conventional and Periodic N-grams in the Transcription of Drum Sequences. *Int. Conf. on Multimedia and Expo (ICME 2003)*, 2003.

[19] G. Peeters. Spectral and Temporal Periodicity Representations of Rhythm for the Automatic Classification of Music Audio Signal. *IEEE Trans. on Audio Speech And Language Processing*, 19(5):1242–1252, 2011.

[20] R. Scholz, V. Dantas, and G. Ramalho. Automating functional harmonic analysis: the Funchal system. *Seventh IEEE International Symposium on Multimedia*, 2005.

[21] G. Sioros and C. Guedes. Complexity-Driven Recombination of MIDI Loops. *Proc. of the 12th Int. Conf. on Music Information Retrieval (ISMIR 2011)*, pages 381–386, 2011.

[22] K. Yoshii, M. Goto, and H. G. Okuno. Automatic Drum Sound Description for Real-World Music Using Template Adaptation and Matching Methods. *Proc. of the 5th Int. Conf. on Music Information Retrieval*, pages 184–191, 2004.

[23] A. Zils, F. Pachet, O. Delerue, and F. Gouyon. Automatic extraction of drum tracks from polyphonic music signals. *Second Int. Conf. on Web Delivering of Music*, pages 179–183. IEEE Comput. Soc, 2002.

ONE IN THE JUNGLE: DOWNBEAT DETECTION IN HARDCORE, JUNGLE, AND DRUM AND BASS

Jason A. Hockman^{1,2}, Matthew E.P. Davies^{3}, and Ichiro Fujinaga^{1,2}
^{1}Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
^{2}Distributed Digital Archives and Libraries (DDMAL), McGill University, Montreal, Canada
^{3}Sound and Music Computing Group, INESC TEC, Porto, Portugal
jason.hockman@mail.mcgill.ca, mdavies@inescporto.pt, ich@music.mcgill.ca

ABSTRACT

Hardcore, jungle, and drum and bass (HJDB) are fast-paced electronic dance music genres that often employ resequenced *breakbeats* or drum samples from jazz and funk percussionist solos. We present a style-specific method for downbeat detection specifically designed for HJDB. The presented method combines three forms of metrical information in the prediction of downbeats: low-level onset event information; periodicity information from beat tracking; and high-level information from a regression model trained with classic breakbeats. In an evaluation using 206 HJDB pieces, we demonstrate superior accuracy of our style specific method over four general downbeat detection algorithms. We present this result to motivate the need for style-specific knowledge and techniques for improved downbeat detection.

1. INTRODUCTION

In the early 1990s, affordable sampling technologies (e.g., Akai S900 and Commodore Amiga) and the popularity of rave culture provided the impetus for the creation of three related genres—hardcore, jungle, and drum and bass (HJDB)—unique in their fast tempi and drum sounds, which are mostly derived from samples of percussion solos in 1960s–80s funk and jazz recordings known as *breakbeats*. Since 1990, over 25,000 artists have contributed over 132,000 tracks on almost 6,000 labels. [1] HJDB became so popular in the mid-1990s that it was showcased on BBC's Radio 1 program, "One In The Jungle". Both popular press [1,16] and academic literature [10] have mostly treated HJDB from a sociology/cultural studies perspective, presenting the music within larger contextual issues, e.g., race, drugs, and cultural politics. A notable exception [3], provides tools for automated breakbeat splicing and resequencing.

[1] http://www.rolldabeats.com/stats

In this study, we present a downbeat detection model created with the intention of finding downbeats within music containing breakbeats, and provide a comparison of its performance against four pre-existing algorithms on a database of 206 HJDB excerpts. We view this as a first step in an automated analysis of the musical surface of HJDB from a computational musicology perspective, towards the eventual goal of understanding how individual artists use breakbeats (e.g., slice ordering and pitch adjustment) in modern music.

1.1 Hardcore, Jungle, and Drum and Bass

Hardcore began around 1990, and was the first of the HJDB genres to fully embrace the use of breakbeats. Tracks soon left the 120–130 beats per minute (BPM) house and techno standard and steadily became faster (upwards of 180 BPM), with longer, more intricate drum patterns. The less synth-driven, breakbeat collage art of jungle appeared around 1992. By 1994, many artists abandoned the rhythmic complexity of jungle in favor of simpler rhythms associated with drum and bass. As is the standard workflow in these genres, breakbeats are recorded into a sampler's memory, segmented, and assigned to MIDI note values. HJDB artists create the rhythmic (and sometimes harmonic and melodic) structure of their arrangements using these samples. While hundreds of breakbeats have been employed in HJDB, many artists use a handful of standards such as the "Amen" breakbeat, originally from The Winston's *Amen, Brother* [17].

1.2 Downbeat Detection

The meter of a piece of music implies a counting mechanism for hierarchical stressed and unstressed beats within a measure. A *downbeat* is the first beat within a measure (or if counting beats, the *one*). While the computational task of downbeat detection has received little attention, the related task of beat tracking has received much more attention in recent years [9,13,15]. A possible reason for this imbalance may be related to the increased complexity of the task; prior to extracting downbeats, the estimation of additional subtasks (e.g., onset detection and beat detection) is often required, which can propagate errors into downbeat estimation. Robust downbeat detection would benefit information retrieval tasks such as structural analysis [8], and would facilitate analysis of phrase structure and hypermeter; both useful in improving automated mixing and DSP effects that rely on musically relevant change-point positions. More relevant to our interests, downbeat detection provides key segmentation points that allow for a comparison of HJDB artists' drum usage.

Generalized downbeat detection methods have been proposed in the literature. Goto [11] employs rhythmic template patterns to the output of a drum detection algorithm. In non-percussive music, downbeats are assumed to be present at temporal locations of large spectral change, and are detected through a process of peak-picking spectral frames, grouping of the resultant segments into beats, and a comparison of beats for harmonic change. Davies and Plumbley [5] present a similar approach, in which downbeats are found by selection of beat positions that maximize spectral change. Klapuri et al. [13] extract the temporal evolution of a hidden metrical sequence exhibited in the output of a comb filter bank. The joint-state estimates of the beat, sub-beat, and meter periods are chosen through a first-order Markov process. Papadopoulos and Peeters [14] propose a method for joint estimation of harmonic structure and downbeats using an HMM that models chords and their metrical position. They present an additional method in [15] that also formulates the problem within an HMM framework, in which beat templates are first estimated from the data, and beats are then associated with positions in a measure by reverse Viterbi decoding.

Unlike the aforementioned algorithms, which are generalized for arbitrary musical input, Jehan [12] presents a regression model that predicts downbeat positions based on learning style-specific characteristics from training data containing rhythmic and timbral characteristics akin to those in the testing data. Evaluation is presented in constrained circumstances, in which testing is performed on part of the same song used for training, or on a test song from the same album on which the remaining songs are used as training.

It is our belief that while generalized downbeat detection models will perform well in many circumstances, there remain niche genres that fall outside the scope of these methods [12]. HJDB, while heavily percussive and almost exclusively in 4/4, presents challenges due to its characteristic fast tempo, high note density, non-standard use of harmony and melody, and emphasis on offbeats.

1.3 Motivation

With the exception of [12,15], the above methods rely on general approaches to downbeat detection, and do not infer information about content between estimated downbeats. Our eventual aim is to use detected downbeats towards an estimation of the ordering of drum segments, and their source, i.e., the breakbeat from which the drums were sampled. To do so, our particular application requires an understanding of likely solo percussion performances. We therefore attempt to leverage knowledge of breakbeat timbres and patterns from the 1960s–80s to inform an understanding of three modern genres that utilize them. At the core of the presented model is a top-down support vector regression technique, similar to [12] trained on these building blocks of the music under analysis. Although HJDB artists often resequence segments of breakbeats, the resequenced patterns often reflect knowledge of standard breakbeat patterns. To improve the robustness of this model we incorporate additional stages including beat tracking, and low-level onset detection to focus on kick drum frequencies.

The remainder of this paper is structured as follows: Section 2 outlines our HJDB-specific downbeat detection method. Section 3 presents our evaluation methodology and dataset. Section 4 presents evaluation results and discussion, and Section 5 provides conclusions and future work.

2. METHOD

Our main interest is to determine if an algorithm trained on breakbeat patterns and timbres can find downbeats in modern forms of music that employ them. We began by re-implementing the algorithm as described in [12], with the aim of utilizing it within the full range of HJDB music. Exact parameterization of the model is not provided in [12], so we first tuned our model by optimizing results on examples described in the paper.

2.1 Support Vector Regression for Downbeats

In [12], support vector regression (SVR) is employed to infer likely downbeat positions. Audio is segmented by onset detection or a tatum grid. Each audio segment, S, is associated with a metrical position, t, within a measure with downbeats at t=0, and last sample points before the next downbeat at t=3. We used the LibSVM [2] epsilon-SVR algorithm in MATLAB with a RBF kernel.

To train the regression model, we require a feature matrix F and associated class vector C, which we derive from breakbeats. Two HJDB artists selected 29 breakbeats from several lists of breakbeats commonly used in HJDB. Audio for each breakbeat was trimmed to the portion of the signal containing only the percussion solos. Each breakbeat, β, is then segmented using an eighth-note grid, and a class vector, c_β, is created using the metrical position of each eighth-note segment in a measure.

The feature matrix f_β is comprised of 58 features extracted from each segment in β consisting of: mean segment Mel-frequency spectral coefficients; loudness of the onset (dB) of each segment; maximum loudness (dB) of the onset envelope; and chroma. Segments are then associated with metrical positions in c_β as in [12]. f_β is normalized to have zero-mean and unit variance across each row (all segments). Features are shingled (time-lagged and weighted linearly) [2] to emphasize more recent segments. We then aggregate feature matrices and

[2] http://www.csie.ntu.edu/∼cjlin/libsvm/

class vectors across all breakbeats, creating an aggregate feature matrix F and aggregate class vector C. A feature and parameter optimization stage found best results using 40 Mel-frequency spectral coefficients and as in [12], 8 to 16 past segments (equivalent to 1 to 2 bars). Principal Component Analysis (PCA) feature reduction is applied to F to extract the top ten features across all breakbeats. A model is then trained using F and C.

To test the regression model using test audio, A, we require a feature matrix F_A. We first segment the audio using an eighth-note grid created by interpolating the temporal location of beats (we assume beats are found at the quarter-note level), γ, as found by Beatroot [7]. F_A is created similarly to f_β. The PCA model prepared in the training set is applied for feature reduction. We then use the trained model created above with feature matrix F_A to predict class values, C_A, which contain the estimated metrical position of each segment. In [12], the derivative of C_A is used as a detection function from which downbeats are chosen.

While we were able to recreate the examples in [12] using the reimplemented method, training on breakbeats and testing on HJDB music showed that C_A often differed significantly from the idealized output (i.e., pure sawtooth waveform), which resulted in the derivative of C_A being an unreliable detection function on its own.

2.2 Limitations of the Model

We now discuss three conditions that might cause these irregularities in C_A. First, breakbeat patterns are not universal; i.e., one breakbeat may employ a kick drum on beat one and snare drum on beat two, yet another may contain a kick drum on beats one and two, and a snare on the offbeat of two. As a result, C_A may not monotonically increase between downbeats. Second, HJDB artists often re-order slices, which will also cause undesirable output between downbeats. However, breakbeats almost invariably begin with kick drums, and drum-types most associated with downbeats are kick drums. This is also the case for breakbeat usage within HJDB, where artists mostly apply downbeat-preserving transformations, in which segments are reordered and manipulated in such a way to preserve the perception of downbeats. Third, C_A may diverge due to a mismatch in training and testing data. The training data contains percussion-only sections of audio, while the testing data is comprised of excerpts of full HJDB pieces, which may include a variety of transformations (e.g., pitch modifications) to the original breakbeats. To overcome these potential problems, we propose subsequent stages to improve the accuracy of the model: post-processing of C_A (Section 2.3); extraction of additional metrical information—namely, a low-frequency detection function (Section 2.4) and weighting at beat-times (Section 2.5); and information fusion with a final estimation of downbeats by dynamic programming (Section 2.6). An overview of the complete algorithm is presented in Figure 1.

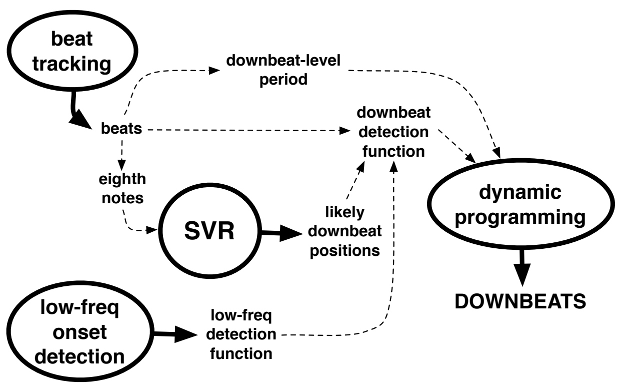


Figure 1: Overview of proposed method. Circles denote stages in the method; solid lines point to variables created in these stages; and dotted lines point to variables created in subsequent steps.

2.3 Regression Output Post-processing

As we are unable to rely solely on the derivative of C_A for an exact location of downbeats, we propose its use in providing a coarse estimation of downbeats. We create likely downbeat position function, E, as the first-order coefficient of the linear regression at each eighth-note position, by applying linear regression of a sliding buffer of eight segments (equivalent to the length of a measure) across C_A. If the eight points of C_A under analysis resemble a positive linear slope, as they do at downbeats, the value of E will be positive. As the buffer shifts, such that it no longer begins on a downbeat (but now includes a downbeat at buffer position 8), the value of E will decrease as it will no longer maintain a positive linear slope. Once the buffer has reached the end of C_A, E is normalized to values between 0 and 1.

2.4 Low-Frequency Onset Detection

The coarseness of E led us to incorporate low-level onset event information related to salience and timing. We introduce a low-frequency onset detection function, L, as follows: As in [6], we segment the input audio into 40 ERB-spaced sub-bands and calculate complex spectral difference across each (with a temporal resolution of 11.6 msec per onset detection function sample). We apply our knowledge of standard usage of basic rock drum kit drum-types (i.e., kick drum, snare drum, and hi-hats) within breakbeats and HJDB music. Since drum types found at downbeats are likely to be kick drums, we focus on lower frequencies and sum the output of the lowest ρ bands to produce L. While the precise number of bands is not critical, we found ρ=5 to provide adequate results.

2.5 Beat-Time Weighting

In Section 2.1, beat time locations, γ, are used to create the eighth-note grid used in the segmentation of the test audio for the SVR model. We also use γ to generate a beat-time weighting, U, for emphasis in L. At γ (here quantized to the resolution of L), U=ω, and otherwise U=1. The precise value of ω is not crucial, however we found ω=1.3 to perform well. To contend with alignment issues of beat times and peaks in L, we additionally weight U=ω at ± 2 detection function samples of γ.

2.6 Information Fusion and Decision

In this stage, we combine low-frequency onset detection function, L, with beat-time weighting, U, and likely downbeat position function, E, to create a final detection function, Θ, used in the determination of downbeat times.

Our motivation in combining these three forms of information is as follows: L provides low-level information pertaining to event location and salience, while E provides informed knowledge of likely downbeat positions based on similarity of the test segment patterns to patterns of drums in the breakbeat training set. The integration of beat-time weighting provides alternate possible downbeat positions that E has either missed or erroneously measured.

As none of these information sources alone is capable of accurate downbeat detection, our hope is that fusing them in a meaningful way will create a hybrid detection function that imparts the key attributes of each, resulting in a more robust detection function from which we will select downbeats. We first interpolate E to match the temporal resolution of L. We then combine L, E, and U:

$$\Theta = (L(1 + E)) * U, \tag{1}$$

where $*$ refers to element-wise multiplication.

An example of the usefulness of both E and U in emphasizing peaks of L at likely downbeat positions (and suppressing peaks not likely associated with downbeats) is presented in Figure 2. The top graph shows L (solid line) without scaling by E (dot-dashed line), and annotated downbeat positions (vertical dashed line). The middle graph shows L after scaling by E (solid line). The bottom graph depicts L after scaling by E and U (solid line).

For the final selection of downbeat positions from Θ, we require a peak-finding method capable of finding strong peaks that exist at regular intervals. Dynamic programming (DP) has been shown useful for such purposes in beat detection [9]. We similarly adopt DP to find downbeats within Θ, with a likely downbeat period τ. Given a high probability of 4/4 time signature and steady tempo in HJDB, it is sufficient to estimate τ as 4 times the median of all inter-beat intervals derived from γ.

3. EVALUATION

The aim of our evaluation is to determine the efficacy of our method and four general models on a dataset solely consisting of HJDB. In this section, we present our dataset, algorithms under evaluation, and methodology.

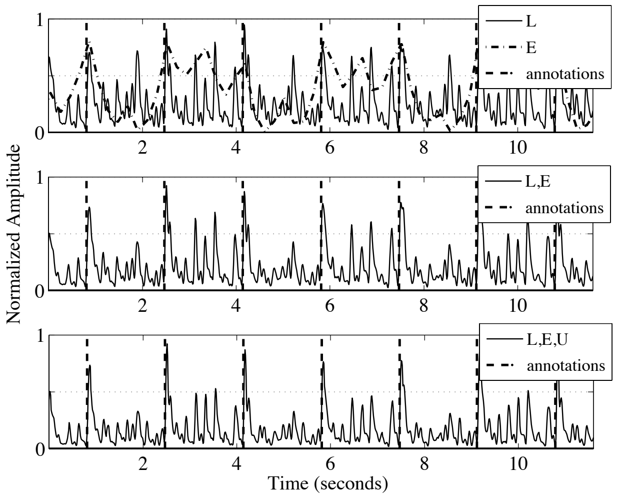


Figure 2: Effect of stages in information fusion: (*top*) L with no scaling, E, and annotations; (*middle*) L scaled by E, and annotations; (*bottom*) L scaled by E and U, and annotations.

3.1 Hardcore, Jungle and Drum and Bass Dataset

Our dataset is comprised of 236 excerpts[3] of between 30 seconds and 2 minutes in duration. Each excerpt was selected from a full-length HJDB piece digitized from its original vinyl format to a 16-bit/44.1kHz WAV file. The pieces span the five years (1990–4) of hardcore's subtle transformation through jungle and into drum and bass.

Well-known, popular HJDB pieces were chosen for inclusion in the dataset. An effort was taken to ensure a wide distribution of artists, styles, and breakbeats used; three professional HJDB DJs were consulted for their opinions. Downbeat annotations were made by a professional drum and bass musician using Sonic Visualiser.[4] 30 excerpts were removed from the test dataset to create a separate parameter tuning dataset used to optimize the parameters in the algorithm presented in Section 2. The remaining 206 excerpts were then used in our evaluation.

3.2 Evaluation Methodology

For evaluation metrics, we chose to modify the continuity-based beat tracking evaluation metrics used in the MIREX 2011 beat-tracking evaluation [4]. The principal difference is that we assess downbeats as the subject of evaluation, rather than beats. Additional modifications include adjustment of the tolerance window threshold, alteration of the possible interpretations of the downbeat to reflect whole beat offsets, and exclusion of the longest continually correct segment metric in [4]. We create a tolerance window of 1/16th note around each annotated downbeat in our dataset (i.e., 6.25% of the inter-annotation-interval). For an estimated downbeat to be correct, it must fulfill three conditions: First, it must be located within the 6.25% tolerance window around the nearest annotation. Second, the previous estimated downbeat must be located

[3] For the track list, see: http://ddmal.music.mcgill.ca/breakscience/dbeat/
[4] http://www.sonicvisualiser.org/

within the 6.25% tolerance window around the previous annotation. Finally, the inter-downbeat-interval must be within 6.25% of the inter-annotation-interval. We then count the total number of correct downbeats and provide a mean accuracy for a given excerpt. Among the various beat offsets allowed by our evaluation measure, our main interest is in the **1** statistic, which indicates how well the estimated downbeats align with annotations. **1** is the mean accuracy across all excerpts. We provide additional statistics, **2**, **3**, and **4**, to quantify errors in downbeat estimations, offset by whole beats. A potential problem for general models is HJDB's fast tempo. We therefore include an additional metric, **1/2x**, which provides an error statistic for estimated downbeats found at the half-tempo rate. **1/2x** is calculated by using the evaluation method above, with the annotations sub-sampled by a factor of two.

3.3 Algorithms Included in Evaluation

Our evaluation focuses on a comparison of the performance of the HJDB-specialized model with four generalized models. We expect this evaluation to be challenging for generalized models due to the lack of harmonic change, fast tempo, and high note density in HJDB music. We compare the following five models: commercial software #1 (CS1); commercial software #2 (CS2); Klapuri et al. (KL) [13]; Davies and Plumbley (MD) [5]; and our HJDB specialized method (HJ). The MD and KL methods are briefly described in Section 1.2. CS1 and CS2 are commercial products from two separate companies.[5] As we do not have access to the methods in CS1 or CS2, we treat them as black boxes.

4. RESULTS AND DISCUSSION

4.1 Parameter-Tuning Set Results

We first compare results of four possible configurations of our model using the 30-excerpt parameter-tuning set, to determine the best system to use in the full evaluation (Section 4.2). Table 1 presents results for these configurations using the **1**, **2**, **3**, and **4** statistics described above. While two of the configurations do not contain beat-time weighting, U, all configurations contain the dynamic programming stage with likely downbeat-level periodicity τ, derived from beats. Informal evaluation of Beatroot's performance on our dataset resulted in an F-measure of 83.0%. The base system (labeled LDF) containing low-frequency detection function, L, performs well, which demonstrates the effectiveness of focusing on kick drum frequencies. Adding either emphasis U (LDF, U) at estimated beat times or estimated likely downbeat detection function E (LDF, E) has a similar positive effect. Adding both U and E has a further positive effect, indicating independence between these features. In addition, errors in statistics **2**, **3**, and **4** in either LDF, U or LDF, E are reduced by addition of the other features—e.g., the 6% error found in LDF,E in the **4** statistic is reduced to 3.3%. Similarly, the 2.8% error found in the LDF, U on the **2** statistic is reduced to 0.6%. Addition of either or both emphasis U or E results in an improvement in accuracy over LDF alone, and a reduction in error rates **2**, **3**, and **4**.

[5] of which one was a beta version

	1	**2**	**3**	**4**
LDF	72.8	3.7	3.4	6.4
LDF, E	79.3	0.8	9.6	6.0
LDF, U	79.9	2.8	**2.8**	4.8
LDF, U, E	**83.4**	**0.6**	3.1	**3.3**

Table 1: Accuracy measure **1** and error metrics **2**, **3**, **4** (in percentages) for four configurations of the presented system using the parameter-tuning dataset. Bold scores denote highest accuracy in **1**, and lowest error in **2**, **3**, **4**.

4.2 HJDB Evaluation Results

Evaluation performance for the five compared methods is displayed in Table 2. Our specialized algorithm HJ (using the LDF, U, E configuration) performs best in the **1** statistic. In addition, HJ achieves the smallest **2** and **1/2x** error statistics (with a low **4** error rate), which when coupled with high **1** performance, is seen rather favorably.

	1	**2**	**3**	**4**	**1/2x**
CS1	38.5	2.8	**4.0**	4.2	2.8
CS2	7.4	11.7	9.5	6.7	1.1
KL	51.3	2.8	9.6	**0.2**	3.0
MD	29.3	4.7	5.5	3.0	1.2
HJ	**74.7**	**2.3**	5.8	2.0	**0.0**

Table 2: Accuracy measure **1** and error metrics **2**, **3**, **4**, **1/2x** (in percentages) for the five models under evaluation using HJDB test dataset. Bold scores denote highest accuracy in **1**, and lowest error in **2**, **3**, **4**, **1/2x**.

When a model finds a downbeat on beats two or four in HJDB music, it is likely to indicate a preference for high-energy note events such as snares (often played on beats two and four). All models have some degree of error reported in the **3** metric, possibly due to similarities in breakbeat drum patterns starting on beats one and three, which results in a confusion of phrase boundaries at these positions. Surprisingly, none of the models displayed an affinity for the **1/2x** metric that our intuition led us to believe generalized models would find more favorable.

4.3 Discussion

While our specialized method outperformed the generalized models, results should be examined with the understanding that only our approach had access to the parameter-tuning set used to adjust parameters of the SVR algorithm. While this may make the comparison somewhat imbalanced, our model is the only algorithm necessitating such parametric tuning, as the other models are general approaches. We have incorporated specific attributes of HJDB music in a model used for its analysis: information about timbre, pitch, and loudness of segments; knowledge of likely patterns; and emphasis on kick drum events and potential downbeat candidates at beat locations. Intuition tells us that the model in its present configuration may not perform as well in a generalized evaluation or niche genres excluding breakbeats, as downbeats in these datasets may not be conveyed similarly.

5. CONCLUSIONS AND FUTURE WORK

We have presented a style-specific model for finding downbeats in music that we applied to hardcore, jungle and drum and bass. At the core of our approach is a learning technique trained on classic breakbeats that form the rhythmic and timbral basis of these musical styles. We expanded this model to incorporate information related to likely onsets in low-frequency bands and beat tracking. Through fusion of these complementary information sources we create a downbeat detection function from which we infer downbeats using dynamic programming.

Evaluation of our style-specific model with generalized downbeat detection methods demonstrates a wide gap in performance. This not only highlights the efficacy of our approach in the confines of HJDB, but also provides further evidence towards the style-specific nature of downbeat detection. We consider the latter conclusion more critical, and expect our method to be less effective in music without breakbeats, and in music in which downbeats are conveyed by chord changes.

In building our model we have attempted to keep as many components as general as possible, leaving the training of the SVR as the sole part explicitly style-adapted to HJDB. In this way, we believe our approach could be readily adapted to other music styles through style-specific training of the SVR. This strategy will form a key component of our future work; both by training multiple models on different styles and investigating methods for automatic selection between these models. We believe the most profitable future advances in downbeat detection will be style-specific, rather than generalized models. Within the domain of HJDB music, we intend to harness the knowledge of downbeats to explore the relationships between the musical corpus and specific breakbeats amid a large-scale study of the genres.

6. ACKNOWLEDGEMENTS

This work is partially funded by the Social Sciences and Humanities Research Council of Canada, the ERDF through the Programme COMPETE and by the Portuguese Government through FCT—Foundation for Science and Technology, project ref. PTDC/EAT-MMU/112255/2009. The authors would like to thank Conor O'Dwyer (*Code*), Jason Chatzilias (*0=0*), and Daniel Lajoie (*ESB*) for their contributions and fruitful discussion.

7. REFERENCES

[1] B. Belle-Fortune, All crews, Vision, London, 2004.

[2] A. Z. Broder, S. C. Glassman, M. Manasse, and G. Zweig, "Syntactic clustering of the web." *J. of Comp. Networks*, Vol. 29, No. 8, pp. 1157–66, 1997.

[3] N. Collins, *Towards autonomous agents for live computer music: Realtime machine listening and interactive music systems.* PhD. diss., Cambridge University, 2006.

[4] M. E. P. Davies, N. Degara, and M. D. Plumbley, "Evaluation methods for musical audio beat tracking algorithms." Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06, 2009.

[5] M. E. P. Davies and M. D. Plumbley, "A spectral difference approach to downbeat extraction in musical audio." In *Proc. of EUSIPCO,* 2006.

[6] M. E. P. Davies, M. D. Plumbley, and D. Eck, "Towards a musical beat emphasis function." In *Proc. of WASPAA,* pp. 61–4, 2009.

[7] S. Dixon, "Evaluation of the audio beat tracking system BeatRoot." *JNMR*, Vol. 36, No. 1, pp. 39–50, 2007.

[8] S. Dixon, F. Gouyon, and G. Widmer, "Towards characterization of music via rhythmic patterns." In *Proc. of 5th ISMIR Conf.,* pp. 509–16, 2004.

[9] D. P. W. Ellis, "Beat tracking by dynamic programming." *JNMR*, Vol. 36, No. 1, pp. 51–60, 2007.

[10] E. Ferrigno, *Technologies of emotion: Creating and performing drum 'n' bass.* PhD. diss., Wesleyan University, 2008.

[11] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds." *JNMR*, Vol. 30, No. 2, pp. 159–71, 2001.

[12] T. Jehan, *Creating music by listening.* PhD. diss., Massachusetts Institute of Technology, 2005.

[13] A. P. Klapuri, A. J. Eronen and J. T. Astola, "Analysis of the meter of acoustic musical signals." *IEEE TASLP,* Vol. 14, No. 1, pp. 342–55, 2006.

[14] H. Papadopoulos and G. Peeters, "Joint estimation of chords and downbeats from an audio signal." *IEEE TASLP,* Vol. 19, No. 1, pp. 138–52, 2010.

[15] G. Peeters and H. Papadopoulos, "Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation." *IEEE TASLP,* Vol. 19, No. 6, pp. 1754–69, 2011.

[16] S. Reynolds, Energy flash: A journey through rave music and dance culture (2nd Ed.). Picador, London, 2008.

[17] "Seven seconds of fire." *The Economist,* pp. 145–6, 17th December 2011.

A MIREX META-ANALYSIS OF HUBNESS IN AUDIO MUSIC SIMILARITY

Arthur Flexer, Dominik Schnitzer, Jan Schlüter
Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria
`arthur.flexer|dominik.schnitzer|jan.schlueter@ofai.at`

ABSTRACT

We use results from the 2011 MIREX "Audio Music Similarity and Retrieval" task for a meta analysis of the hub phenomenon. Hub songs appear similar to an undesirably high number of other songs due to a problem of measuring distances in high dimensional spaces. Comparing 17 algorithms we are able to confirm that different algorithms produce very different degrees of hubness. We also show that hub songs exhibit less perceptual similarity to the songs they are close to, according to an audio similarity function, than non-hub songs. Application of the recently introduced method of "mutual proximity" is able to decisively improve this situation.

1. INTRODUCTION

In a number of recent publications [21,27,28] the so-called "hubness" phenomenon has been described and explored as a general problem of machine learning in high dimensional data spaces. Hubs are data points which keep appearing unwontedly often in nearest neighbor lists of many other data points. This effect is particularly problematic in algorithms for similarity search, as the same "similar" objects are found over and over again. In Music Information Retrieval (MIR), the hub problem has been primarily studied in the context of music recommendation based on modeling of audio similarity. Songs which act as hubs are reported as being similar to very many other songs and hence keep a significant proportion of the audio collection from being recommended at all. This paper tries to answer the following questions concerning hubs in audio music similarity which so far have not been solved to a satisfactory degree: (i) Do different parameterizations and algorithms produce different hubs? (ii) Are hub songs perceptually meaningful?

This is done by conducting a meta-analysis of 17 algorithms and utilizing 8500 human gradings of the perceptual similarity of song pairs. A recently published method [29] ("mutual proximity") is applied to reduce the negative effects of hubness.

2. RELATED WORK

One of the central notions of Music Information Retrieval (MIR) is that of *music similarity*. Proper modeling of music similarity is at the heart of every application allowing automatic organization and processing of music data bases. A fundamental constituent to computation of music similarity is timbre similarity based on parameterization of audio using Mel Frequency Cepstrum Coefficients (MFCCs) plus Gaussian mixtures as statistical modeling [22]. It is precisely for this approach to music similarity where existence of hubs has been first documented and established in MIR by Aucouturier and Pachet in 2004 [3]. Hub songs were defined as songs which are, according to the audio similarity function, similar to very many other songs and therefore keep appearing unwontedly often in recommendation lists preventing other songs from being recommended at all. Such songs that do not appear in any recommendation list have been termed "orphans". The authors further stated that hub songs "objectively have nothing to do with the seed song" [3], i.e. they share no perceptual similarity with the songs they are recommended for according to the audio similarity function. Only anectodic evidence in the form of two examples is provided for this rather general statement. Since the data set for this study was quite small (350 songs from 37 artists), the authors remark that "a further study should be done with a larger database". Similar observations about false positives in music recommendation that are not perceptually meaningful have been made elsewhere [24] using an even smaller data set.

Following this initial report about the hub problem a number of results concerning hubness in the context of MIR have been established. Aucouturier and Pachet [4] showed that hubs are distributed along a scale-free distribution, i.e. non-hub songs are extremely common and large hubs are extremely rare. This is true for MFCCs modelled with different kinds of Gaussian mixtures as well as Hidden Markov Models, irrespective whether parametric Kullback-Leibler divergence or non-parametric histograms plus Euclidean distances are used for computation of similarity. But is also true that hubness is not the property of a song per se since non-parametric and parametric approaches produce very different hubs. The hub effect is not an artefact of using small data sets in computer experiments since it also exists in very large databases (> 250000 songs) and gets even worse with growing size of databases

[14]. Not all parameterizations of audio are equally prone to hubness. Fluctuation patterns (FP) [16, 23] have been shown to produce almost no hubs and a combination of MFCCs and FPs is able to reduce hubness while maintaining an overall high quality of audio similarity [12, 14]. It has also been noted that audio recorded from urban soundscapes, different from polyphonic music, does not produce hubs [2] since its spectral content seems to be more homogeneous and therefore probably easier to model. The same has been observed for monophonic sounds from individual instruments [17]. Direct interference with the Gaussian models during or after learning has also been explored (e.g. homogenization of model variances) although with mixed results. Whereas some authors report an increase in hubness [4], others observed the opposite [18]. Using a Hierarchical Dirichlet Process instead of Gaussians for modeling MFCCs seems to avoid the hub problem altogether [20]. The existence of the hub problem has also been reported for music recommendation based on collaborative filtering instead of on audio content analysis [8]. Similar effects exist for image [10, 19] and text retrieval [28] making this phenomenon a general problem in multimedia retrieval and recommendation.

Berenzweig [7] was probably the first to suspect a connection between the hub problem and the high dimensionality of the feature space. The hub problem was seen as a direct result of the curse of dimensionality [5], a term which refers to a number of challenges due to the high dimensionality of data spaces. Radovanović et al [27, 28] were able to provide more insight by linking the hub problem to the property of *concentration* [15] which occurs as a natural consequence of high dimensionality. 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 [1, 15]. Radovanović et al [28] 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. Points which are further away from the center have a high probability of being "orphans", i.e. points that never appear in any nearest neighbor list. This concentration of distances has also been reported for audio data [21].

Already in the context of concentration of distances it has been noted that the degree of concentration depends on the intrinsic rather than embedding dimension of the feature space [15]. Whereas the embedding dimension is the actual number of dimensions of a feature space the intrinsic dimension is the, often much smaller, number of dimensions necessary to represent a feature space without loss of information. It has also been demonstrated that hubness depends on the intrinsic rather than embedding dimensionality [28].

A direct consequence of the presence of hubs is that a large number of nearest neighbor relations in the distance space are asymmetric, i.e., a hub y is the nearest neighbor of x, but the nearest neighbor of the hub y is another point a ($a \neq x$). This is because hubs are nearest neighbors to very many data points but only k data points can be nearest neighbors to a hub since the size of a nearest neighbor list is fixed. This behavior is especially problematic if x and y belong to the same class but a does not, violating the pairwise stability of clusters [6]. In a recent publication [29] a general unsupervised method to attenuate the negative effects of hubness by repairing asymmetric nearest neighbor relations has been presented. It transforms arbitrary distance matrices to matrices of so-called probabilistic mutual proximity (MP). On a range of audio data sets it has been demonstrated that it is indeed able to decrease hubness while improving audio similarity as measured with genre classification accuracy. Since we will use this method to improve results for the MIREX data set it will be described in more detail in section 5.3. Please note that MP can be seen as a refinement of the so-called "P-norm" [26], which has been applied to the hub problem by other authors too [9].

3. DATA AND ALGORITHMS

For our meta-analysis of hubness we use the data from the recent 2011 "Audio Music Similarity and Retrieval" task [1] within the annual MIREX [11] evaluation campaign for MIR algorithms. Each of 18 competing algorithms was given 7000 songs (30 second audio clips). The data consists of 10 almost equally sized genre classes: 700 songs from BAROQUE, COUNTRY, EDANCE, JAZZ, METAL, RAPHIPHOP, ROCKROLL, ROMANTIC, 699 from BLUES, 701 from CLASSICAL. Every algorithm was given these 7000 song excerpts and returned either a full 7000×7000 distance matrix (algorithms CTCP1, CTCP2, CTCP3, DM2, DM3, ML1, ML2, ML3, SSKS3, SSPK2, STBD1, STBD2, STBD3) or a matrix of size 7000×100 (GKC1, HKHLL, PS1, YL1) containing the first 100 nearest neighbors to each song. The resulting distance matrix for algorithm ZYC2 is faulty containing the same distance of the same pair of songs over and over again and is therefore excluded from our analysis [2]. Please note that some of the systems are very closely related, sometimes using just different parameters for the same algorithm (e.g. CTCP1-3, DM2-3, ML1-3, STBD1-3).

From the 7000 songs, "100 songs were randomly selected from the 10 genre groups (10 per genre) as queries

[1] The 2011 results and details can be found at: `http://www.music-ir.org/mirex/wiki/2011:Audio_Music_Similarity_and_Retrieval_Results`

[2] Please note that ZYC2 did participate in the MIREX task and even scored in the mid-field of results. This seems to be due to nonrandom genre order during evaluation and not to its real performance.

and the first 5 most highly ranked songs out of the 7000 were extracted for each query (after filtering out the query itself, returned results from the same artist were also omitted). Then, for each query, the returned results (candidates) from all participants were grouped and were evaluated by human graders" [1]. For each individual query/candidate pair, a single human grader provided both a FINE score (from 0 (failure) to 100 (perfection)) and a BROAD score (not similar NS, somewhat similar SS, very similar VS) indicating how similar the songs are in their opinion. Since we use FINE scores only, this altogether gives $17 \times 100 \times 5 = 8500$ human gradings for our analysis.

4. EVALUATION

The following measures are used to evaluate the hubness phenomenon in section 5. Abbreviations correspond to labels in the result table 1.

k-occurrence statistics (H25/cov, H50/cov, maxH): As a measure of the hubness of a given song we use the so-called k-occurrence N_k [4], i.e. the number of times the song occurs in the first k nearest neighbors of all the other songs in the data base. Please note that the mean k-occurrence across all songs in a data base is equal to k. Any k-occurrence significantly bigger than k therefore indicates existence of a hub. Since human graders evaluated the five most similar songs we used $k = 5$. We compute the absolute number of the maximum k-occurrence $maxH$ (i.e. the biggest hub) and the number of songs for which the k-occurrence is bigger than 25 or 50 (i.e. the number of small hubs $H25$ and large hubs $H50$). Additionally we give the number of these hubs that appear as candidate songs in the human grading evaluation, e.g. "$H25/cov = 6/3$" means that out of six hub songs with k-occurrence bigger than 25 three songs have been evaluated (covered) by human graders.

Hubness (*hub*): We compute the hubness for each algorithm's distance matrix according to Radovanović et al. [28]. Hubness is defined as the skewness of the distribution of k-occurrences N_k of a whole data set. Positive skewness indicates high hubness, negative values low hubness.

Reachability (reach): This is the percentage of songs from the whole data base that are part of at least one of the nearest neighbor lists. If a song is not part of any of the recommendation lists of size $k = 5$ it is an orphan song which will never be recommended as a candidate song.

Number of hub gradings (#H): For every algorithm, 500 human gradings exist for further analysis. The number of hub gradings is the number of gradings where the candidate song in a query/candidate pair is a hub song. This is given using k-occurrences of 25 (H25) and 50 (H50) to distinguish between hubs and non-hubs.

Fine Score (fineH, fineNH, fine): To evaluate the perceptual quality of hubs and non-hubs we compute average fine scores. For every song in the whole data base we check whether it was the candidate in any of the query/candidate pairs in the human evaluation experiment. We average all respective fine scores for hub songs (fineH) and non-hub songs (fineNH) separately. This is done using k-occurrences of 25 (H25) and 50 (H50) to distinguish between hubs and non-hubs. We also include the average across all fine scores (fine) irrespective of whether candidate songs are hubs or not.

Accuracy (acc): To evaluate the quality of audio similarity for the whole data base, and not just for the songs which have been evaluated by human graders, we computed the genre classification performance. Since songs within a certain genre will also sound more similar than songs from different genres, high genre classification results indicate good audio similarity measures. It has also been demonstrated that algorithms achieving high genre classification results are able to produce results that correlate higher with human music similarity judgements [25, p. 26]. We compute the $k = 5$-nearest neighbor classification accuracy, i.e. the percentage of the five nearest songs that have the same label as the query song. Songs from the same artist as a query song are omitted from the nearest neighbor list by using an artist filter [13].

5. RESULTS

All results discussed in the following section can be found in table 1 listing all evaluation measures (from column hub to $fineNH$, see section 4) for all algorithms (from CTCP1 to YL1, see section 3) plus improved results using "mutual proximity" (rows mp, see section 5.3).

5.1 Hubness across algorithms

As already discussed in section 2 hubness is not a property of an individual song but is connected to what features are computed from the audio, what models are being learned from the features and how these processing steps are affected by the concentration of distances in high dimensional spaces. The MIREX audio similarity results provide the opportunity to compare 17 different algorithms with respect to their hubness. Looking at the results in table 1 it is apparent that the different algorithms produce very different degrees of hubness. The hubness values (column hub) range from 0.96 (HKHLL1) to 3.98 (STBD3) indicating that all distributions of k-occurrences are skewed to the right, i.e. are prone to hubness. Looking at the numbers of small ($H25$) and large ($H50$) hubs it is also clear that the different algorithms produce very different numbers of hubs. The number of small hubs range from 0 (SSKS3) to 256 (STBD3), those of large hubs from 0 (CTCP1-3, GKC1, HKHLL1, ML1, ML3, SSKS3, SSPK2) to 60 (STBD3). The largest k-occurrence $maxH$ ranges from 21 (HKHLL1) to 122 (STBD3). The average correlation of k-occurrences of all songs across all pairs of algorithms is only 0.14 showing that different algorithms produce very different hubness for a song. It is interesting to note that closely related algorithms show much higher correlations (e.g. an average of 0.76 for CTCP1, CTCP2 and CTCP3). The reachability $reach$ ranges from small 65.5% (STBD3) up to 95.9% (SSKS3).

								H25			H50		
algo	hub	acc	H25/cov	H50/cov	maxH	reach	fine	#H	fineH	fineNH	#H	fineH	fineNH
CTCP1	1.28	59.42	6/3	0/0	31	93.8	57.3	3	60.0	57.3	0	-	57.3
mp	1.32	58.75	9	0	35	92.5							
CTCP2	1.08	59.66	1/1	0/0	26	94.9	58.6	2	41.0	58.7	0	-	58.6
mp	1.02	59.20	0	0	25	94.4							
CTCP3	1.41	60.07	12/3	0/0	35	92.3	56.2	3	44.3	56.3	0	-	56.2
mp	1.41	59.38	10	0	36	90.3							
PS1	2.43	59.52	32/15	2/2	81	88.0	57.7	19	55.9	57.8	2	76.5	57.6
mp	1.02	54.80	1	0	31	99.1							
SSKS3	0.93	60.12	0/0	0/0	25	95.9	58.1	0	-	58.1	0	-	58.1
mp	1.13	59.61	3	0	31	93.8							
SSPK2	1.19	59.67	5/3	0/0	33	94.5	58.6	3	73.0	58.6	0	-	58.6
mp	1.33	58.95	5	0	38	93.7							
DM2	2.80	50.62	68/22	4/3	90	84.2	50.5	29	48.2	50.6	5	36.2	50.6
mp	2.15	51.19	40	1	58	88.9							
DM3	2.83	50.69	76/32	7/3	85	84.4	50.3	43	47.2	50.6	4	42.0	50.4
mp	2.15	51.03	48	1	58	88.9							
GKC1	1.31	25.83	11/3	0/0	34	90.8	31.8	3	30.0	31.9	0	-	31.8
mp	0.16	26.52	0	0	15	97.8							
HKHLL1	0.96	38.40	0/0	0/0	23	93.3	42.2	0	-	42.2	0	-	42.2
mp	0.48	38.76	0	0	24	98.3							
ML1	2.19	45.95	61/22	0/0	47	85.9	47.8	26	46.5	47.9	0	-	47.8
mp	1.06	48.22	2	0	30	93.8							
ML2	2.17	44.22	60/19	1/0	54	87.4	47.3	25	36.1	47.9	0	-	47.3
mp	1.08	45.74	2	1	30	94.4							
ML3	1.49	45.17	12/3	0/0	38	90.8	47.8	3	47.7	47.8	0	-	47.8
mp	0.94	44.74	1	0	26	95.2							
STBD1	3.46	26.62	161/71	17/10	90	81.0	33.9	86	29.5	34.8	16	22.9	34.3
mp	1.40	28.81	5	0	45	93.7							
STBD2	2.88	25.91	135/60	9/5	70	82.0	30.6	72	28.0	31.0	6	21.3	30.7
mp	1.24	27.37	3	0	34	95.5							
STBD3	3.98	25.38	256/113	60/35	122	65.5	30.4	149	29.3	30.9	54	23.0	31.3
mp	2.18	26.88	64	2	57	86.1							
YL1	2.16	41.14	65/21	1/0	54	84.4	42.4	27	33.4	42.9	0	-	42.4
mp	1.82	41.01	2	1	65	95.6							

Table 1. All results (please see section 5 for details) for all evaluation measures (from column hub to $fineNH$, see section 4) for all algorithms (from CTCP1 to YL1, see section 3) plus improved results using "mutual proximity" (rows mp, see section 5.3). Algorithms CTCP1 to SSPK2 (top six rows) already use "P-norm" and therefore do not show improvements due to mp, see section 5.3.

To sum up, different algorithms indeed produce very different degrees of hubness.

5.2 Hubness and perceptual quality

The next question we like to clarify is whether hub songs exhibit less perceptual similarity to the songs they are close to (according to an audio similarity function) than non-hub songs.

The correlation between hubness and average fine score of all algorithms (columns hub and $fine$ in table 1) is -0.56. This indicates that algorithms generating large hubness show low fine scores, i.e. overall bad perceptual similarity. Notable exceptions are maybe GKC1 with low hubness and low fine scores ($hub = 1.31$, $fine = 31.8$) and PS1 with rather high hubness and high fine scores ($hub = 2.43$, $fine = 57.7$).

Analyzing the differences in fine scores between hubs and non-hubs, we can see that the average fine scores for small hubs ($fineH$,$H25$) are almost always smaller than those for non-hubs ($fineNH$,$H25$). The only exceptions are algorithms CTCP1 and SSPK2. The average difference in fine scores is 3.65. This average is taken across all algorithms where grading information for hubs is available ($\#H > 0$). The average fine scores for large hubs ($fineH$,$H50$) are again almost always smaller than those for non-hubs ($fineNH$,$H50$), with algorithm PS1 being the only exception. The average difference in fine scores is 5.49. Again this average is taken across all algorithms where grading information for hubs is available ($\#H > 0$). Please note that e.g. for PS1, only two human gradings of large hub songs do exist.

To sum up, both small and large hubs seem to be less perceptually meaningful than non-hub songs but the average difference in human gradings is only 3.65 to 5.49 on a scale from 0 to 100. Audio similarity computed with algorithms showing high hubness seems to be less perceptually meaningful than that of algorithms with low hubness in general.

5.3 Reducing hubness

To reduce the negative effects of hubness we apply "mutual proximity" (MP) [29] to the distance matrices from all algorithms. MP takes a distance matrix and (i) transforms 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) combines these (asymmetric) probabilistic distances from x to y and y to x via the product rule. 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 thereby repairing sometimes contradicting, asymmetric nearest neighbor information which seems to cause hubness in similarity measures. Please note that MP requires knowledge of the full distance matrix since it needs to compute means and variances across full rows and columns during the re-scaling step. For algorithms GKC1, HKHLL, PS1 and YL1 we only have distances to the first 100 nearest neighbors of each song. In this case we use this limited set of distances instead of full rows and columns for computation of MP.

Before discussing the improvements due to MP it has to be said that six of the competing algorithms (CTCP1-3, PS1, SSKS3, SSPK2) already use a transformation of the distance matrix similar to MP. Usage of the so-called "P-norm" [26], which can be seen as a predeccesor to MP, together with application of MP does not seem to improve hubness. As a matter of fact, it sometimes even worsens the hubness situation. On the other hand, the six algorithms employing the "P-norm" already are the six best ranking systems according to their average fine score. Therefore we now discuss only those algorithms that do not use the "P-norm" already (DM2-3, GKC1, HKHLL1, ML1-3, STBD1-3, YL1).

Comparing hubness indices (column hub in table 1) between original algorithms and their improved MP version (rows mp) it can be seen that all values improve. The average decrease in hubness (hub) is 45.5%. The number of small hubs $H25$ also always decrease with an average of 83% less hubs. The average decrease in the number of large hubs $H50$ is 79.6%. The average decrease of the largest hub $maxH$ is 33.2%. Only HKHLL1 and YL1 show a slightly larger $maxH$, with all other indices also diminishing. The reachability $reach$ for all algorithms is enhanced, on average by 8.95 percentage points. This means that audio simialrity re-scaled with MP produces less orphan songs and includes a larger part of the data base in the nearest neighbor lists. All these improvements seem to be accompanied with unchanged quality in audio similarity. At least all genre classification results (column acc) remain more or less constant, with some insignificant increases and decreases.

To sum up, mutual proximity (MP) is able to decisively improve the hubness situation while not changing the overall performance in audio similarity.

6. DISCUSSION AND CONCLUSION

In this paper we were able to explore two important questions concerning hubness in audio music similarity which so far have not been answered satisfactory. We have corroborated earlier results indicating that different features computed from the audio in combination with different models being learned produce very different degrees of hubness. This was done by comparing a yet unprecedented number of different approaches (17 algorithms from the 2011 MIREX "Audio Music Similarity and Retrieval" task). We were also able to show that hub songs, when being recommended as being very similar, are judged to be less perceptually meaningful than non-hub songs by human evaluators. This was done by conducting the first systematic and extensive study on the perceptual quality of hub songs based on human evaluations again using MIREX data. Last but not least we were able to show that it is possible to reduce the many negative effects of hubness by applying "mutual proximity" to re-scale audio similarity distances.

7. ACKNOWLEDGEMENTS

Many thanks are due to the spiffy MIREX people (especially Stephen Downie and Andreas Ehmann) for conducting the audio music similarity and retrieval evaluation in the first place and for making the data needed for our research available. This research is supported by the Austrian Science Fund (FWF, grant P24095) and the Vienna Science and Technology Fund (WWTF, project "Audiominer" MA09-024).

8. REFERENCES

[1] Aggarwal C.C., Hinneburg A., Keim D.A.: On the Surprising Behavior of Distance Metrics in High Dimensional Spaces, Proceedings of the 8th International Conference on Database Theory, Springer-Verlag London, UK, pages 420-434, 2001.

[2] Aucouturier J.-J., Defreville B., Pachet F.: The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music, Journal of the Acoustical Society of America, 122 (2), 881-891, 2007.

[3] Aucouturier, J.-J., Pachet F.: Improving Timbre Similarity: How high is the sky?, Journal of Negative Results in Speech and Audio Sciences, 1(1), 2004.

[4] Aucouturier J.-J., Pachet F.: A scale-free distribution of false positives for a large class of audio similarity measures, Pattern Recognition, Vol. 41(1), pp. 272-284, 2007.

[5] Bellman R.E.: Adaptive Control Processes: A Guided Tour, Princeton University Press, 1961.

[6] Bennett K.P., Fayyad U., Geiger D.: Density-based indexing for approximate nearest-neighbor queries, Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '99, pages 233-243, New York, NY, USA, 1999.

[7] Berenzweig A.: Anchors and hubs in audio-based music similarity, PhD thesis, Columbia University, New York, USA, 2007.

[8] Celma, O.: Music Recommendation and Discovery in the Long Tail, PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2008.

[9] Charbuillet C., Tardieu D., Peeters G.: GMM supervector for content based music similarity, Proceedings of the 14th Int. Conference on Digital Audio Effects (DAFx-11), Paris, France, 2011.

[10] Doddington G., Liggett W., Martin A., Przybocki M., Reynolds D.A.: SHEEP, GOATS, LAMBS and WOLVES: A Statistical Analysis of Speaker Performance in the NIST 1998 Speaker Recognition Evaluation, in Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP'98), Sydney, Australia, 1998.

[11] Downie J.S.: The Music Information Retrieval Evaluation eXchange (MIREX), D-Lib Magazine, Volume 12, Number 12, 2006.

[12] Flexer A., Gasser M., Schnitzer D.: Limitations of interactive music recommendation based on audio content, Proc. of the 5th Audio Mostly Conference, pp. 96-102, 2010.

[13] Flexer A., Schnitzer D.: Effects of Album and Artist Filters in Audio Similarity Computed for Very Large Music Databases, Computer Music Journal, Volume 34, Number 3, pp. 20-28, 2010.

[14] Flexer A., Schnitzer D., Gasser M., Pohle T.: Combining features reduces hubness in audio similarity, Proc. of the Eleventh International Society for Music Information Retrieval Conference (ISMIR 2010), 2010.

[15] Francois D., Wertz V., Verleysen M.: The concentration of fractional distances, IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No. 7, pp. 873-886, 2007.

[16] Fruehwirt M., Rauber A.: Self-Organizing Maps for Content-Based Music Clustering, Proceedings of the Twelth Italian Workshop on Neural Nets, IIAS, 2001.

[17] Gasser M., Flexer A., Schnitzer D.: Hubs and Orphans - an Explorative Approach, Proceedings of the 7th Sound and Music Computing Conference (SMC'10), 2010.

[18] Godfrey M.T., Chordia P.: Hubs and Homogeneity: Improving Content-Based Music Modeling, Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR'08), 2008.

[19] Hicklin R.A., Watson C.I., Ulery B.: The Myth of the Goats: How Many People Have Fingerprints that are Hard to Match?, The National Institute of Standards and Technology (NIST), NIST Interagency/Internal Report (NISTIR) - 7271, 2005.

[20] Hoffman M., Blei D., Cook P.: Content-Based Musical Similarity Computation Using the Hierarchical Dirichlet Process, Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR'08), 2008.

[21] Karydis I., Radovanović M., Nanopoulos A., Ivanović M.: Looking through the "glass ceiling": A conceptual framework for the problems of spectral similarity, Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR'10), pages 267-272, 2010.

[22] Logan B.: Mel Frequency Cepstral Coefficients for Music Modeling, Proceedings of the International Symposium on Music Information Retrieval (ISMIR'00), 2000.

[23] Pampalk E.: Computational Models of Music Similarity and their Application to Music Information Retrieval, Vienna University of Technology, Austria, Doctoral Thesis, 2006.

[24] Pampalk E., Dixon S., Widmer G.: On the Evaluation of Perceptual Similarity Measures for Music, Proceedings of the 6th International Conference on Digital Audio Effects (DAFx-03), pp. 7-12, London, U.K., 2003.

[25] Pohle T.: Automatic Characterization of Music for Intuitive Retrieval, PhD Thesis, Johannes Kepler University, Linz, Austria, 2010.

[26] Pohle T., Schnitzer D., Schedl M., Knees P., Widmer G.: On rhythm and general music similarity, Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR'09), 2009.

[27] Radovanović M., Nanopoulos A., Ivanović M.: Nearest neighbors in high-dimensional data: The emergence and influence of hubs, Proceedings of the 26th International Conference on Machine Learning (ICML'09), ACM International Conference Proceeding Series, volume 382, pages 865-872, 2009.

[28] Radovanović M., Nanopoulos A., Ivanović M.: Hubs in space: Popular nearest neighbors in high-dimensional data, Journal of Machine Learning Research, 11:2487-2531, 2010.

[29] Schnitzer D., Flexer A., Schedl M., Widmer G.: Using Mutual Proximity to Improve Content-Based Audio Similarity, in Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR'11), Miami, Florida, USA, 2011.

HOW SIGNIFICANT IS STATISTICALLY SIGNIFICANT? THE CASE OF AUDIO MUSIC SIMILARITY AND RETRIEVAL

Julián Urbano
University Carlos III of Madrid
Brian McFee
University of California at San Diego

J. Stephen Downie
University of Illinois at Urbana-Champaign
Markus Schedl
Johannes Kepler University Linz

ABSTRACT

The principal goal of the annual Music Information Retrieval Evaluation eXchange (MIREX) experiments is to determine which systems perform well and which systems perform poorly on a range of MIR tasks. However, there has been no systematic analysis regarding how well these evaluation results translate into real-world user satisfaction. For most researchers, reaching statistical significance in the evaluation results is usually the most important goal, but in this paper we show that indicators of statistical significance (i.e., small p-value) are eventually of secondary importance. Researchers who want to predict the real-world implications of formal evaluations should properly report upon practical significance (i.e., large effect-size). Using data from the 18 systems submitted to the MIREX 2011 Audio Music Similarity and Retrieval task, we ran an experiment with 100 real-world users that allows us to explicitly map system performance onto user satisfaction. Based upon 2,200 judgments, the results show that absolute system performance needs to be quite large for users to be satisfied, and differences between systems have to be very large for users to actually prefer the supposedly better system. The results also suggest a practical upper bound of 80% on user satisfaction with the current definition of the task. Reflecting upon these findings, we make some recommendations for future evaluation experiments and the reporting and interpretation of results in peer-reviewing.

1. INTRODUCTION

Evaluation experiments are the main research tool in Information Retrieval (IR) to determine which systems perform well and which perform poorly for a given task [1]. Several effectiveness measures are used to assign systems a score that estimates how well they perform. The assumption underlying these evaluations is that systems with better scores are actually perceived as more useful by the users and therefore are expected to bring more satisfaction.

Researchers are usually interested in the comparison between systems: is system A better or worse than system B? After running an experiment with a test collection, researchers have a numeric answer to that question that measures the effectiveness difference between systems. Statistical procedures are then used to check whether that difference is statistically significant or not. Statistical significance is usually thought of as a sort of bulletproof evidence that one system really is better than another. Teams usually follow one or another research line based solely on statistical significance, and it has also become an essential requirement for publication in peer-reviewed venues.

However, there are several misconceptions regarding statistical significance [2, 11]. In the case of IR evaluation experiments, null hypotheses about differences in performance are false by definition, so observing a small p-value to conclude significance is just a matter of meeting certain conditions in the experiment. On the other hand, very little attention is paid to the effect-sizes and their implications in practical terms. In fact, even if statistical significance is present, the difference between two systems may very well be so subtle that users do not note the difference.

However, IR evaluations are traditionally focused on the algorithmic aspect of the systems, and whether the results do predict user satisfaction or not is very seldom studied [8]. Evaluation experiments make different assumptions regarding the operational settings and the needs and behavior of the users, so the extent to which results can be extrapolated should be questioned [9].

In this paper we focus on the evaluation of the Audio Music Similarity and Retrieval task (AMS), as carried out in the annual Music Information Retrieval Evaluation eXchange (MIREX). AMS is one of the tasks that most closely resemble a real-world music retrieval scenario, and it is also one of the tasks that receives most attention from the research community. We carried out an experiment with 100 users that allowed us to map system effectiveness onto user satisfaction, providing a new perspective in the interpretation of evaluation results. We also argue that researchers should not only focus on achieving statistical significance in effectiveness differences, but also on the size and practical implications of such differences.

2. SYSTEM EFFECTIVENESS AND USER SATISFACTION

In the MIREX AMS evaluation experiments, the similarity of a document to a query is assessed by humans and based on two different scales. The Broad scale has three levels: 0 (not similar), 1 (somewhat similar) and 2 (very similar). The Fine scale has 101 levels, from 0 (not similar at all) to 100 (identical to the query). Only one measure is reported to assess the effectiveness of the participating systems: $AG@5$ (Average Gain after 5 documents retrieved):

$$AG@k = \frac{1}{k}\sum_{i=1}^{k} gain_i$$

where $gain_i$ is the gain of the i-th document retrieved (the similarity score assigned). Two versions of $AG@5$ are actually reported, following the Broad and Fine scales.

$AG@k$ assumes that a document retrieved at rank 3 is as useful as a document retrieved at rank 30. A measure with a more realistic user model is $nDCG@k$ (Normalized Discounted Cumulated Gain after k retrieved) [3]:

$$nDCG@k = \frac{\sum_{i=1}^{k} gain_i / \log_2(i+1)}{\sum_{i=1}^{k} gain_i^* / \log_2(i+1)}$$

where $gain_i^*$ is the gain of the i-th document in the ideal ranking (i.e. $\forall i : gain_i^* \geq gain_{i+1}^*$). The gain contribution of a document is discounted with the logarithm of the rank at which it is retrieved, thus penalizing late arrival of relevant documents. Also, the gain contribution of documents is divided by the ideal contribution, bounding the measure between 0 and 1. Therefore, and for the sake of simplicity when comparing results across measures, we normalize $AG@k$ between 0 and 1 too:

$$nAG@k = \frac{1}{k \cdot l^+}\sum_{i=1}^{k} gain_i$$

where l^+ is the maximum similarity score allowed by the scale: 2 in the Broad scale and 100 in the Fine scale.

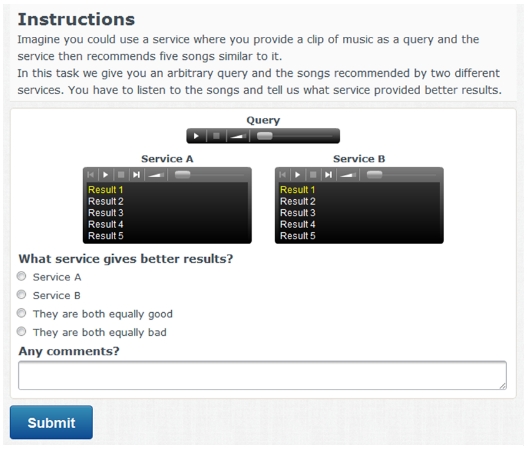
2.1 Interpretation of Effectiveness Scores

After running an evaluation of AMS systems, researchers interpret the results and make design decisions accordingly [9]. The ultimate goal is answering this question: what system would yield more user satisfaction? But we need to ask something else first: what measure and what similarity scale are better to predict user satisfaction?

Intuitively, if a system obtained a $nAG@5$ or $nDCG@5$ score of 1, our interpretation would be that an arbitrary user would be 100% satisfied with the results of the system, or satisfied 35% of the times if the effectiveness score achieved were 0.35. On the other hand, if system A obtained an effectiveness score larger than the one obtained by system B, we should expect users to prefer A. By choosing one or another measure, researchers make different assumptions as to the behavior and needs of the final users, and by choosing one or another similarity scale they follow different criteria to differentiate satisfying from unsatisfying results. To the best of our knowledge, none of these assumptions has been validated in the literature so far.

3. EXPERIMENTAL DESIGN

We devised an experiment with actual users that allowed us to map system effectiveness onto user satisfaction. Subjects were presented with a query clip and two ranked lists of five results each, as if retrieved by two different AMS systems A and B [8]. They had to listen to the clips and then select one of the following options: system A provided better results, system B did, they both provided *good* results, or they both returned *bad* results (see Figure 1). From these we can differentiate 4 judgments:


Instructions

Imagine you could use a service where you provide a clip of music as a query and the service then recommends five songs similar to it.

In this task we give you an arbitrary query and the songs recommended by two different services. You have to listen to the songs and tell us what service provided better results.

Query

Service A: Result 1, Result 2, Result 3, Result 4, Result 5

Service B: Result 1, Result 2, Result 3, Result 4, Result 5

What service gives better results?

- Service A
- Service B
- They are both equally good
- They are both equally bad

Any comments?

Submit

Figure 1. Task template used in the experiment.

- **Positive preference**, if the subject selected the system whose results yield *larger* effectiveness.
- **Negative preference**, if the subject selected the system whose results yield *smaller* effectiveness.
- **Good nonpreference**, if the subject indicated both systems are equally *good*.
- **Bad nonpreference**, if the subject indicated both systems are equally *bad*.

Such a design allows us to analyze the results from two different approaches: evaluation of a single system and comparison of two systems. Subjects indicating that both systems are *good* suggest that they are satisfied with both ranked lists. That is, their answer serves as an indication that the effectiveness measured for those systems translates into user satisfaction. If, on the other hand, they indicate that both systems are *bad*, we can infer that those effectiveness scores do not translate into user satisfaction. Subjects indicating a preference for one ranked list over the other one suggest that there is a difference between them large enough to be noted. That is, their answer serves as an indication that the difference in effectiveness between the systems translates into users being more satisfied with one system than with the other.

3.1 Data

We used the similarity judgments collected for the 2011 edition of the MIREX AMS task: a total of 18 systems by 10 research teams were evaluated with 100 queries, leading to a total of 6,322 unique similarity judgments. This is the largest edition as of the writing of this paper [1].

According to the definition of $nAG@k$ with Broad judgments, the difference between two systems is always a multiple of 0.1. For each difference $\Delta \in \{0, 0.1, ..., 1\}$, we selected 200 random queries and artificially created two random ranked lists of 5 documents such that their difference in $nAG@5$ would be Δ according to the Broad judgments made for that query in MIREX 2011. Therefore, we have a total of 2,200 examples. Note that for the extreme value $\Delta = 1$ we need at least 5 very similar documents and 5 not

[1] http://www.music-ir.org/mirex/wiki/2011:MIREX2011_Results

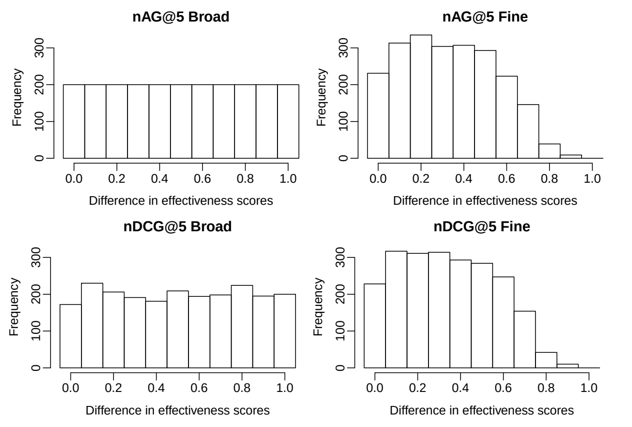


Figure 2. Distribution of effectiveness differences in all 2,200 examples, for $nAG@5$ (top) and $nDCG@5$ (bottom), and Broad (left) and Fine (right) judgments.

similar documents for the query. Due to this restriction, we could actually use only 73 of the total 100 queries. Across all 2,200 examples, we had 2,869 unique ranked lists of results, with 3,031 unique clips (including the 73 queries).

Figure 2 shows the distributions of effectiveness differences in the 2,200 examples. As mentioned, differences for $nAG@5$ with Broad judgments follow a uniform distribution, but with the Fine judgments there are very few examples with large differences. We note though that this is an artifact of the Fine scale itself and not a sampling flaw: for $\Delta = 0.9$ we need 5 documents with very high similarity scores (90 to 100) and 5 documents with very low scores (0 to 10); however, assessors very rarely assign such small and large scores. Therefore, it is very rare to observe differences that large when using the Fine scale.

These 2,200 pairs of artificial ranked lists can also be evaluated as per $nDCG@5$. As Figure 2 shows, the distributions of differences in $nDCG@5$ are very similar to $nAG@5$. Our examples do therefore cover the wide range of possible evaluation outcomes.

3.2 Procedure

All 2,200 judgments were collected via crowdsourcing. Previous work by Lee [6] and Urbano et al. [10] demonstrated that music similarity judgments gathered through crowdsourcing platforms are very similar to the ones collected with experts, with fast turnaround and low cost. Another advantage of using crowdsourcing for our experiment is that it offers a large and diverse pool of subjects around the globe. Using a controlled group of students or experts would probably bias our results, but using a diverse pool of workers allows us to draw conclusions that should generalize to the wider population of users.

However, using crowdsourcing has other issues. The quality of judgments via crowdsourcing can be questioned because some workers are known to produce spam answers and others provide careless answers to profit without actually doing the task. We decided to use the platform Crowdflower to gather the judgments, which delegates the work to other platforms such as Amazon Mechanical Turk. It also provides a quality control layer at the process level that separates good from bad workers by means of trap examples [5,8]: some of the examples shown to workers have known answers (provided by us) that are used to estimate worker quality. Workers that show low quality on the trap examples are rejected, and those that show high agreement are allowed to participate. We provided Crowdflower with 20 such trap examples (5 for each of the four answers), assigning each of them a subjective level of difficulty based on the answers by two experts.

3.3 Task Design

Figure 1 shows the task template we used. A first section listed the task instructions, and then a Flash player permitted subjects to listen to the query clip. Below, they could find the two ranked lists of 5 results each, followed by radio buttons to select the answer. Finally, a textbox was provided for workers to optionally leave feedback. All 3,031 audio clips were uploaded to our servers, and served upon request. The order in which examples are shown to workers is random, as is the assignment of the ranked lists as system A or system B. Also, we limited the maximum number of answers by a single worker to 50, minimizing the possible bias due to super-workers.

We collected all answers in four batches of 550 examples each. Lee collected similarity judgments paying \$0.20 for 15 query-document pairs [6], while Urbano et al. collected preference judgments paying \$0.02 for each query-document-document [10]. In both studies workers were therefore paid approximately \$0.007 per audio clip. Music-related tasks are known to be enjoyable by workers, and given that quality does not significantly degrade when decreasing wages [7], we decided to pay \$0.03 for each example, leading to approximately \$0.003 per clip. Adding the corresponding feeds to Crowdflower, all 2,200 judgments were collected for a grand total of \$100.

4. RESULTS [2]

The four batches were completed in less than 24 hours. We collected answers from 881 unique workers from 62 countries and 7 different crowdsourcing markets. These workers provided a grand total of 6,895 answers, from which Crowdflower accepted 3,393 (49%) as trustworthy. Note that the extra answers are due to repeatedly showing trap examples to workers. Only 100 workers were responsible for these trusted answers, so 781 workers (87%) were rejected. The average quality of these 100 workers, as computed by Crowdflower [5], ranges from 60% to 100%, with an average of 95%. In fact, 27 of our 2,200 examples contained the exact same documents, in the exact same order, in both ranked lists. Only twice did we not get, as should have, an unsigned preference in these cases. Therefore, the results reported herein comprise 2,200 answers by 100 different users who, apparently, provided honest responses.

4.1 Evaluation of a Single System

For 884 of the 2,200 examples (40%) we received a non-preference (i.e. subjects judged both systems as equally good or bad). Therefore, we have 1,768 ranked lists that subjects considered equally satisfying. Figure 3 shows the

[2] All data can be downloaded from http://julian-urbano.info.

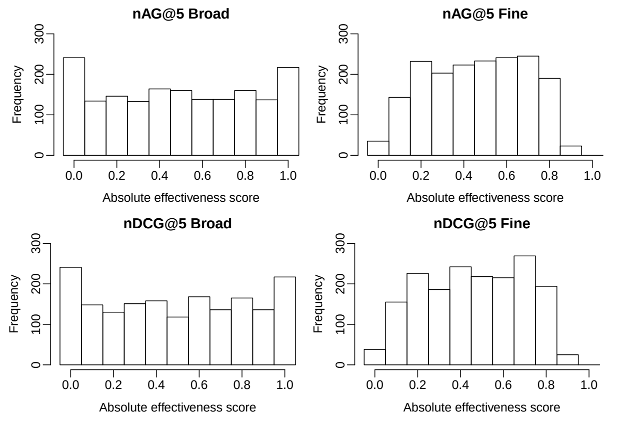


Figure 3. Distribution of absolute effectiveness scores in the 884 examples with unsigned preferences, for $nAG@5$ (top) and $nDCG@5$ (bottom), and Broad (left) and Fine (right) judgments.

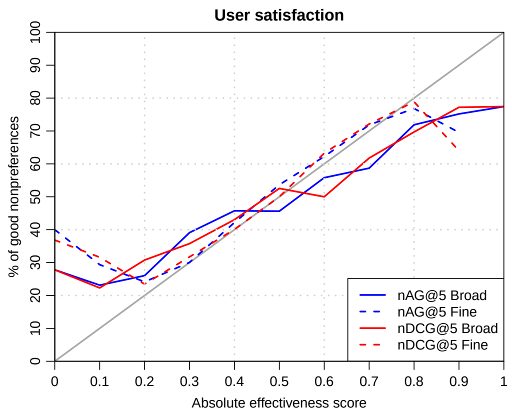


Figure 4. Ratio of good and bad nonpreferences in 884 examples, as a function of absolute system effectiveness, for $AG@5$ and $nDCG@5$ combined with the Broad and Fine judgments.

distributions of absolute effectiveness scores. As can be seen, a wide range of scores are covered, following a somewhat uniform distribution as well. The number of good and bad nonpreferences was almost the same too: 440 vs. 444.

Figure 4 shows the ratio of good nonpreferences observed in these 884 examples as a function of absolute effectiveness. As expected, there is a very tight positive correlation between effectiveness and user satisfaction. In fact, the relationship appears to be nearly linear. There is no appreciable difference between measures, but the Fine scale seems to adhere better to the diagonal than the Broad scale does. Note that the deviations from the trend with the Fine judgments ($\Delta < 0.2$ and $\Delta > 0.8$) are just an artifact of the very small number of observations in that range (see Section 3.1 and Figure 3).

Figure 4 shows a pretty straightforward mapping between $nAG@k$ and $nDCG@k$ scores and user satisfaction. However, the Broad scale seems to reveal a practical lower bound of 20% and an upper bound of 80% on user satisfaction. This could be merely due to noise in the crowdsourced data or a fault in the measures or scales. But given the symmetry, we believe these bounds are due to the natural diversity of users: some might consider something a very good result while others do not [4]. This means that even if a system obtains a $nAG@5$ score of 0, about 20% of the users will like the results (or dislike if $nAG@5 = 1$).

This is evidence of the room for improvement through personalization. Therefore, the AMS evaluations should include a user factor, possibly through user profiles, so that systems can attempt to reach 100% satisfaction on a per user basis. Otherwise, the final user satisfaction should not be expected to pass 80% for arbitrary users.

4.2 Evaluation of Two Systems

For 1,316 of the 2,200 examples (60%) we did receive a preference (i.e. subjects indicated that one system provided better results than the other one). Whether those user preferences were positive or negative (i.e. agreeing with the effectiveness difference or not), depends on the combination of measure and scale used. Figure 5 shows the ratio of preference signs across all 2,200 examples.

In terms of *positive* preferences (left plot), ideally we would want users to show a preference for the better system whenever we observe an effectiveness difference in the evaluation, regardless of how large this difference is. But there is a very tight positive correlation instead: the larger the difference in effectiveness, the more likely for users to prefer the supposedly better system. The relationship is again nearly linear, though this time we can observe a very clear difference between the Broad and Fine scales: for the same magnitude of the difference, the Fine judgments are always closer to the ideal 100% of positive user preferences. In fact, the Broad scale seems to indicate once again an upper bound of 80%. In addition, the plot shows that for users to prefer the supposedly better system more than the random 50% of the times, a difference of at least 0.3 in the Fine scale is needed, or 0.5 in the Broad scale. Note that the deviations from the trend with the Fine judgments ($\Delta > 0.8$) are also here just an artifact of the very small number of observations in that range.

As a consequence, there is a very clear negative correlation in terms of *nonpreferences* (middle plot): the larger the differences between systems, the more likely for users to prefer one of them. Again, the Fine scale seems to behave better than the Broad scale.

As the right plot shows, all four combinations of measure and similarity scale yield very similar ratios of *negative* preferences. There is a very slight negative correlation with difference in effectiveness, but in general about 5-10% of the user preferences disagree with the sign of the effectiveness difference. That is, about 5-10% of the times users prefer the supposedly worse system.

5. UNDERSTANDING EVALUATION RESULTS

The effectiveness of IR systems is assessed with different measures such as $nAG@k$ and $nDCG@k$. These measures are used to assign systems a score that represents how well they would satisfy users. For an arbitrary system A a measure M defines a distribution of effectiveness scores Y_A, describing the effectiveness of the system for an arbitrary query. The goal of evaluation experiments is usually finding the mean of that distribution: y_A.

Computing the parameter y_A allows researchers to assess how well the system performs and what is the expected user satisfaction according to the user model un-

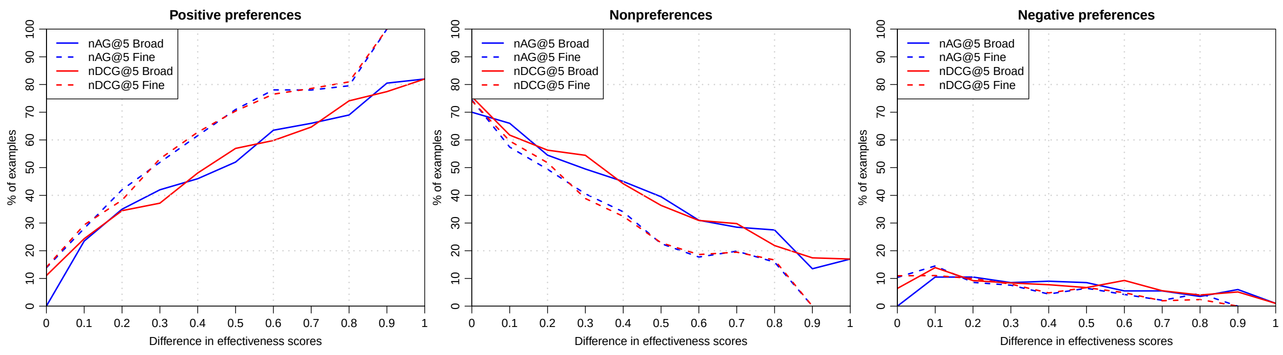


Figure 5. Ratio of positive preferences (left), nonpreferences (middle) and negative preferences (right) observed in the 2,200 examples, as a function of system effectiveness difference, for $nAG@5$ and $nDCG@5$ combined with the Broad and Fine scales.

derlying M. However computing this distribution would require running the system for the universe of all queries, which is clearly impossible. Instead, IR evaluation experiments are run with a sample of queries $\mathcal{Q}$, so they are used as estimators of the true y_{A}. The average effectiveness across queries, $\overline{y}_{\mathsf{A}}$, is used as the estimate $\hat{y}_{\mathsf{A}}$. Like any other estimate, $\hat{y}_{\mathsf{A}}$ bears some uncertainty, so statistical techniques such as confidence intervals should be employed to report the confidence on the estimation.

When comparing two systems, say A and B, one is usually interested in the distribution of the difference D_{AB}, representing the *paired* difference in effectiveness between A and B for an arbitrary query. Again, a comparative IR evaluation experiment only provides an estimate $\hat{d}$, whose sign indicates which system is expected to perform better.

5.1 Statistical Significance: p-values

Given that $\overline{d}$ is an estimate, the immediate question is: how confident can we be of this difference? The observed $\overline{d}$ could be just a random and rare observation due to the particular sample of queries used. Again, statistical techniques are needed to compute some sort of confidence on the difference. The most popular is hypothesis testing.

In a statistical hypothesis testing procedure, a *null* hypothesis H_0 is defined, such as $H_0 : d = 0$. The *alternative*, or research hypothesis, is then defined as the opposite: $H_1 : d \neq 0$. All hypothesis testing procedures are based on probability distributions, so there is always some degree of uncertainty when estimating parameters such as d. Thus, researchers may commit one of two errors: a Type I error if they conclude H_0 is not true when it actually is, or a Type II error if they conclude H_0 is true when it is not. The maximum probability of committing a Type I error is known as the *significance level*, usually $\alpha = 0.05$. The probability of committing a Type II error is denoted with the letter β, and $1-\beta$ is known as the *power* of the test: the probability of detecting a difference if there really is one.

The result of a hypothesis testing procedure is a probability called *p-value*. These are usually mistaken with the probability of H_0 being true [2, 11], but they are actually the probability of observing the difference $\overline{d}$ (or one larger) under the assumption that H_0 is true. That is, p-values are the probability of the data given the hypothesis, not the probability of the hypothesis given the data. If the reported p-value is smaller than the significance level α, we then reject the null hypothesis in favor of the alternative, and say that the difference is *statistically significant*. But it is important to note that the test does *not* tell anything about H_0 being true or false: that dichotomous interpretation is made by *us* based on the p-value and α, not by the test.

This is the ultimate goal of an IR evaluation: reaching significance. However, observing a statistically significant difference between two systems is usually misinterpreted as having high confidence that one system *really* is better than the other one because H_0 was rejected [2, 11]. In fact, all these null hypotheses are false by definition: any two different systems produce a distribution of differences with $d \neq 0$. What is important is the magnitude of d: differences of 0.0001, for instance, are probably irrelevant, but differences of 0.8 definitely are. However, a difference of just 0.0001 will always be statistically significant under certain experimental conditions, so focusing on statistical significance alone becomes, at some point, meaningless.

5.2 Practical Significance: effect-sizes

The most popular procedure to test such hypotheses about population means is the paired t-test. In IR evaluation, the hypotheses use to be $H_0 : d \leq 0$ and $H_1 : d > 0$. The test statistic is then computed as (note that in our case $d = 0$):

$$t = \frac{\overline{d} - d}{s_d / \sqrt{|\mathcal{Q}|}} \tag{1}$$

where s_d and $\overline{d}$ are the standard deviation and mean of the sample of D_{AB} computed with the set of queries $\mathcal{Q}$ in the test collection. Using the t-distribution's cumulative distribution function, the p-value is then calculated as the area that is to the right of t. If p-value $< \alpha$, *we* reject the null hypothesis and plainly conclude $d > 0$.

Examining Eqn. (1) we can see that the test is more likely to come up significant with larger observed differences $\overline{d}$ and smaller deviations s_d. But most important is to note that the power of the test is also directly proportional to the sample size $|\mathcal{Q}|$: the more queries we use to evaluate systems, the more likely to observe a significant difference. This shows that focusing on significance alone is eventually meaningless: all a researcher needs to do in order to obtain significance is evaluate with more queries.

Increasing the sample size (number of queries) increases the power of the test to detect ever smaller differences because the standard error on the mean, $s_d/\sqrt{|\mathcal{Q}|}$, decreases. Thus, observing a statistically significant difference does not mean that the systems really are different, in fact *they always are*. It just means that the observed difference and the sample size used were large enough to conclude *with confidence* that the true difference is larger than zero.

What really matters is how far apart from zero d is. This is the effect-size, which measures the *practical* significance of the result. As shown in Section 4.2, large differences in effectiveness scores (large effect-sizes) do predict more user satisfaction, but small differences do not really. However, with a sufficiently large number of queries we may be able to detect a statistically significant difference whose effect-size is extremely small, having no value for real users. In such a case we would have statistical significance, but no practical significance at all.

5.3 Reporting and Interpreting Results

We showed above that obtaining small p-values (statistical significance) should not be the sole focus of researchers when running evaluation experiments. The focus should really be on obtaining large effect-sizes (practical significance). The easiest way to report effect-sizes is just to report the effectiveness difference between systems or the absolute score of a single system. But these figures are just estimates of population means, and therefore subject to error. A better way to report effect-sizes is with confidence intervals, computed as $\overline{d} \pm t_{\alpha/2} \cdot s_d/\sqrt{|Q|}$. Confidence intervals for the absolute effectiveness of a single system are computed likewise, but using the $\overline{y}$ and s_y estimates.

Along with the results in Section 4, these confidence intervals can be used to interpret evaluation results from the ultimate perspective of user satisfaction. For instance, the HKHLL1 system in MIREX AMS 2011 obtained a $nAG@5$ score of 0.422 for the Fine judgments, with a 95% confidence interval ranging from 0.376 to 0.468. According to the results in Figure 4, this system is expected to satisfy an arbitrary user from about 35% to 45% of the times.

On the other hand, the difference between SSPK2 and DM2 was found to be statistically significant. The magnitude of the difference was just 0.082, with the 95% confidence interval ranging from 0.051 to 0.112. According to Figure 5 though, such difference is hardly ever noted by the users. Indeed, substituting in Eqn. (1) we find that any $\overline{d}$ larger than 0.031 would have been deemed as statistically significant for these two systems. This is an example of a statistically significant difference that makes no practical difference for arbitrary users.

In summary, we suggest to report not only the observed scores but also their confidence intervals, and the actual p-values rather than an indication of significance. For instance, a proper report for a single system would read as $nAG@5 = 0.584 \pm 0.023$. For the difference between two systems, we suggest $\Delta nAG@k = 0.037 \pm 0.031 (p = 0.02)$. By reporting the p-value we leave the interpretation of significance to the reader and his operational context: a large effect-size (e.g. $\overline{d} = 0.43$), even if not statistically significant (e.g. p-value $= 0.06$), is definitely worth implementing. After all, the levels $\alpha = 0.05$ and $\alpha = 0.01$, despite widely accepted, are completely arbitrary. People generally consider p-value $= 0.054$ as significant, while others request p-value < 0.005. It depends on the context of the reader and factors such as the cost of committing a Type I error or the cost of implementing one or another technique.

6. CONCLUSIONS

Reaching statistical significance in IR evaluation experiments is usually the most important goal for researchers. A difference between systems is usually regarded as important if significance is involved, when in reality all systems are different. With the development of ever larger test collections, statistical significance can easily be misunderstood, suggesting large differences between systems when they are actually very similar. To predict the real-world implications of these differences, researchers need to focus on effect-sizes as indicators of practical significance. That is, it does not matter whether there is a difference or not (in fact, there always is), what matters is how large it is. Final user satisfaction is only predicted with effect-sizes. Statistical significance serves just as a measure of confidence.

However, even when reporting on the magnitude of effectiveness differences, there is no established relationship with final user satisfaction. To fill this gap we carried out a user study with 100 real users in the context of the Audio Music Similarity and Retrieval task, where subjects indicated their preferences between different system outputs. Our results allow researchers to map observed absolute scores and relative effectiveness differences directly onto expected user satisfaction. In addition, they suggest room for improvement if considering personalization, as well as further work on the development of measures and evaluation criteria that more closely capture the user model underlying the task.

7. REFERENCES

[1] D. Harman. Information retrieval evaluation. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 2011.

[2] J. Ioannidis. Why most published research findings are false. *PLoS Medicine*, 2005.

[3] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Systems*, 2002.

[4] M. Jones, J. Downie, and A. Ehmann. Human similarity judgments: implications for the design of formal evaluations. In *ISMIR*, 2007.

[5] J. Le, A. Edmonds, V. Hester, and L. Biewald. Ensuring quality in crowdsourced search relevance evaluation: the effects of training question distribution. In *ACM SIGIR Workshop on Crowdsourcing for Search Evaluation*, 2010.

[6] J. Lee. Crowdsourcing music similarity judgments using Mechanical Turk. In *ISMIR*, 2010.

[7] W. Mason and D. Watts. Financial incentives and the performance of crowds. In *ACM SIGKDD Workshop on Human Computation*, 2009.

[8] M. Sanderson, M. Paramita, P. Clough, and E. Kanoulas. Do user preferences and evaluation measures line up? In *ACM SIGIR*, 2010.

[9] J. Urbano. Information retrieval meta-evaluation: challenges and opportunities in the music domain. In *ISMIR*, 2011.

[10] J. Urbano, J. Morato, M. Marrero, and D. Martín. Crowdsourcing preference judgments for evaluation of music similarity tasks. In *ACM SIGIR Workshop on Crowdsourcing for Search Evaluation*, 2010.

[11] S. Ziliak and D. McCloskey. *The cult of statistical significance: how the standard error costs us jobs, justice, and lives*. University of Michigan Press, 2008.

STATISTICAL CHARACTERISATION OF MELODIC PITCH CONTOURS AND ITS APPLICATION FOR MELODY EXTRACTION

Justin Salamon
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
justin.salamon@upf.edu

Geoffroy Peeters, Axel Röbel
Sound Analysis-Synthesis Team
IRCAM - CNRS STMS, 75004 Paris, France
{geoffroy.peeters,axel.roebel}@ircam.fr

ABSTRACT

In this paper we present a method for the statistical characterisation of melodic pitch contours, and apply it to automatic melody extraction from polyphonic music signals. Within the context of melody extraction, pitch contours represent time and frequency continuous sequences of pitch candidates out of which the melody must be selected. In previous studies we presented a melody extraction algorithm in which contour features are used in a heuristic manner to filter out non-melodic contours. In our current work, we present a method for the statistical modelling of these features, and propose an algorithm for melody extraction based on the obtained model. The algorithm exploits the learned model to compute a "melodiness" index for each pitch contour, which is then used to select the melody out of all pitch contours generated for an excerpt of polyphonic music. The proposed approach has the advantage that new contour features can be easily incorporated into the model without the need to manually devise rules to address each feature individually. The method is evaluated in the context of melody extraction and obtains promising results, performing comparably to a state-of-the-art heuristic-based algorithm.

1. INTRODUCTION

Melody extraction algorithms can be divided into several categories, based on their underlying approach. Some systems extract the melody by first separating it from the rest of the audio signal using source separation techniques [6, 11]. Purely data-driven approaches have also been proposed, such as [14] in which the entire short-time magnitude spectrum is used as training data for a support vector machine classifier. Still, the largest set of methods to date are those that can be referred to as *salience-based* algorithms, which derive an estimation of pitch salience over time and then apply tracking or transition rules to extract the melody line without separating it from the rest of the audio [4, 8, 12, 16, 18]. A review of salience-based systems can be found in [15].

For salience-based methods, one of the most important steps is the tracking and selection of pitch candidates. That is, given a set of pitch candidates at each frame, the system must decide which candidate belongs to the melody. This is also one of the steps that varies most between systems: in [8], tracking agents compete for candidates on a per frame basis using a set of heuristics, and the most salient agent at the end of the tracking is selected as the final melody. Tracking agents are also used in [5], where pitch candidates are first grouped into tone objects which are added to the agents using rules based on auditory streaming. In [16] Hidden Markov Models (HMM) are used to model the pitch evolution of single notes, and then the models are combined into a single HMM with inter-note transition probabilities learned from a training data-set.

In [18], we proposed a method for melody extraction based on pitch contour characteristics. In our approach, pitch candidates are first grouped over time into *pitch contours* – time and frequency continuous sequences of pitch candidates, whose length may vary from a single note in the shortest case to a short phrase in the longest. Given all the pitch contours generated from the audio signal of a polyphonic piece of music, we compute a set of contour characteristics, or features, related to contour salience, length, height and pitch evolution (namely pitch deviation and the presence of vibrato). These contour features are then used to devise filtering rules for filtering out non-melodic contours. Given the final set of contours after filtering, the melody is selected as the pitch candidate belonging to the most salient contour present in each frame. In the most recent Music Information Retrieval Evaluation eXchange (MIREX 2011) [3], the algorithm was shown to outperform all alternative approaches, obtaining the highest mean overall accuracy achieved by a melody extraction algorithm for the current MIREX data-sets [17].

Similar to other extraction algorithms such as [5, 8], one characteristic of our approach is that it heavily relies on heuristics for the candidate selection stage. Whilst this in itself is not a problem (some of the most successful algorithms also rely on heuristics [5]), it has the disadvantage that new heuristics must be devised whenever we want to incorporate new musical information into our algorithm (i.e. new contour features). This motivates us to explore the possibility of exploiting contour features in an automated manner, that is, creating a model based on contour features that can be easily updated whenever we want to incorporate new features.

In this paper, we present a method for the statistical characterisation of pitch contours using contour feature distributions. We do this by combining the distributions of different contour features into a single multivariate Gaussian distribution which embodies most of the features currently used by the algorithm. By learning separate feature distributions for melodic and non-melodic contours, we are able to create two different multivariate distributions, one for computing the likelihood that a contour is melodic, and the other for computing the likelihood that it's not melodic (i.e. accompaniment). The likelihoods are used to compute a single "melodiness" index, which is then used to select the final melody sequence. As can be inferred from the above description, the proposed method is flexible in that new features can be easily incorporated into the model without the need to manually devise rules to address them.

The structure of the remainder of the paper is as follows. In Section 2 we describe the proposed approach, including the creation of pitch contours, computation of contour features and their distributions, and the statistical modelling of these distributions. In Section 3, we describe the evaluation of the proposed approach, including evaluation material, measures and results. Finally, in Section 4 we conclude the paper with discussion of the results and some propositions for future work.

2. METHOD

In this section we describe the steps performed to obtain our contour feature model. These include the creation of pitch contours, computation of contour features and feature distributions, and finally the modelling of these distributions as a multivariate normal distribution.

2.1 Creating pitch contours

A summary of the contour creation process is provided here. For further details, the reader is referred to [18]. A block diagram of the process is provided in Figure 1.

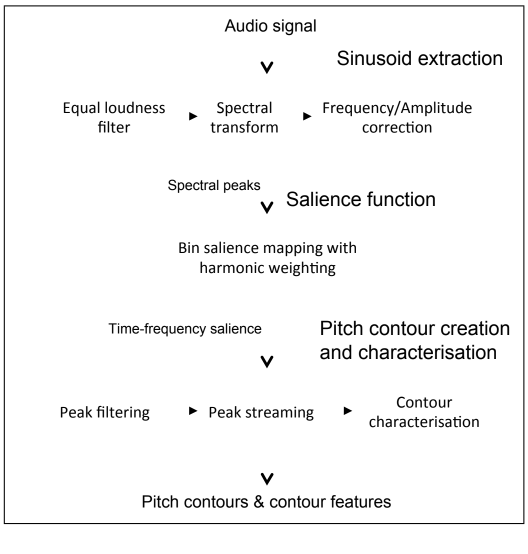


Figure 1. Block diagram of the steps involved in the creation of pitch contours.

In the first stage, sinusoids (spectral peaks) are extracted from the signal at each frame. We start by applying an equal loudness filter which attenuates frequencies where the melody is usually not present [19]. Next we compute the Short-Time Fourier Transform, and take the peaks of the spectrum at each frame. Peak frequencies and amplitudes are re-estimated by computing each peak's instantaneous frequency using the phase vocoder method [7]. In the next stage, the spectral peaks are used to create a salience function based on weighted harmonic summation [19]. The salience function is quantised into 600 bins covering a range of nearly five octaves from 55Hz to 1760Hz. The peaks of the salience function at each frame are considered as pitch candidates for the melody. In the next stage, the pitch candidates are grouped over time and frequency using rules based on auditory streaming [2] to create pitch contours. In Figure 2 we provide examples of contours generated from excerpts of different musical styles. Contours belonging to the melody are highlighted in bold.

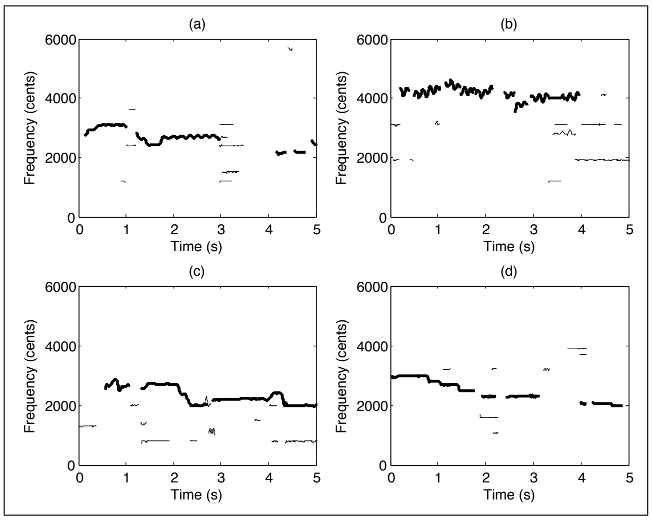


Figure 2. Pitch contours generated from excerpts of (a) vocal jazz, (b) opera, (c) pop and (d) instrumental jazz.

2.2 Contour features and distributions

Once the contours are created, we compute a set of contour characteristics, or features, for each contour. Similarly to some melody extraction systems, we define features based on contour pitch, length and salience [12]. However, by avoiding the quantisation of contours into notes we are able to extend this set by introducing features extracted from the pitch trajectory of the contour, namely its pitch deviation and the presence of vibrato. Every pitch contour is represented by two discrete series $c(n)$ and $s(n)$, $n = 1 \dots N$. The former contains the frequency values (in cents) of every pitch candidate in the contour, and the latter its corresponding salience value. Thus, for every pitch contour we compute the following characteristics:

- **Pitch mean** $C_{\bar{p}} = \frac{1}{N} \sum_{n=1}^{N} c(n)$.
- **Pitch deviation** $C_{\sigma_p} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (c(n) - C_{\bar{p}})^2}$.

- **Total salience** $C_{\Sigma s} = \sum_{n=1}^{N} s(n)$.
- **Mean salience** $C_{\bar{s}} = \frac{1}{N} C_{\Sigma s}$.
- **Salience deviation** $C_{\sigma_s} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(s(n) - C_{\bar{s}})^2}$.
- **Length** $C_l = N \cdot \frac{H}{f_S}$ (in seconds, where H and f_S are the hop size (128) and sampling frequency (44100) used by algorithm respectively).
- **Vibrato presence** C_v**:** whether the contour has vibrato or not (true/false). Vibrato is automatically detected by the system using a method based on [9].

In [18] these features were used to filter out non-melody pitch contours. To do this, we computed the distribution of each feature [1] for melody and non-melody contours using a representative data-set (c.f. Section 3.1), reproduced in Figure 3. Each plot includes the feature distribution for melody contours (solid red line) and non-melody contours (dashed blue line). In plots (c), (d) and (e) the feature values are normalised by the mean feature value for each excerpt. Observing these graphs we see how melodic contour characteristics differ from non-melodic contours: a mid-frequency pitch range, greater pitch variance, greater salience (both mean and total) and salience variance, and longer contours. These observations concur with voice leading rules derived from perceptual principles [10]. Note that in most (but not all) of the excerpts in this data-set the melody is sung by a human voice. Additionally, for vibrato presence we found that 95% of all contours in which vibrato was detected were melody contours.

In [18], these observations were exploited by devising a set of heuristic filtering rules to remove non-melodic contours. As mentioned in the introduction, whilst this approach was shown to be very successful for filtering non-melodic contours, in our current work we raise the question of whether the contour feature distributions can be exploited in a more general way using statistical modelling.

2.3 Statistical Modelling

Our goal is to define a statistical model that encompasses all of the contour feature distributions provided in Figure 3. To do so, we represent all feature distributions as two multivariate normal distributions, one for melodic contour features and one for non-melodic contour features. In [13] a multivariate Gaussian was shown to obtain comparable classification performance to GMMs when the amount of training data is relatively small.

As seen in the plots, though some distributions (in particular the distribution of pitch height for melodic contours) appear normal, this is not the case for all distributions. Thus, in the first step of the modelling we apply a power transform to obtain a normal-like distribution for each contour feature. Specifically, we apply the Box-Cox transform [1], which for a variable Y with data samples $y_i > 0$ is defined as:

[1] With the exception of vibrato presence which is a binary value (true/false).

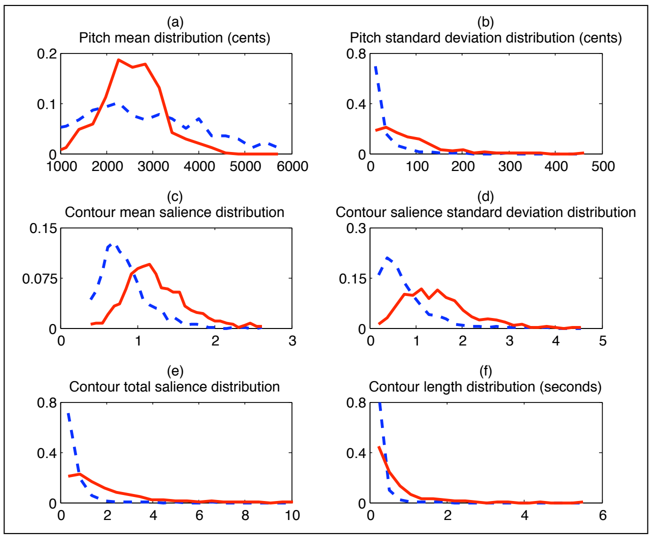


Figure 3. Pitch contour feature distributions (relative frequency vs. feature value): (a) Pitch mean, (b) Pitch std. dev., (c) Mean salience, (d) Salience std. dev., (e) Total salience, (f) Length. The red solid line represents the distribution of melody contour features, the blue dashed line represents the distribution of non-melody contour features.

$$y_i^{(\lambda)} = \begin{cases} \frac{(y_i^{\lambda}-1)}{\lambda}, & \text{if } \lambda \neq 0, \\ \log(y_i), & \text{if } \lambda = 0, \end{cases} \tag{1}$$

where the power parameter λ is selected such that it maximises the log-likelihood of λ given the transformed data, which is assumed to be normally distributed. An example of the distributions for the contour total salience feature $C_{\Sigma s}$ before an after transformation is provided in Figure 4. In plots (a) and (b) we show the feature distribution for melodic contours before and after transformation respectively, and in plots (c) and (d) we plot the corresponding distributions for non-melodic contours. In plots (b) and (d) we also display the normal distribution that best fits the transformed data.

The mean vectors μ and covariance matrices Σ (of size $N \times N$ where N is the number of features used) of the transformed distributions are obtained using the standard maximum likelihood estimators, allowing us to construct a multivariate normal distribution with parameters $\theta = (\mu, \Sigma)$ of the form:

$$f_\theta(\mathbf{x}) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right). \tag{2}$$

This procedure is repeated twice, once for the melodic contour feature distributions, and once for the non-melodic (i.e. background) contour feature distributions, resulting in two multivariate normal distributions which we denote $f_{\theta m}$ and $f_{\theta bg}$ respectively. Given the feature vector $\mathbf{x}$ of a pitch contour, we can now use $f_{\theta m}$ and $f_{\theta bg}$ to compute the likelihood of the contour being a melodic contour and the likelihood of it being a non-melodic contour (equations 3 and 4 respectively):

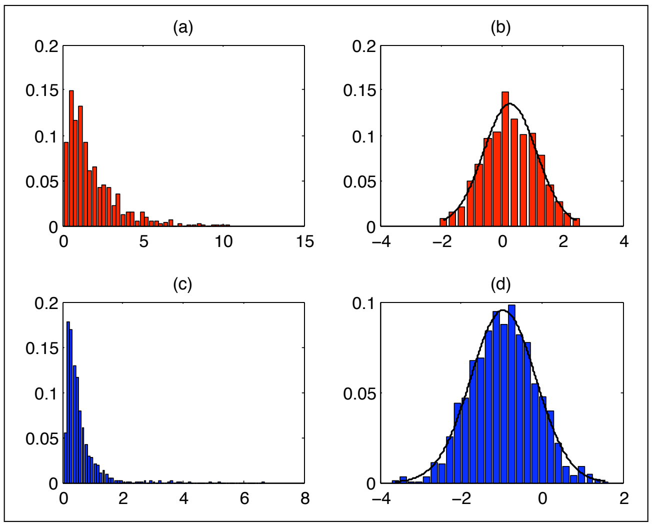


Figure 4. Contour total salience distributions. For melodic contours: (a) raw data, (b) after Box-Cox. For non-melodic contours: (c) raw data, (d) after Box-Cox.

$$\mathcal{L}(\theta_m|\mathbf{x}) = f_{\theta m}(\mathbf{x}) \tag{3}$$

$$\mathcal{L}(\theta_{bg}|\mathbf{x}) = f_{\theta bg}(\mathbf{x}) \tag{4}$$

Given the two likelihoods, we define the "melodiness" index $\mathcal{M}(\mathbf{x})$ of a pitch contour with feature vector $\mathbf{x}$ as the likelihood ratio of the melodic and non-melodic likelihoods:

$$\mathcal{M}(\mathbf{x}) = \frac{\mathcal{L}(\theta_m|\mathbf{x})}{\mathcal{L}(\theta_{bg}|\mathbf{x})} \tag{5}$$

3. EVALUATION

We evaluate the proposed contour characterisation approach in the context of melody extraction. To do so, we define a straight-forward rule for melody selection based on our proposed melodiness index $\mathcal{M}(\mathbf{x})$. Given the contours generated for a musical excerpt, at each frame we check to see which contours are present in the frame and select as melody the pitch candidate belonging to the contour whose features $\mathbf{x}$ result in the highest melodiness index $\mathcal{M}(\mathbf{x})$. The resulting melody sequence is evaluated using the standard measures employed for melody extraction evaluation. In the following sections we describe the music collection and measures used for evaluation, and compare the results obtained using the proposed method to those obtained by a state-of-the-art melody extraction algorithm.

3.1 Music Collection

The collection used for evaluation is comprised of musical excerpts with per-frame annotations of the melody F0 which are freely available for research purposes (cf. collection 3 in [18]). Our collection includes 65 audio excerpts from a variety of musical genres including rock, pop, R&B, jazz and opera singing. Excerpt durations range from 5 to 35 seconds. For each excerpt, the annotation is comprised of two columns, one containing the timestamp for the frame, and the other containing the F0 of the melody in that frame. If there is no melody present in the frame (i.e. the frame is 'unvoiced'), a value of 0Hz is placed in the annotation.

3.2 Evaluation Measures

The extracted melody sequence is evaluated using the standard measures employed for melody extraction evaluation in the MIREX campaign. The measures are designed to evaluate the performance of the algorithm in several aspects: voicing detection (determining when the melody is present and when it is not), pitch accuracy (estimating the correct F0 when the melody is present), and overall accuracy (the combination of voicing and pitch accuracy). It should be noted that our proposed approach, as presented above, does not include a method for voicing detection. That is, at each frame a non-zero frequency value is returned by choosing a pitch candidate from one of the contours present in the frame. The only exception are frames in which no contours are present, in which case the algorithm outputs 0Hz. For this reason, in the first part of the evaluation the proposed approach is evaluated only in terms of its pitch accuracy. In the second stage of the evaluation, we combine the proposed approach with the voicing detection method proposed in [18], and evaluate the results obtained using this combined approach. When voicing detection is included, the algorithm indicates whether a frame is voiced or unvoiced by returning either a positive or negative frequency value respectively (e.g. 300Hz or -300Hz). The negative values represent the pitch estimate of the algorithm for frames it has detected as unvoiced. When evaluating the algorithm's pitch accuracy the sign is ignored, meaning incorrect voicing detection will not affect the pitch (and chroma) accuracy. The overall accuracy (see below) serves as a global measure which considers both pitch and voicing detection accuracy. A summary of the evaluation measures, which are detailed in [15], is provided in Table 1.

Voicing Recall Rate: the proportion of frames labeled as voiced in the ground truth that are estimated as voiced by the algorithm.
Voicing False Alarm Rate: the proportion of unvoiced frames in the ground truth that are estimated as voiced by the algorithm.
Raw Pitch Accuracy: the proportion of voiced frames in the ground truth for which the F0 estimated by the algorithm is within $\pm\frac{1}{4}$ tone (50 cents) of the ground truth annotation.
Raw Chroma Accuracy: same as the raw pitch accuracy, except that both the estimated and ground truth F0 sequences are mapped into a single octave, in this way ignoring octave errors in the estimation.
Overall Accuracy: combines the performance of the pitch estimation and voicing detection to give an overall performance score. Defined as the proportion of frames (out of the entire piece) correctly estimated by the algorithm, where for non-voiced frames this means the algorithm labeled them as non-voiced, and for voiced frames it means the algorithm both labeled them as voiced and provided a correct F0 estimate for the melody (i.e. within $\pm\frac{1}{4}$ tone of the ground truth).

Table 1. Evaluation measures for melody extraction.

3.3 Results

To avoid any bias in the results, we separate the training and evaluation material by conducting a 3-fold cross validation, and report the results averaged across all folds. In Table 2 we provide the results obtained by our proposed approach (without any voicing detection method). For completeness we calculate all evaluation measures, though as explained above, since we do not attempt to perform any voicing detection, only the raw pitch and raw chroma measures (highlighted in bold) should be taken into consideration at this point. For comparison, we include the results obtained by the algorithm presented in [18] (which includes voicing detection), which obtained the highest mean overall accuracy results in MIREX 2011 (denoted SG) [17].

Alg.	Voicing Recall	Voicing False Alarm	**Raw Pitch**	**Raw Chroma**	Overall Accuracy
Prop.	0.95	0.60	**0.77**	**0.83**	0.65
SG	0.86	0.19	**0.81**	**0.83**	0.77

Table 2. Results obtained using the proposed method without voicing detection, compared to those obtained by SG.

We see that the proposed approach obtains the same chroma accuracy as the state-of-the-art algorithm. The lower raw pitch accuracy indicates that the proposed approach makes slightly more octave errors. Nonetheless, the results are definitely promising, and their comparability to SG suggests that the proposed approach is also comparable with other state-of-the-art melody extraction algorithms evaluated in MIREX [2].

In the second stage of the evaluation, we combine the proposed approach with the voicing detection method proposed in [18]. The approach is based on filtering out contours by setting a salience threshold determined from the distribution of contour mean salience $C_{\bar{s}}$ in a given excerpt. The reader is referred to the article cited above for further details. Thus, the combined approach consists of first filtering out non-voiced contours using the voicing filter, and then selecting the melody out of the remaining contours based on their melodiness index $\mathcal{M}(\mathbf{x})$ as before. The F0 estimate for non-voiced frames (recall that algorithms can return F0 estimates for non-voiced frames so that pitch and voicing accuracies can be evaluated independently) is produced by selecting the pitch candidate belonging to the non-voiced contour (i.e. a contour that was removed by the voicing filter) with the highest $\mathcal{M}(\mathbf{x})$ out of all non-voiced contours present in the frame. The results are presented in Table 3, once again alongside the results obtained by SG for comparison. This time we focus on the voicing evaluation measures and the overall accuracy.

As expected, by combining our proposed approach with a voicing detection method we are able to considerably reduce the voicing false-alarm rate (from 60% to 25%) whilst maintaining a relatively high voicing recall rate (87%). As a result, the overall accuracy of the proposed approach

Alg.	**Voicing Recall**	**Voicing False Alarm**	Raw Pitch	Raw Chroma	**Overall Accuracy**
Prop.	**0.87**	**0.25**	0.78	0.83	**0.74**
SG	**0.86**	**0.19**	0.81	0.83	**0.77**

Table 3. Results obtained using the proposed method with voicing detection, compared to those obtained by SG.

goes up from 65% without voicing detection to 74% with voicing detection. Though the same voicing detection approach is applied in both cases, we note the voicing false alarm is not the same. This is because some steps in SG, though not designed to address voicing detection, have been shown to have a positive effect on it [18]. Whilst the combined approach does not outperform SG (no other system has, to date), the results serve as a promising proof-of-concept, with an overall accuracy which is comparable to other state-of-the-art melody extraction algorithms.

As a final step, we inspect the values of our melodiness index $\mathcal{M}(x)$ for melody and non-melody contours. In Figure 5, we plot the values of $\mathcal{M}(x)$ for all pitch contours of all excerpts (on a log scale). For each excerpt we first normalise all $\mathcal{M}(x)$ values by the highest value in the excerpt, so that we can plot all values from all excerpts in a single graph. Values for melody contours are represented by a red circle, and values for non-melody contours by a blue x.

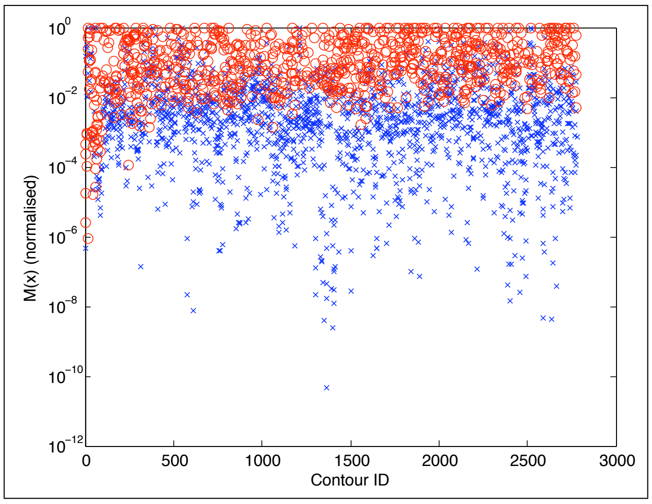


Figure 5. Normalised $\mathcal{M}(x)$ values for melody contours (red circle) and non-melody contours (blue x).

We see that the $\mathcal{M}(x)$ values for the two classes are fairly distinguishable, with the vast majority of melody contours having higher $\mathcal{M}(x)$ values than non-melody contours. This, apart from suggesting that $\mathcal{M}(x)$ is indeed a good indicator for melody contours, means we might be able to refine our melody selection algorithm by studying the distributions of $\mathcal{M}(x)$ for melody and non-melody contours. We intend to explore this possibility in future work.

4. CONCLUSION

In this paper we presented an approach for the statistical characterisation of melodic pitch contours. We explained how pitch contours can be generated from an audio excerpt

[2] Music Information Retrieval Evaluation eXchange [Online]: http://www.music-ir.org/mirex/wiki/Audio_Melody_Extraction (Apr. 12).

and how to calculate contour features. We then showed how these features can be used to build a model to describe melodic and non-melodic contours, leading to the computation of a melodiness index $\mathcal{M}(\mathbf{x})$. The proposed approach was evaluated in the context of melody extraction by using the melodiness index to select the melody out of the generated contours. The results of the evaluation showed that the approach achieves pitch and chroma accuracies comparable to a state-of-the-art melody extraction algorithm. By combining the proposed approach with a voicing detection method, we were able to obtained satisfying overall accuracy values as well.

When considering the caveats of the proposed approach compared to the state-of-the-art algorithm (SG), one clear difference is that whilst in SG temporal information is also taken into account, in the proposed approach the melody selection at each frame is performed using the melodiness index $\mathcal{M}(\mathbf{x})$ only, and no temporal continuity is taken into account. This means the pitch trajectory of the melody is allowed to contain large jumps which are not common in melodies, which tend to have a relatively smooth pitch trajectory in accordance with voice leading principles [10]. Thus, a possible direction for improving the performance of the proposed approach is to combine the melodiness index with some type of temporal evolution constraint. For instance, we could use the melodiness index in combination with one of the tracking techniques mentioned in the introduction of the paper, such as HMMs [16] or tracking agents [5, 8]. Another possibility for improvement is to consider more contour features. For instance, earlier in the paper it was explained that in 95% of the cases where the system detected vibrato in a contour, that contour belonged to the main melody. This information is not exploited in the current model (with the exception of the voicing detection method). An additional important research direction would be the gathering of more data to enable the use of more sophisticated statistical models (e.g. GMMs). Finally, another interesting research direction would be to learn genre specific feature distributions, and depending on the genre of the excerpt use a different model to compute $\mathcal{M}(\mathbf{x})$. This could be done by creating a two stage classification/melody extraction system, where the contour features could also be used for the classification stage as in [20].

5. ACKNOWLEDGMENTS

This work was supported by the Programa de Formación del Profesorado Universitario (FPU) of the Ministerio de Educación de España and the Quaero Program funded by Oseo French State agency for innovation.

6. REFERENCES

[1] G. E. P. Box and D. R. Cox. An analysis of transformations. *J. of the Royal Statistical Soc. Series B (Methodological)*, 26(2):211–252, 1964.

[2] A. Bregman. *Auditory scene analysis*. MIT Press, Cambridge, Massachussetts, 1990.

[3] J. S. Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[4] K. Dressler. Audio melody extraction for mirex 2009. In *5th Music Inform. Retrieval Evaluation eXchange (MIREX)*, 2009.

[5] K. Dressler. An auditory streamin approach for melody extraction from polyphonic music. In *12th Int. Soc. for Music Inform. Retrieval Conference*, pages 19–24, Miami, USA, Oct. 2011.

[6] J.-L. Durrieu. *Automatic Transcription and Separation of the Main Melody in Polyphonic Music Signals*. PhD thesis, Télécom ParisTech, 2010.

[7] J. L. Flanagan and R. M. Golden. Phase vocoder. *Bell Systems Technical J.*, 45:1493–1509, 1966.

[8] 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*, 43:311–329, 2004.

[9] P. Herrera and J. Bonada. Vibrato extraction and parameterization in the spectral modeling synthesis framework. In *Proc. Workshop on Digital Audio Effects (DAFx-98)*, pages 107–110, 1998.

[10] D. Huron. Tone and voice: A derivation of the rules of voice-leading from perceptual principles. *Music Perception*, 19(1):1–64, 2001.

[11] A. Ozerov, P. Philippe, F. Bimbot, and R. Gribonval. Adaptation of Bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *IEEE Trans. on Audio, Speech, and Language Process.*, 15(5):1564–1578, Jul. 2007.

[12] R. P. Paiva, T. Mendes, and A. Cardoso. Melody detection in polyphonic musical signals: Exploiting perceptual rules, note salience, and melodic smoothness. *Comput. Music J.*, 30:80–98, Dec. 2006.

[13] G. Peeters. Automatic classification of large musical instrument databases using hierarchical classifiers with inertia ratio maximization. In *Audio Engineering Society Convention 115*, Oct. 2003.

[14] G. Poliner and D. Ellis. A classification approach to melody transcription. In *Proc. 6th Int. Conf. on Music Inform. Retrieval*, pages 161–166, London, Sep. 2005.

[15] G. E. Poliner, D. P. W. Ellis, F. Ehmann, E. Gómez, S. Steich, and B. Ong. Melody transcription from music audio: Approaches and evaluation. *IEEE Trans. on Audio, Speech and Language Process.*, 15(4):1247–1256, 2007.

[16] M. Ryynänen and A. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Comput. Music J.*, 32(3):72–86, 2008.

[17] J. Salamon and E. Gómez. Melody extraction from polyphonic music: Mirex 2011. In *5th Music Inform. Retrieval Evaluation eXchange (MIREX)*, Miami, USA, Oct. 2011.

[18] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, Aug. 2012.

[19] J. Salamon, E. Gómez, and J. Bonada. Sinusoid extraction and salience function design for predominant melody estimation. In *Proc. 14th Int. Conf. on Digital Audio Effects (DAFx-11)*, pages 73–80, Paris, France, Sep. 2011.

[20] J. Salamon, B. Rocha, and E. Gómez. Musical genre classification using melody features extracted from polyphonic music signals. In *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, Kyoto, Japan, Mar. 2012.

DETECTING MELODIC MOTIFS FROM AUDIO FOR HINDUSTANI CLASSICAL MUSIC

Joe Cheri Ross*, Vinutha T. P.[Ŧ] and Preeti Rao[Ŧ]
Department of Computer Science and Engineering* Department of Electrical Engineering[Ŧ]
Indian Institute of Technology Bombay,
Mumbai 400076, India
joe@cse.iitb.ac.in*
{vinutha,prao}@ee.iitb.ac.in[Ŧ]

ABSTRACT

Melodic motifs form essential building blocks in Indian Classical music. The motifs, or key phrases, provide strong cues to the identity of the underlying *raga* in both Hindustani and Carnatic styles of Indian music. Thus the automatic detection of such recurring basic melodic shapes from audio is of relevance in music information retrieval. The extraction of melodic attributes from polyphonic audio and the variability inherent in the performance, which does not follow a predefined score, make the task particularly challenging. In this work, we consider the segmentation of selected melodic motifs from audio signals by computing similarity measures on time series of automatically detected pitch values. The methods are investigated in the context of detecting the signature phrase of Hindustani vocal music compositions (*bandish*) within and across performances.

1. INTRODUCTION

Hindustani classical music is primarily an oral tradition. While large archives of audio recordings are available, there are few written scores even for widely performed compositions. In such a scenario, retrieval of music based on any relevant high level music descriptors such as *raga* (melodic mode) or *bandish* (a *raga*-specific composition for vocal music) relies entirely on available textual metadata, if any. It is therefore very attractive to consider the automatic extraction of such metadata from audio recordings of concerts. Automatic detection of melodic motifs, for instance, can provide useful inputs to *raga* identification as well as identification of the *bandish* itself [1]. Also, within a concert audio recording, it would be interesting to detect the occurrence of the characteristic melodic phrases thus providing a rich transcription to the listener and the serious student of music. In this work, we consider the problem of detecting specific phrases from recorded performances given one instance of the phrase as template. We also attempt to understand the limitations of the approach in terms of the detection of phrases across performances and artistes.

There is no known previous work on the audio based detection of melodic phrases in Hindustani classical music. A considerable body of recent work, however, has addressed the discovery of melodic patterns from symbolic scores in Western folk music [2]. A continuous-time pitch contour is derived from the score for use in segment alignment and classification. Likewise, the limited reported work on audio signals is based on first obtaining note representations by monophonic pitch transcription [3]. In the case of Hindustani classical music, however, the available symbolic notation is inadequate to deal with tuning variations and complex ornamentation that are fundamentally linked to *raga* characteristics, calling for a different approach to data representation and pattern matching.

In the next section, we review the music background required to appreciate the problem, and outline the challenges. The database and evaluation methods are described next. A framework for the signal processing and pattern matching is proposed. The performance of the system is presented followed by a discussion of the results and prospects for future work.

2. MOTIFS IN HINDUSTANI MUSIC

Hindustani music, especially the modern *khyal* style, is a predominantly improvised music tradition operating within a well-defined *raga* (melodic) and *tala* (rhythmic) framework. Apart from the permitted scale intervals (*swaras*) that define a *raga*, it is its characteristic phrases that complete the grammar and give it a unique identity [4]. Most *ragas* can be adequately represented by up to 8 phrases which then become the essential building blocks of any melody. Thus the detection of recurring phrases can help to identify the *raga*. Indeed musical training involves learning to associate characteristic phrases, or motifs, with ragas. Melodic improvisation involves weaving together a unique melody bound by the chosen rhythmic cycle (*tala*) and consistent with the *raga* phraseology. Often the improvisation is anchored within a known composition, or song, known as *bandish* which provides a platform for exposing a specific *raga*. The *bandish* is identified by its lyrics (especially its title phrase) and melody corresponding to a specific *raga*, *tala* and *laya* (tempo). In the improvised section of the concert known as the *bol-alap*, the singer elaborates within each rhythmic cycle of the *tala* using the words of the *bandish* interspersed with solfege and held vowels, purposefully reaching the strongly accented first beat (the *sam*) of the next rhythmic cycle on a fixed syllable of the signature

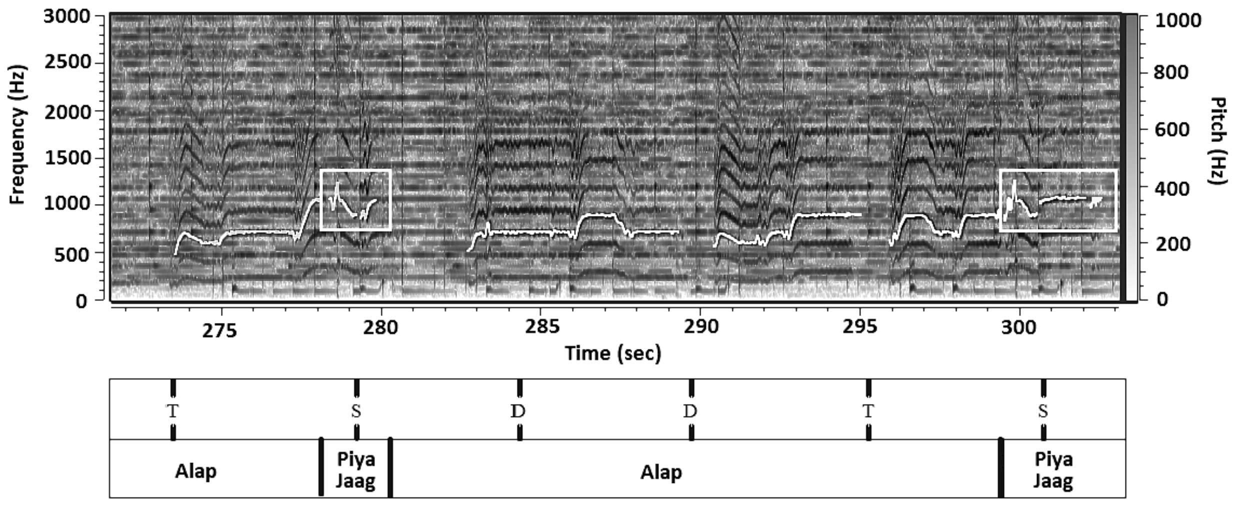


Figure 1. Top: spectrogram with superposed vocal pitch and *mukhda* in boxes; below: first beat of each subcycle (S= *sam*) with aligned lyrics in vocal regions.

phrase of the *bandish*. This recurring phrase, known as the *mukhda,* is the title phrase of the *bandish* and is defined by its text as well as its melodic shape. It acts like a refrain throughout the exposition, which can last several minutes, whereas the other lyrics of the *bandish* can undergo extensive variation in melodic shape in the course of improvisation.

While segmentation of characteristic phrases of the *raga* from a recorded performance is clearly an interesting task that falls within the scope of melodic motif detection, a trained musician is required to notate the recorded performances in order to generate the ground-truth needed for evaluation of any automatic system. On the other hand, the *mukhda* of the *bandish* is easy to segment manually due to the characteristic words of the lyrics and its specific location within the rhythmic cycle. Automatic detection of the *mukhda* can serve to identify the *bandis*h apart from making possible a partial transcription of the performance itself for the interested listener. Although the *mukhda* is detected easily by listening for the lyrics, automatic segmentation cannot rely on such cues due to the known difficulties of speech recognition from singing in polyphonic audio. We focus therefore on the melodic and rhythmic invariances to provide cues for the automatic detection of all occurrences of the *mukhda*, provided one reference instance, across the audio recordings of *bandish* of prominent artistes. Such work can also serve as the basis for more general melodic phrase detection contexts.

3. DATABASE AND EVALUATION METHODS

We selected 4 full-length CD-quality recorded concerts of well-known Hindustani *khyal* vocalists. In all cases, the accompanying instruments are the *tanpura* (drone), *harmonium* and *tabla*. The section of each concert corresponding to *bandish*-based improvisation (*bol-alap*) is extracted for this study. Table 1 shows the artiste names and *bandish* titles with other relevant details including CD cover metadata and the duration of the *bol-alap* section. All the performances use the popular *tintal* rhythm cycle with 16 beats divided equally over 4 sections. The beats are realized by the strokes of the *tabla* (percussion) with the first beat of each section considered to be stressed in 3 of the 4 sections. All the *mukhda* phrases, which may occur around any *sam* (first beat of the cycle) throughout the performance, are manually labeled. This serves as the ground truth ("positives") for the motif detection evaluation. The tempo indicated for each piece is an average, with slow fluctuations in cycle length observed throughout the recordings. Of the four recordings in Table 1, the first two correspond to the same *bandish* by different artistes. The last recording is by a female vocalist. It was observed that this recording with its slow tempo exhibits the largest variations in the duration of the *mukhda* even after accounting for local variations in cycle length.

Artiste	**Raga**	**Tala**	**Bandish**	**Tempo (bpm)**	**Dur. (min)**	**#Phrases**	
						Positive	Negative
Bhimsen Joshi (BJ)	Marwa	Tintal	Guru Bina Gyan	193	4.58	13	55
Ajoy Chakraborti(AC)	Marwa	Tintal	Guru Bina Gyan	205	9.08	33	295
Bhimsen Joshi (BJ)	Puriya	Tintal	Jana na na na	204	9.36	17	97
Kishori Amonkar (KA)	Deshkar	Tintal	Piya Jaag	43	22.3	44	176

Table 1. Description of database

For further processing, the audio is converted to 16 kHz mono at 16 bits/sample. Fig. 1 shows the spectrogram (of the 3 kHz frequency range) of a duration slightly greater than 1 full rhythmic cycle extracted from the *Piya Jaag* recording by Kishori Amonkar. Superimposed on the spectrogram is the detected pitch contour (as obtained by the method presented later in Sec. 4). Beneath the spectrogram is an annotation block depicting the aligned *tala* cycles. The first beat of the cycle is the *sam* (S) corresponds to the *dha* stroke of the tabla. In Fig. 1, the first beat of each sub-cycle is labeled (*dha* (D) or *tha* (T)). The penultimate sub-cycle before the S is the *khali*, as also evident from the absence of low frequency tabla partials in the spectrogram of this segment. The *mukhda* segments corresponding to the utterance "*Piya Jaag*" are enclosed in boxes. The *mukhda* segments are observed to be melodically similar and also similarly aligned within the *tala* cycles. Note that the song syllable that coincides with the *sam* (S) is sometimes left incomplete by the vocalist.

The proposed motif detection method is evaluated in the following experiments. 1) Within-concert detection accuracy where each manually labeled motif serves once as the reference template for all remaining motifs in the same artiste-*bandish* recording; 2) across-concerts detection accuracy where the reference template of a particular artiste is used to find the motifs of the same *bandish* by a different artiste.

4. AUTOMATIC MOTIF DETECTION

The pitch contour depicted in Fig. 1 can be viewed as a time series in which the desired phrase segments are embedded. As such, finding segments in the overall contour that are similar to a given phrase would involve matching the pitch at every time instant of the given phrase to the pitch at every other time instant throughout the time series [3]. It is of interest to explore methods to reduce the search complexity. In the present context, we can exploit the additional knowledge about the rhythmic relationship. As discussed in Sec. 3, the vocalist embeds the *mukhda* phrase in the metrical cycle (*tala*) so that a fixed syllable coincides with the *sam* instant. The metrical space of each cycle is occupied by improvisation culminating with the *mukhda*. Motivated by this, we approach the automatic detection of the *mukhda* phrase from the audio by first identifying a limited set of candidate phrases based on the detected rhythm cycle structure, and then computing a melodic similarity distance between the reference template and each of the candidates.

As in any classification task, it is necessary to design an appropriate data representation and a suitable similarity model for the matching. In this section, we describe the signal processing implementation of a pitch-based data representation and consider similarity models that are suited to the comparison of such time series. Finally, candidate segments with distances from the reference template lower than a threshold are the detected positives.

4.1 Signal Processing

4.1.1 Vocal Pitch Detection

In Hindustani classical vocal music, the accompanying instruments include the drone (*tanpura*), *tabla*, and often, the *harmonium* as well. The singing voice is usually dominant and the melody can be extracted from the detected pitch of the predominant source in the polyphonic mix. Melody detection involves identifying the vocal segments and tracking the pitch of the vocalist. The drone and *harmonium* are strongly pitched instruments. We therefore employ a predominant-F0 extraction algorithm designed for robustness in the presence of pitched accompaniment [4]. This method is based on the detection of spectral harmonics helping to identify multiple pitch candidates in each 10 ms interval of the audio. Next pitch saliency and continuity constraints are applied to estimate the predominant melodic pitch. The best of pitch detection methods achieve no more than 80% accuracy on polyphonic audio. An important factor limiting the accuracy is the fixed choice of analysis parameters, which ideally should be matched to the characteristics of the audio such as the pitch range of the singer and the rate of variation of pitch. In the regions of rapid pitch modulation, characteristic of Indian classical singing, shorter analysis windows serve better to estimate the vocal harmonic frequencies and amplitudes. Hence for better pitch detection accuracy, it is necessary to adapt the window length to the signal characteristics. This is achieved automatically by the maximization of a signal sparsity measure computed at each analysis instance (every 10 ms) for local pitch detection [6]. Finally, it is necessary to identify the vocal regions in the overall tracked pitch. This is achieved by using the peculiar characteristics of Hindustani music where the vocal segments are easily discriminated from the instrumental pitches due to the different temporal dynamics [7].

4.1.2 Motif Candidate Selection

Motivated by the characteristic of the *mukhda*, namely it's alignment with the *sam* stroke of the rhythm cycle, which the artiste pays great importance to achieve, the search algorithm starts by restricting candidate melodic segments to those that match rhythmically. This can achieved via the automatic detection of the beat instants in the audio. In the spectrogram of Fig. 1, the *tabla* strokes corresponding to the beats of the *tala* cycle are visible as vertical impulsive onsets. While the *sam* stroke itself is not particularly distinctive, the *dha* strokes (including the *sam*) can be detected as the highest onsets in the combined energies of two frequency bands: [5000, 8000] and [0, 1000] Hz. The former band is relatively free of interference from vocal partials while the latter band captures the low frequency partial of the *dha* stroke. The filtered output power is subjected to a first-order difference and then half-wave rectified. Spurious peaks are removed by a local threshold. The consistency of the spacing of detected onsets with the known average tempo is considered further to identify the largest peaks as the *dha* stroke onsets.

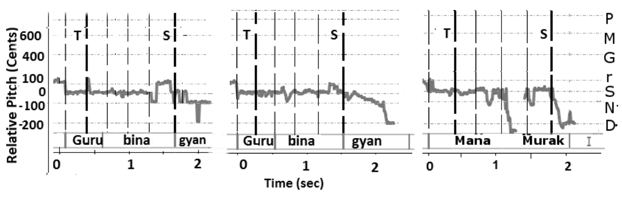


Figure 2. Two positive and one negative phrase of *Guru Bina Gyan*

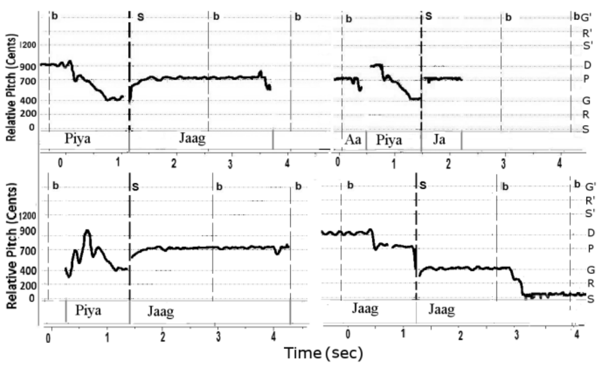


Figure 3. Three positive and one negative (bottom right) phrase of *Piya Jaag*

All audio segments whose alignment around a detected onset matches that of the *mukhda* are treated as potential candidates for motif detection. The extracted segment extends from the instant (*sam*-t1) to (*sam*+t2) where t1 and t2 are nominal values (number of beats in the 16-beat cycle) chosen based on the reference *mukhda* instance. Such a data representation is inherently robust to the slow tempo variations that occur during the concert.

The sequence of pitch values (in cents) obtained across the extracted candidate audio segment is a time series representation that is used further for similarity matching with a reference time series that is similarly obtained. Figures 2 and 3 depict the pitch contours of a few candidate segments showing examples of the melodic and timing variability across *mukhda* realization within concerts. We observe that there are prominent differences in the melodic pitch contour, both in terms of fine pitch variation as well as timing. The note (*swara*) sequence of the *Guru Bina Gyan* phrase is seen to be [Sa, Sa, Ni, Re, Ni, Dha]. However, since the word *Gyan* is often left unsung by the artiste, the *sam* itself serves as the right limit (i.e. t1=5, t2=0) of the *mukhda* in our task. The *swara* corresponding to *Piya Jaag* are [Da, Pa, Ga, Pa]. Here t1=1 and t2=2 were applied to delimit the *mukhda*. Any pitch gaps within the boundaries correspond to pauses. These are filled by linear interpolation or extrapolation of neighbouring pitch values before similarity matching.

4.2 Similarity Modeling

Due to the beat-based method of candidate extraction, the segments tend to be of different absolute durations depending on local tempo variations. Also, singing expressiveness manifests itself in timing changes that can affect the total duration of the sung phrase. The sequence of pitch values obtained at 10 ms intervals throughout the candidate audio segment can be viewed as a non-uniform length time-series representation. We explore two distinct similarity measures for non-uniform length temporal sequences.

Piecewise aggregate approximation has been used to obtain dimension-reduced uniform length time-series for motif discovery in bioinformatics [8]. We apply this method, called SAX, to convert a non-uniform length time series of pitch in cents, computed every 10 ms, to a uniform length, dimension-reduced sequence of symbols. The string length W is varied to determine the optimum dimension of the data representation. A given time-series is aggregated into W uniform length segments each represented by the averaged value of the segment. The real-valued pitch in cents is retained as such but we also consider quantizing pitch to the nearest semitone. Since the tonic frequency is singer dependent in Indian music, the semitone grid is anchored on the most prominent peak of an overall pitch histogram derived from the vocal pitch track across the test audio. Since our present work is confined to within-concert matching, a tonic detection error is inconsequential. Next, the Euclidean distance between the two W-length sequences, the reference and the candidate, is used as a similarity measure.

Another widely used method to compare real-valued time series related to each other through, possibly, nonlinear time-scaling, is the dynamic time-warping (DTW) distance measure [9]. The distance between the so aligned reference and candidate phrases is used as the similarity measure. Pathological warpings are avoided by incorporating the Sakoe-Chiba constraint on the width of a diagonal band in the DTW path matrix. The absolute difference in cents between pitch values is used as the local distance in the DTW path optimization. Any absolute difference within 25 cents (i.e. a quarter tone) is rounded down to 0 cents. This is found to help reduce the influence of practically imperceptible pitch differences on the warping path and therefore any unnecessary stretching of the path.

5. EXPERIMENTS AND DISCUSSION

Given the database described Table 1, we evaluate the different data representations and similarity measures on within-concert and across singer-concert motif detection tasks. Each candidate phrase extracted from the detected onsets as presented in Sec. 4 is labeled positive or negative depending on whether or not it is the actual motif (i.e. *mukhda* phrase). Table 1 shows the number of such phrase segments available for the evaluation of the motif detection methods. To maximize the use of the available annotated data, each labeled motif is considered as the reference once with all other motifs serving as positive tokens and the remaining candidates as negative tokens. Thus, the *Piya Jaag* motif detection task can be evaluated on 44x43 = 1892 positive pairs and 44x176 = 7744 negative pairs (i.e. each positive with all negatives). Table 2 summarizes the experiments. The Experiment A considers motif detection from within the *Guru Bina* recording of Bhimsen Joshi given a reference template from the same recording. Similarly, the Experiments B, C and D consider the within-concert detection as specified in Table 2. The Experiment E uses the positive tokens of *Guru Bina* by BJ to detect the *mukhda* in the same *bandish* concert by a different vocalist, AC. As it turns out, the two male singers are tuned to the same tonic. In each experiment, the rate of false alarms for a given hit rate (correct detections) is computed for each combination of similarity model and data representation. The similarity measures include SAX and DTW. The data representations chosen for the study are either the continuous pitch values (i.e. 1200 cents per octave) indicated by "q1200", or the quantized versions (12 semitones per octave on an equitempered scale) indicated by "q12".

Fig. 4 shows an example of the distribution of distances for positive-positive pairs and positive-negative pairs. The recording is *Piya Jaag* (Experiment D) evaluated with DTW-q1200. We observe that the distances between the positive phrases cluster closely relative to the distances between the positive-negative phrase pairs. There is a limited overlap between the two distributions. That the spread of the negative distances is relatively wide indicates the robustness of the distance measure in terms of its discrimination of melodic shapes. We also note the presence of a small isolated cluster of positive distances. A closer examination revealed that this stemmed from the wide timing variability across *Piya Jaag* phrases with its particularly slow tempo. Thus there were at least two distinct behaviours within the set of positive phrases. The inter-phrase distances between the longer duration phrases tended to be lower than the distances involving shorter duration phrases. Fig. 5 shows the ROC (hit rate versus false alarm rate) derived from the distributions of Fig. 4 by varying the decision threshold. We observe two bends in the curves, consistent with the bimodal shape of the pdf.

Table 3 summarizes the classification results across the experiments in terms of false alarm rate (FA) for a fixed HR chosen near the knee of the ROCs of the corresponding data. Given that the extracted candidate phrases have durations varying in the range of 2-4 sec (200-400 length string), we vary the SAX string length around W=50 (corresponding to the aggregation of 4-8 samples). Preliminary experiments revealed that W substantially lower than this (i.e. more averaging) led to worsened performance. We note that the performance of SAX improves with pitch quantization at a fixed string length of 50. Increasing or decreasing the string length around this does not improve performance on the whole. The DTW system performs substantially better than SAX in terms of reducing the FA at fixed hit rate. As in the case of SAX, pitch quantization helps to improve performance further in some cases. That DTW does relatively better indicates that non-uniform time warping is essential to achieve the needed alignment between phrases before distance computation. This is consistent with what is known about the tradition where the essential melodic sequence of the *mukhda* phrase is strongly adhered to by the singer while metrical alignment is focused only on getting to the specific syllable onset (e.g. *Gyan* in Fig. 2 and *Jaag* in Fig. 3) on the first beat of the cycle (the *sam*).

Expt	Bandish	Singer	#Phrases	
			POS	NEG
A	Guru Bina	BJ	156	715
B	Guru Bina	AC	1056	9735
C	Jana na na na	BJ	272	1649
D	Piya Jaag	KA	1892	7744
E	Guru Bina	BJ vs AC	429	3835

Table 2. Description of experiments with number of positive and negative phrase candidates available in each

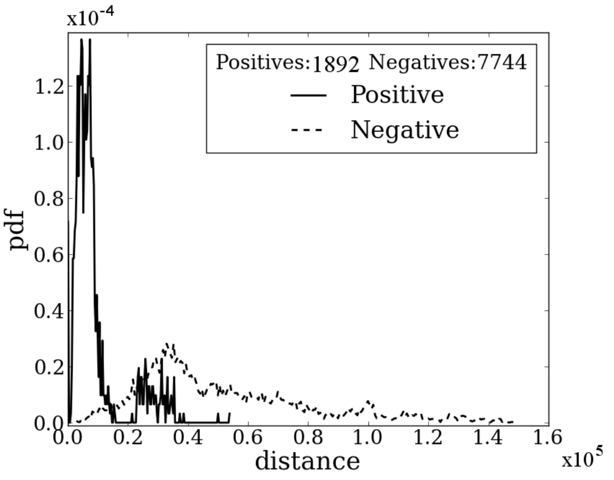


Figure 4. DTW distances distribution for *Piya Jaag* recording

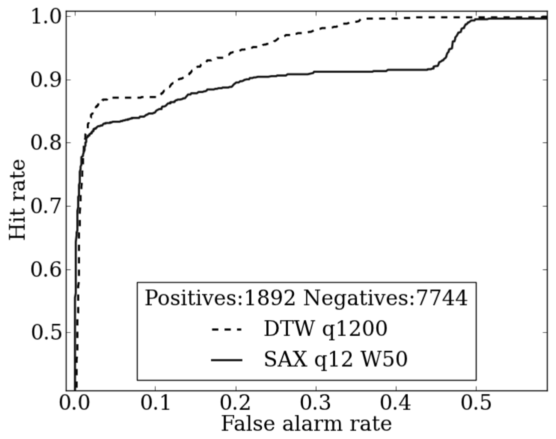


Figure 5. ROC curves for *Piya Jaag* distribution

Further, comparing the Experiments E and B as a case of between-concert to within-concert performance of the motif detection methods, we see that the FA is somewhat higher in Experiment E which involves a reference motif

Method	**Experiment A**		**Experiment B**		**Experiment C**		**Experiment D**		**Experiment E**	
	HR	FA	HR	FA	HR	FA	HR	FA	HR	FA
SAX-q1200 -W50	1	.096	.94	.019	.86	.239	.87	.130	.94	.035
SAX-q12-W40	1	.084	.94	.016	.86	.231	.87	.135	.94	.024
SAX-q12-W50	1	.071	.94	.015	.86	.216	.87	.124	.94	.029
SAX-q12-W60	1	.091	.94	.014	.86	.210	.87	.133	.94	.023
DTW-q1200	1	.044	.94	.007	.86	.044	.87	.032	.94	.015
DTW-q12	1	.053	.94	.008	.86	.042	.87	.025	.94	.014

Table 3. Performance of SAX and DTW motif detection under different configurations. WX = SAX string dimension is X; qY= quantized pitch levels per octave; HR = hit rate; FA = number of false alarms

from the concert of the same *bandish* by a different artiste. This is consistent with the anticipated higher variability in motif contour across artistes.

6. CONCLUSION

Similarity measures traditionally used in time-series matching have been shown to perform well in the context of melodic motif detection in the improvised *bandish* of Hindustani vocal concert recordings. The processing of the polyphonic audio signals needed to achieve a suitable data representation was presented. Musical knowledge related to the metrical relation between the *mukhda* motif and the underlying rhythmic structure was exploited to achieve a reduced search space, using available similarity measures, and possibly more robust detection. While the *mukhda* context considered in this work is relevant in both Hindustani and Carnatic vocal music (in the *bol-alap* and *niraval* respectively), the detection of other characteristic *raga* phrases would be a logical extension. It is not clear whether rhythmic cues would help in this more general melodic segmentation. Further extension to unsupervised clustering of phrases in a concert recording can contribute to higher-level classification tasks such as *raga* recognition as well to further research in audio transcription for such musical traditions.

7. ACKNOWLEDGEMENT

This work received partial funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 267583 (CompMusic).

8. REFERENCES

[1] J. Chakravorty, B. Mukherjee and A. K. Datta: "Some Studies in Machine Recognition of Ragas in Indian Classical Music," *Journal of the Acoust. Soc. India*, Vol. 17, No.3&4, 1989.

[2] Z. Juhasz: "Analysis Of Melody Roots In Hungarian Folk Music Using Self-Organizing Maps With Adaptively Weighted Dynamic Time Warping," *Journal Applied Artificial Intelligence*, Vol.21, No.1, 2007.

[3] R. B. Dannenberg and N. Hu: "Pattern Discovery Techniques for Music Audio," *Journal of New Music Research*, Vol. 32, No.2, 2002.

[4] S. Rao, W. van der Meer and J. Harvey: "The Raga Guide: A Survey of 74 Hindustani Ragas," Nimbus Records with the Rotterdam Conservatory of Music, 1999.

[5] V. Rao and P. Rao: "Vocal Melody Extraction in the Presence of Pitched Accompaniment in Polyphonic Music," *IEEE Trans. Audio Speech and Language Processing*, Vol. 18, No.8, 2010.

[6] V. Rao, P. Gaddipati and P. Rao: "Signal-driven Window-length Adaptation for Sinusoid Detection in Polyphonic Music," *IEEE Trans. Audio, Speech, and Language Processing,* Vol. 20, No.1, 2012.

[7] V. Rao, C. Gupta and P. Rao: "Context-aware Features for Singing Voice Detection in Polyphonic Music," *Proc. of Adaptive Multimedia Retrieval*, 2011.

[8] J. Lin, E. Keogh, S. Lonardi and B. Chiu: "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms," In *Proc. of the Eighth ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2003.

[9] D. Berndt and J. Clifford: "Using Dynamic Time Warping to Find Patterns in Time Series," *AAAI-94 Workshop on Knowledge Discovery in Databases,* 1994.

CHARACTERIZATION OF INTONATION IN CARNATIC MUSIC BY PARAMETRIZING PITCH HISTOGRAMS

Gopala K. Koduri[1], Joan Serrà[2], and Xavier Serra[1]
[1] Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain.
[2] Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Barcelona, Spain.
gopala.koduri@upf.edu, jserra@iiia.csic.es, xavier.serra@upf.edu

ABSTRACT

Intonation is an important concept in Carnatic music that is characteristic of a raaga, and intrinsic to the musical expression of a performer. In this paper we approach the description of intonation from a computational perspective, obtaining a compact representation of the pitch track of a recording. First, we extract pitch contours from automatically selected voice segments. Then, we obtain a a pitch histogram of its full pitch-range, normalized by the tonic frequency, from which each prominent peak is automatically labelled and parametrized. We validate such parametrization by considering an explorative classification task: three raagas are disambiguated using the characterization of a single peak (a task that would seriously challenge a more naïve parametrization). Results show consistent improvements for this particular task. Furthermore, we perform a qualitative assessment on a larger collection of raagas, showing the discriminative power of the entire representation. The proposed generic parametrization of the intonation histogram should be useful for musically relevant tasks such as performer and instrument characterization.

1. INTRODUCTION

Carnatic music is the south Indian art music tradition and Raaga is the melodic framework on which Indian art music thrives. The intonation of a single swara[1] can be different due to the melodic context established by different raagas. Therefore, to understand and model Carnatic music computationally, intonation analysis becomes a fundamental step.

We define intonation as the pitches used by a performer in a given musical piece. From this definition our approach will consider a performance of a piece as our intonation unit. In Carnatic music practice, it is known that the intonation of a given swara can vary significantly depending on the artist and the raaga [8, 15]. The study of intonation differs from that of tuning in its fundamental emphasis. When we talk about tuning we refer to the discrete frequencies with which we tune an instrument, thus it is more of a theoretical concept than the one of intonation, with which we focus in the pitches used during a performance. The two concepts are basically the same when we study instruments that can only produce a fixed set of discrete frequencies, like the piano.

Given than in Indian music there is basically no instrument with fixed frequencies (the harmonium is an important exception), in practice tuning and intonation can be considered the same. Here we will maintain the terms, tuning or intonation, used by the different studies.

Krishnaswamy [7] discusses various tuning studies in the context of Carnatic music, proposing a hybrid tuning scheme based simple frequency ratios plus various tuning systems, specially equal temperament. His work also points out the lack of empirical evidence thus far. Recently, Serrà et al. [12] have shown important quantitative differences between the tuning systems in modern Carnatic and Hindustani[2] musics. In particular, they show that Carnatic music follows a tuning system which is very close to just-intonation, whereas Hindustani music follows a tuning system which tends to be more equi-tempered. Although there are several studies on intonation and tuning in Indian music, the emphasis so far has been in the interpretation of ancient texts rather than on analysing real musical practise(see [7, 8, 12] and references therein).

In a study that was conducted with Hindustani music performances [8], pitch consistency is shown to be highly dependent on the nature of gamaka[3] usage. The swaras sung with gamakas were often found to have a greater variance within and across the performances, and across different performers. Furthermore, the less dissonant swaras were also found to have greater variance. However, it was noted that across the performances of the same raaga by a given performer, this variance in intonation was minor. The same work concludes that the swaras used in the analyzed performances do not strictly adhere to either just-intonation or equal-tempered tuning systems. Belle *et al* [1] use the intonation information of swaras to classify Hindustani raagas. Another recent experiment conducted with

[1] A swara-sthana is a frequency region which indicates the note and its allowed intonation in different melodic contexts.

[2] The north Indian art music tradition.

[3] Gamakas are a class of short melodic movements sung around and between swaras.

Carnatic music performances draws similar conclusions about the variance in intonation [15]. However, the methodology employed in these experiments cannot easily be scaled to a larger set of recordings due to the human involvement at several phases of the study, primarily in cleaning the data and the pitch tracks, and also in interpreting of the observations made.

Approaches to tuning analysis of real musical practise usually follow a so-called 'stable region' approach, in which only stable frequency regions are considered for the analysis (cf. [12]). However, it is known [14] that the most portion of the performance in Carnatic music is gamaka-embellished. Since gamakas are characteristic to a given raaga, such an approach is not suitable to understand the crucial information provided by them. So far, the tuning analysis was approached to explain the interval positions of Carnatic music with one of the known tuning methods like just-intonation or equal-temperament. But considering that these intervals are prone to be influenced by factors like raaga, performer [8] and instrument [7], computational analysis of swara intonation for different raagas, artists and instruments has much more relevance to the Carnatic music tradition.

2. HISTOGRAM PEAK PARAMETRIZATION

In this contribution we propose a methodology based on histogram peak parametrization that helps to describe the intonation of a given recording by characterizing the distribution of pitch values around each swara. From the observations made by Krishnaswamy [7] and Subramanian [14], it is apparent that steady swaras only tell us part of the story that goes with a given Carnatic music performance. The gamaka-embellished swaras pose a difficult challenge for automatic swara identification. Therefore, alternative means of deriving meaningful information about the intonation of swaras becomes important. The gamakas and the role of a swara are prone to influence the aggregate distribution of a swara. We believe that this information can be derived by parametrizing the distribution around each swara.

Our intonation description method can be broadly divided into six steps. In the first step, the prominently vocal segments of each performance are extracted using a trained support vector machine (SVM) model. In the second step, the pitch corresponding to the voice is extracted using multipitch analysis. In the third step, using all the performances of each raaga, a pitch histogram for every raaga is computed and its prominent peaks detected (we will refer to them as reference peaks). In the fourth step, we compute the pitch histogram for each single performance, detecting the relevant peaks and valleys using information from the overall histogram of the corresponding raaga. In the fifth step, each peak is characterized by using the valley points and an empirical threshold. Finally, in the sixth step, the parameters that characterize each of the distributions are extracted.

2.1 Segmentation

Cleaning the data is a crucial pre-processing step for our experiments. All the Carnatic vocal performances are accompanied by a violin and one or more percussion instruments. We just use the sections in which the voice is alone or very prominent. In order to do this automatically, we train an SVM model [5] on 300 minutes of audio data, equally split between vocal, violin and percussion sections of 10 seconds each. The features extracted from the audio and used in the classification task are [4]: Mel-frequency cepstral coefficients, pitch confidence, spectral flatness, spectral flux, spectral rms, spectral rolloff, spectral strong-peak, zero crossing rate and tristimulus. This method scores an accuracy of 96% in a 10-fold cross validation test.

2.2 F0 Analysis

With the segmentation module in place, we minimize to a large extent pitch errors due to the interfering accompanying instruments. However, there is a significant number of the obtained voice segments in which the violinist fills short pauses or in which the violin is present in the background, mimicing the vocalist very closely with a small time lag. This is one of the main problems we encountered when using pitch tracking algorithms like YIN [3], since the violin was also being tracked in quite a number of portions. The solution has been to extract the predominant melody [10] using a multi-pitch analysis approach. Then, given that the pitch accuracy of YIN is better, we compare the pitch obtained from the multi-pitch analysis with YIN at each time frame, and we only keep the pitch from those frames where both methods agree within a threshold. Though it is a computationally intensive step, this helps in obtaining clean pitch tracks, free of f0-estimation and octave errors. The frequencies are then converted to cents and normalized with the tonic frequency obtained using [11]. The octave information is retained.

2.3 Histogram Computation

As Bozkurt et al. [2] point out, there is a trade-off in choosing the bin resolution of a pitch histogram. A good bin resolution keeps the precision high, but significantly affects the peak detection accuracy. However, unlike Turkish-maqam music where the octave is divided into 53 Holdrian commas, Carnatic music uses roughly 12 swaras [13]. Hence, in this context, choosing a finer bin width is not as much a problem as it is in Turkish-maqam music. In order to retain the preciseness in estimating the parameters for such distribution, we keep the bin resolution at one cent. We then compute the histogram H by placing the pitch values into their corresponding bins:

$$H_k = \sum_{n=1}^{N} m_k, \tag{1}$$

where H_k is the k-th bin count, N is the number of pitch values, $m_k = 1$ if $c_k \leq P(n) \leq c_{k+1}$ and $m_k = 0$ otherwise, P is the array of pitch values and (c_k, c_{k+1}) are the bounds on k-th bin.

Features/Classifier	Naive Bayes	1-Nearest Neigh.	SVM	Logistic Regression	Random Forest
Mean and Height	63.43%	56.67%	61.81%	56.33%	64.62%
All parameters combined	63.76%	68.90%	65.19%	68.86%	70.71%

Table 1. Results of an exploration raaga classification test with 42 recordings in 3 raagas using different classifiers. The random baseline accuracy in this case is 28.57%.

Features/Classifier	Naive Bayes	1-Nearest Neigh.	SVM	Logistic Regression	Random Forest
Mean and Height	39.6%	39.85%	41.25%	43.65%	48.85%
All parameters combined	58.05%	67.6%	74.25%	77.45%	74.45%

Table 2. Results of an exploration raaga classification test with 26 recordings in 2 raagas using different classifiers. The random baseline accuracy is 20% in this case.

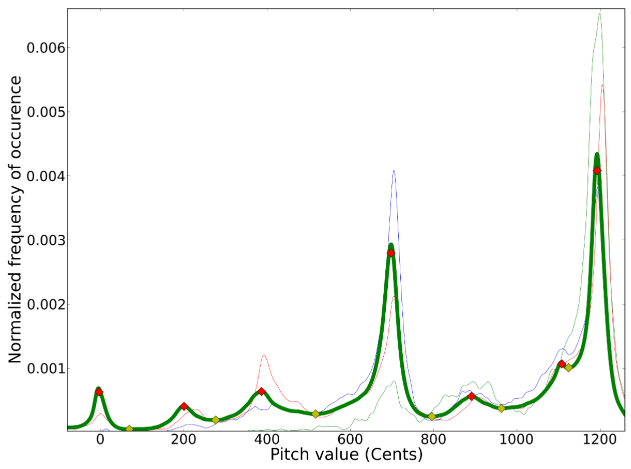


Figure 1. Histograms corresponding to the recordings of Kalyani raaga shown in thin lines of different colors, together with the average histogram labelled with peaks and valleys, shown in thick green line. Only the middle octave, which contains the most information, is shown.

In the histograms of a few performances, we observe that a number of pitch distributions for specific swaras are very narrow. However we can observe that the distributions still play a role in characterizing the performance. To validate this observation, an average histogram is computed for each raaga with all the performances in the raaga. This histogram has a clearer distribution for each swara in the raaga and it serves as a reliable reference to verify the peaks identified in individual performances (Fig. 1).

2.4 Peak Detection and parametrization

A given histogram is convolved with a Gaussian kernel using a standard deviation of five cents. This step is necessary for the peak detection algorithm to avoid identifying the spurious peaks. The peaks are identified using a depth parameter (D_p) and with an empirically set lookahead parameter (L_p). A local maxima is labelled as a peak only if it has valleys deeper than D_p on either side, and is also the maxima at least within L_p values ahead. In the case of H_{avg}, D_p and L_p are set high (in our experiment, they correspond to $2.5 \cdot 10^{-5}$ and 20 respectively), which result in fewer, but reliable peaks. In addition, for each peak in H_{avg}, an upper and lower octave peak is added if there does not already exist a peak in a given proximity. We call these *extended reference peaks*. To compute the histogram of a given performance, D_p and L_p are set to lower values (in our experiment, they correspond to $2 \cdot 10^{-5}$ and 15 respectively), which result in more peaks which include several unwanted ones. However, only those peaks which have a corresponding match in *extended reference peaks* of H_{avg} (within a given proximity) are retained. This step not only helps to identify all the possible peaks for a given performance, but also compensates the choice of higher bin resolution, which otherwise generally results in some unwanted peaks.

In order to parametrize a given peak in the performance, it needs to be a bounded distribution. Generally, we observe that two adjacent peaks are at least 80 cents apart. The valley point between the peaks becomes a reasonable bound if the next peak is close by. But in cases where they are not, we have used a 50 cent bound to limit the distribution. The peak is then characterized by five parameters: peak location, mean, variance, skew and kurtosis.

3. RESULTS & DISCUSSION

One of the crucial factors that influence intonation of a given swara is raaga. This can be attributed to the factors like the role the swara plays in the raaga, the gamakas used with it, and the characteristics of neighbouring swaras. For a given swara, all of them change with the raaga [13]. This affects the distribution of pitches around it. Therefore, we choose to evaluate our approach by using intonation of swaras to characterize raagas. Hence, we focused on a data set which is representative enough of the variations allowed on swaras: 170 performances in 16 raagas each with at least 5 recordings per raaga are selected. These performances feature 35 vocal artists in total.

We evaluate our approach using three self-contained tasks. The first one is an explorative raaga disambiguation task in which the proposed parameters of the peak are shown to consistently increase the accuracy of the system. The second task is a qualitative study showing the use-

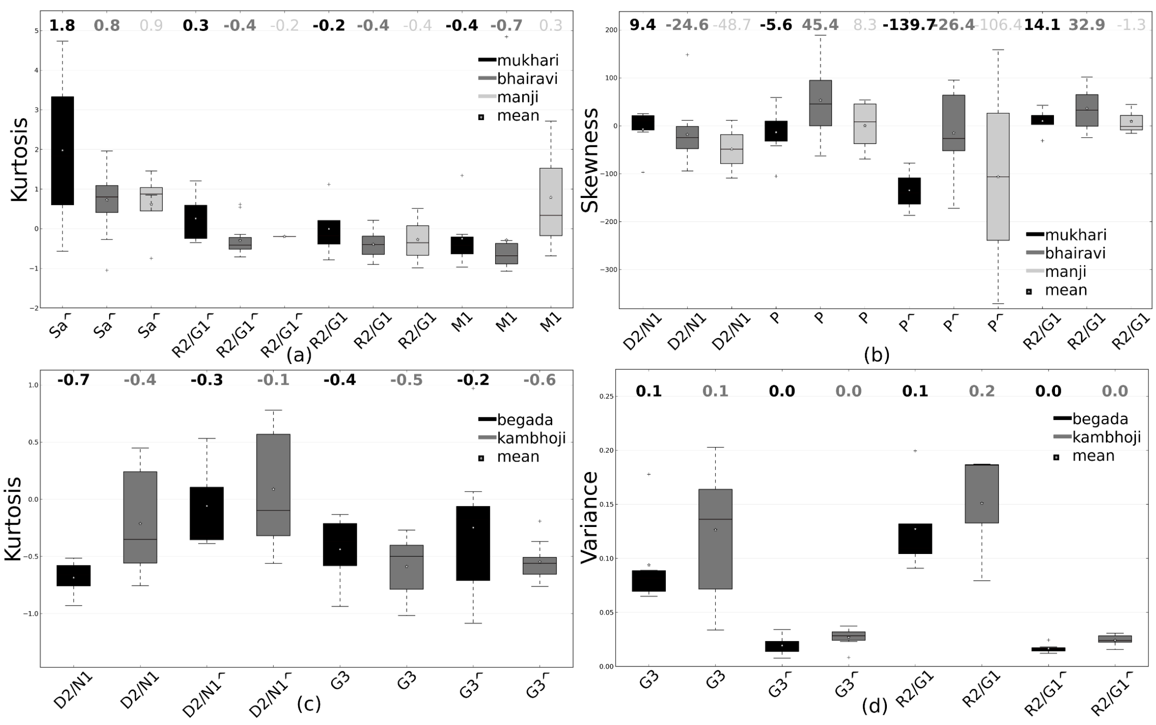


Figure 2. (a). Kurtosis values for swaras of Bhairavi, Mukhari and Manji raagas. (b). Pearson's first skewness coefficient values for swaras of Bhairavi, Mukhari and Manji raagas. (c). Kurtosis values for swaras of Begada and Kambhoji raagas. (d). Variance values for swaras of Begada and Kambhoji raagas. The x-axis corresponds to swara names in all the subplots.

fulness of peak parameters in discriminating raagas which share the same set of swaras. In the third task, the peak positions are used in deriving a general template for preferred mean values of swaras. Further, this general template is juxtaposed against different raagas showing notable deviations.

3.1 Raaga classification task

Previous raaga classification techniques employ only two parameters extracted by histogram analysis: peak position and height [6]. However, in using just these two parameters for classifying raagas which share the same set of swaras, there is a high chance of error. In order to assert the usefulness of our approach to describe intonation, we take 42 recordings in three raagas (Bhairavi, Thodi and Hindolam) which share five common swaras. Due to the limitation on the number of available recordings, choosing many swaras in our task will make it difficult to assess the complementarity of the new parameters, and could also potentially result in over-fitting (more features than instances). Therefore, we chose one swara to perform the raaga classification task.

The task is performed using two feature sets, both having four features. One set consists of just the position and height of the swara in the middle and upper octaves (common swara parametrization, hence used as a baseline). To ensure fairness, we have used two feature selection methods and different classifiers [5, 9] [4] over sub-

[4] The implementations provided in Weka were used with default parameters.

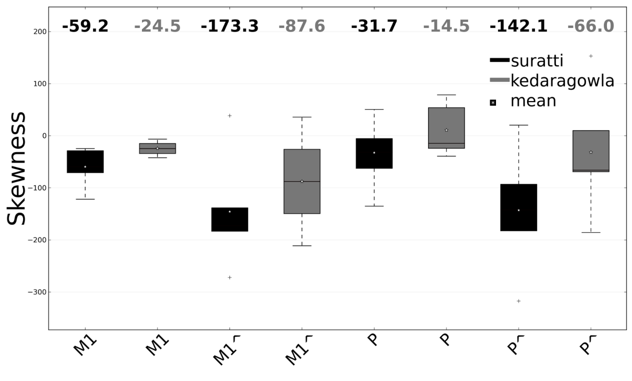


Figure 3. Skewness values for swaras of Surati and Kedaragowla raagas.

sampled data sets in a 10-fold cross validation test. The other feature set is obtained by using information gain-based and correlation-based feature selection methods [9] on a combination of position, height, skewness, kurtosis, variance and the mean of the distribution. Both feature selection methods select new parameters different from the position and the height.

Table 1 shows the averaged results obtained. Though the accuracy increments are not statistically significant, they are indicative of the worthiness of additional parameters. However, if we just consider the classification of two raagas with 26 recordings and perform the same test, the results show a significant increase in the overall accuracy (Table 2). These two explorative tasks show a consistent increase in the accuracy of the system, which indicates the usefulness and complementarity of the proposed peak characteristics.

3.2 Allied Raagas

In the experiment described in the previous section the raagas share a subset of the swaras. However, there is a class of raagas which share exactly the same set of swaras, but have different characteristics, called allied raagas. Since the swaras are common for the raagas, the discriminative capability of the peak position and/or mean will be considerably low. Therefore, these raagas constitute a good repertoire to test our approach. We consider three sets of allied raagas which together have 60 recordings in 7 raagas. The first set has three raagas, and the second and the third sets have two raagas each.

Fig. 2 (a) shows the kurtosis values for swaras of the first set of allied raagas (Bhairavi, Mukhari and Manji). In all the figures that follow, we have only shown the values for the most relevant swaras due to space constraints. $R_2/G_1^\wedge$ [5] and M_1 swaras can be seen to play a notable role in discriminating the raagas in this set. Fig. 2 (b) shows the skewness values for swaras of the same set of allied raagas. The distinction is observed even better through the skewness values of D_2/N_1, P and $P^\wedge$ swaras. Generally, in a given raaga, the melodic context of a swara is kept consistent across octaves, which means that the intonation characteristics of a swara across octaves should be consistent. The skewness values of P and $P^\wedge$ swaras assert this.

Figs. 2 (c) & (d) show the kurtosis and the variance values, respectively, for swaras of the second set of allied raagas (Begada and Kambhoji). All the swaras can be observed to play an equal role in distinguishing raagas of this set. The consistency of the variances of G_3, and R_2/G_1, and kurtosis of G_3 and D_2/N_1 across octaves is quite evident. Fig. 3 shows the skewness values for swaras of the third set of allied raagas (Surati and Kedaragowla). The observations from this set further reinforce the usefulness of the peak parametrization approach to describe intonation.

3.3 Analysis of peak positions

Table 3 shows the average of peak positions of each swara across all the available recordings (we now use the full data set), and the absolute sum of the differences of all observations from the corresponding equi-tempered and just-intonation intervals. The swaras which are observed in less than 20 recordings are not shown. There is a general tendency to just-intonation intervals compared to equi-tempered, which is in agreement with the results obtained by Serrà et al. [12]. However, this tendency is not very evident, supporting the claims in [7]. What interests us more here is the relevance of the values shown in the table for understanding the intonation of swaras in different raagas. For that we consider the set of the average values (Table 3) for each swara to be a general template. A similar template is obtained for each raaga, and the differences between the two templates are analysed and interpreted to check if they are musically meaningful.

Swara	Mean	D_E	D_J	Recordings
Sa	1.92	6.43	6.43	142
$R2/G1$	200.93	10.93	11.62	68
$G3$	384.67	16.34	10.51	56
$M1$	495.05	9.94	9.56	123
P	700.18	6.01	6.21	164
$D2/N1$	889.58	13.99	11.06	87
$D3/N2$	987.87	16.0	14.11	56
$Sa^\wedge$	1196.62	6.35	6.35	174
$R2/G1^\wedge$	1401.37	8.86	8.97	96
$G3^\wedge$	1583.63	17.21	10.04	61
$M1^\wedge$	1693.05	12.71	12.17	91
$P^\wedge$	1897.86	7.45	8.08	118
$D2/N1^\wedge$	2095.6	8.78	13.15	54
$D3/N2^\wedge$	2192.15	14.17	13.33	28

Table 3. Mean of peak positions of each swara, and the differences from corresponding Equi-tempered (D_E) and Just-Intonation (D_J) intervals.

Figs. 4 (a), (b) and (c) show the boxplots for positions of D_2, N_2 and M_1 respectively, in various raagas. They were examined by a trained musician who interpreted them, and asserted that they made sense in the context of today's practice of the raagas. For instance, it is said that D_2 in Khamas raaga is sung without any gamaka, whereas the same swara in Kalyani raaga is sung with a particular gamaka that might have been responsible for the observed phenomenon. This explains the observations made from Fig. 4 (d). Furthermore, as a sanity test, we have plotted the positions of the swara P in various raagas. This swara is normally expected to be sung without any gamaka. Hence, we expect the peak positions corresponding to P of all the raagas to be centered around the mean position observed in the general template (Table 3). Fig. 4 asserts this, except for minor deviations.

4. CONCLUSIONS

We have proposed a peak parametrization approach to describe intonation in Carnatic music and evaluated it qualitatively using three tasks. All the tasks discriminate raagas with the obtained information of swara intonation. However there are a few challenges in this approach. Few swaras, by the nature of the role they play, will not be manifested as peaks at all. Rather, they will appear as a slide that cannot be identified by a peak detection algorithm. Characterizing pitch distributions near all the theoretical intervals or from the general template shown in Sec. 3.3, irrespective of whether it is identified as a peak or not, is one possible way to address this issue. However, identifying few heuristics that will help in locating such slides can be a good substitute, since it falls in line with our methodology in not assuming any particular tuning. The future direction of this work is to extend it to the Hindustani music tradi-

[5] $\wedge$ denotes the swara in upper octave.

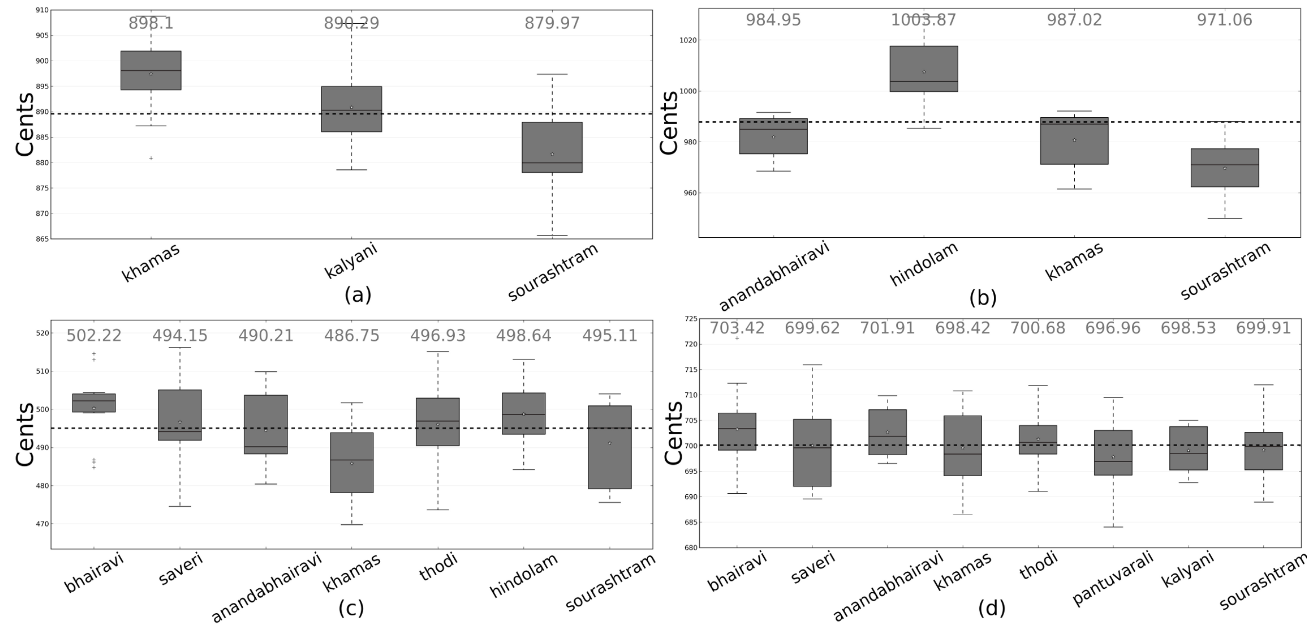


Figure 4. (a). Peak positions of D_2 for recordings in Khamas, Kalyani and Sourashtram raagas. (b). Peak positions of N_2 for recordings in Ananda Bhairavi, Hindoam, Khamas, and Sourashtram raagas. (c). Peak positions of M_1 for recordings in several raagas. (d). Peak positions of P for recordings in several raagas. The dashed line shows the mean of the corresponding swara obtained from the general template.

tion, and also to characterize performers and instruments by their preferred intonation.

5. ACKNOWLEDGMENTS

This research was partly funded by the European Research Council under the European Union's Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583). JS acknowledges JAEDOC069/2010 from Consejo Superior de Investigaciones Científicas, 2009-SGR-1434 from Generalitat de Catalunya, TIN2009-13692-C03-01 from the Spanish Government, and EU Feder Funds.

6. REFERENCES

[1] Shreyas Belle, Rushikesh Joshi, and Preeti Rao. Raga Identification by using Swara Intonation. *Journal of ITC Sangeet Research Academy*, 23, 2009.

[2] B. Bozkurt, O. Yarman, M. K. Karaosmanolu, and C. Akkoç. Weighing Diverse Theoretical Models on Turkish Maqam Music Against Pitch Measurements: A Comparison of Peaks Automatically Derived from Frequency Histograms with Proposed Scale Tones. *Journal of New Music Research*, 38(1):45–70, 2009.

[3] A. de Cheveigné and H. Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[4] J. D. Deng, C. Simmermacher, and S. Cranefield. A Study on Feature Analysis for Musical Instrument Classification. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(2):429–438, 2008.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H Witten. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[6] G. K. Koduri, P. Rao, and S. Gulati. A Survey Of Raaga Recognition Techniques And Improvements To The State-Of-The-Art. In *Sound and Music Computing*, 2011.

[7] A. Krishnaswamy. On the twelve basic intervals in South Indian classical music. *Audio Engineering Society Convention*, page 5903, 2003.

[8] M. Levy. *Intonation in North Indian Music*. Biblia Implex Pvt. Ltd, New Delhi, 1982.

[9] T. Ngo. Data mining: practical machine learning tools and technique, third edition by Ian H. Witten, Eibe Frank, Mark A. Hell. *SIGSOFT Softw. Eng. Notes*, 36(5):51–52, 2011.

[10] Justin Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *Audio, Speech, and Language Processing, IEEE Transactions on*, (99):1–1, 2012.

[11] Justin Salamon, Sankalp Gulati, and Xavier Serra. A Multipitch Approach to Tonic Identification in Indian Classical Music. In *ISMIR*, page In press, 2012.

[12] J. Serrà, G. K. Koduri, M. Miron, and X. Serra. Assessing the tuning of sung indian classical music. In *ISMIR*, pages 157–162, 2011.

[13] V. Shankar. *The art and science of Carnatic music*. Music Academy Madras, Chennai, 1983.

[14] M. Subramanian. Carnatic Ragam Thodi Pitch Analysis of Notes and Gamakams. *Journal of the Sangeet Natak Akademi*, XLI(1):3–28, 2007.

[15] D. Swathi. *Analysis of Carnatic Music : A Signal Processing Perspective*. Masters thesis, IIT Madras, 2009.

DISCRIMINATIVE NON-NEGATIVE MATRIX FACTORIZATION FOR MULTIPLE PITCH ESTIMATION

Nicolas Boulanger-Lewandowski, Yoshua Bengio and Pascal Vincent
Dept. IRO, Université de Montréal
Montréal, Québec, Canada H3C 3J7
{boulanni, bengioy, vincentp}@iro.umontreal.ca

ABSTRACT

In this paper, we present a supervised method to improve the multiple pitch estimation accuracy of the non-negative matrix factorization (NMF) algorithm. The idea is to extend the sparse NMF framework by incorporating pitch information present in time-aligned musical scores in order to extract features that enforce the separability between pitch labels. We introduce two discriminative criteria that maximize inter-class scatter and quantify the predictive potential of a given decomposition using logistic regressors. Those criteria are applied to both the latent variable and the deterministic autoencoder views of NMF, and we devise efficient update rules for each. We evaluate our method on three polyphonic datasets of piano recordings and orchestral instrument mixes. Both models greatly enhance the quality of the basis spectra learned by NMF and the accuracy of multiple pitch estimation.

1. INTRODUCTION

Non-negative matrix factorization (NMF) is an unsupervised technique to discover parts-based representations underlying non-negative data [12], i.e. a set of characteristic components that can be combined additively to reconstitute the observations. When applied to the magnitude spectrogram of a polyphonic audio signal, NMF can discover a basis of interpretable recurring note events and their associated time-varying encodings, or *activities*, that together optimally reconstruct the original spectrogram.

In general, the extracted representation will converge to individual note spectra provided the following conditions are met [5]. First, each observed spectrogram frame must be representable as a non-negative linear combination of the isolated note spectra, an approximation that depends on the interference between overlapping harmonic partials in a polyphonic mix but that is nevertheless reasonable [22]. The second condition requires that basis spectra be linearly independent, and the third condition requires that all combinations of individual notes be present in the database. This last assumption is of course difficult to achieve completely but partial combinations seem sufficient in practice. Consequently, the activities extracted by NMF have proven useful as features to detect individual note pitches played simultaneously at a given instant in a polyphonic audio signal, a task known as multiple pitch estimation, and for the related task of transcribing audio excerpts into musical notation [1, 3, 4, 19]. Sparsity, temporal and spectral priors have proven useful to enhance the accuracy of multiple pitch estimation [3, 7, 20].

Since NMF is an unsupervised technique, it can be applied in principle to an unlimited number of musical recordings without the need for ground-truth *pitch labels*. However, such information is often readily available as recorded expressive performances, symbolic sequences (e.g. a MIDI file) or time-aligned musical scores. In those cases, we would like to exploit the pitch information to steer the NMF decomposition in a supervised way to obtain discriminative features more useful for multiple pitch estimation. A few attempts have been made in this direction, notably by adding a linear discriminant analysis (LDA) stage to the activities extracted by NMF [23], or by embedding Fisher-like discriminant constraints inside the decomposition [9, 21, 23]. Discriminative dictionaries have also been developed for sparse coding [15]. Those methods however are designed for classification, which means choosing a single label, whereas multiple pitch estimation is a multi-label task, i.e. multiple pitch labels can be associated with a single spectrogram frame. In this context, we propose two discriminative criteria that maximize inter-class scatter for each label separately and estimate the predictive power of a given decomposition using logistic regressors. Those ideas are applied in the conventional latent variables framework of NMF and in a deterministic autoencoder model to directly maximize test-time discriminative performance. Efficient update rules are devised for each, and we show that our method greatly improves the quality of the basis spectra learned by NMF and the accuracy of multiple pitch estimation on three polyphonic datasets of piano recordings and orchestral instrument mixes.

The remainder of this paper is organized as follows. In Sections 2 and 3, we review the NMF algorithm and its application to multiple pitch estimation. In Sections 4 and 5 we introduce the latent variables and autoencoder discriminative models. We describe our experiments and evaluate our method in Sections 6 and 7.

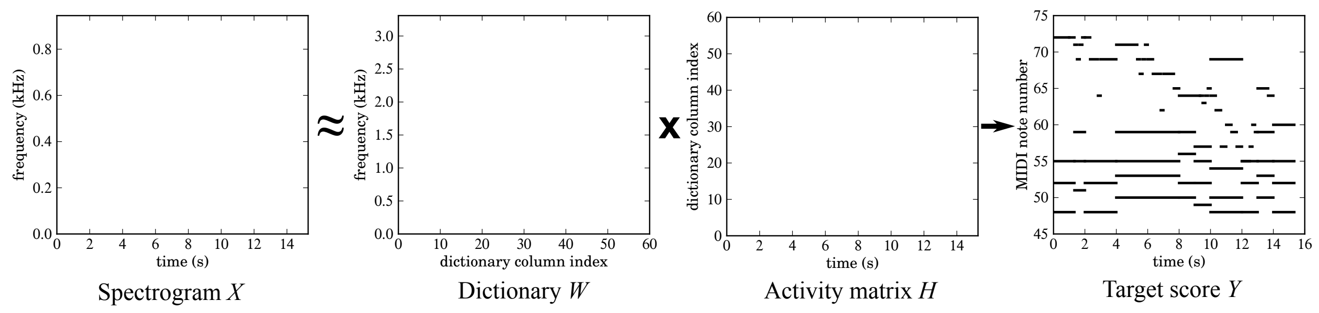


Figure 1. Illustration of the sparse NMF decomposition ($\lambda = 0.01$, $\mu = 10^{-5}$) of an excerpt of Drigo's *Serenade*. Using a dictionary W pretrained on a polyphonic piano dataset, the spectrogram X is transformed into an activity matrix H approximating the piano-roll transcription Y. The columns of W were sorted by increasing estimated pitch for visualization.

2. NON-NEGATIVE MATRIX FACTORIZATION

The NMF method aims to discover an approximate factorization of an input matrix X:

$$\overset{n_f \times n_t}{X} \simeq \overset{n_f \times n_t}{\Lambda} \equiv \overset{n_f \times m}{W} \cdot \overset{m \times n_t}{H} \tag{1}$$

where X is the observed magnitude spectrogram with time and frequency dimensions n_t and n_f respectively, Λ is the reconstructed spectrogram, W is a dictionary matrix of m basis spectra and H is the activity matrix. Non-negativity constraints $W_{i,j} \geq 0, H_{i,j} \geq 0$ apply on both matrices. NMF seeks to minimize the *reconstruction error*, a distortion measure between the observed spectrogram X and the reconstruction Λ. A popular choice is the Euclidean distance:

$$C_{LS} \equiv ||X - \Lambda||^2 \tag{2}$$

with which we will demonstrate our method although it can be easily generalized to other distortion measures in the β-divergence family [11]. Minimizing C_{LS} can be achieved by alternating multiplicative updates to H and W [13]:

$$H \leftarrow H \circ \frac{W^T X}{W^T \Lambda} \tag{3}$$

$$W \leftarrow W \circ \frac{X H^T}{\Lambda H^T} \tag{4}$$

where the $\circ$ operator denotes element-wise multiplication, and division is also element-wise. These updates are guaranteed to decrease the reconstruction error assuming a local minimum is not already reached. While the objective is convex in either W or H separately, it is non-convex in W and H together and thus finding the global minimum is intractable in general.

2.1 Sparsity constraints

In a polyphonic signal with relatively few notes played at any given instant, it is reasonable to assume that active elements H_{ij} should be limited to a small subset of the available basis spectra. To encourage this behavior, a sparsity penalty C_S can be added to the total SNMF objective [10]:

$$C_S = \lambda |H| \tag{5}$$

where $|\cdot|$ denotes the L_1 norm and λ specifies the relative importance of sparsity. In order to eliminate underdetermination associated with the invariance of WH under the transformation $W \rightarrow WD$, $H \rightarrow D^{-1}H$, where D is a diagonal matrix, we impose the constraint that the basis spectra have unit norm. Equation (3) becomes:

$$H \leftarrow H \circ \frac{W^T X}{W^T \Lambda + \lambda} \tag{6}$$

and the multiplicative update to W (equation 4) is replaced by projected gradient descent [14]:

$$W \leftarrow W - \mu(\Lambda - X)H^T \tag{7}$$

$$W_{:i} \leftarrow \frac{W_{:i}}{||W_{:i}||} \tag{8}$$

where $W_{:i}$ is the i-th column of W, μ is the learning rate and $1 \leq i \leq m$.

3. NMF FOR MULTIPLE PITCH ESTIMATION

The ability of NMF to extract fundamental note events from a polyphonic mixture makes it an obvious stepping stone for multiple pitch estimation. In the ideal scenario, the dictionary W contains the spectrum profiles of individual notes composing the mix and the activity matrix H approximately corresponds to the ground-truth score. An example of the sparse NMF decomposition of an excerpt of Drigo's *Serenade* using a dictionary pretrained on a simple polyphonic piano dataset is illustrated in Figure 1. The dictionary contains mostly monophonic basis spectra that were sorted by increasing estimated pitch for visualization. We also observe a clear similarity between the activity matrix and the target score in a piano-roll representation Y.

There are many options to exploit the NMF decomposition to perform actual multiple pitch estimation. The *dictionary inspection* approach [1, 18, 19] consists in estimating the pitch (or lack thereof) of each column of W, which can be done automatically using harmonic combs [20], and to transcribe all pitches for which the associated H_{ij} activities exceed a threshold η:

$$Y_{kj} = 1 \Leftrightarrow \sum_{i|L(i)=k} H_{ij} \geq \eta \tag{9}$$

where $L(i)$ is the estimated pitch label (index) of the i-th basis spectrum. For this method, a new factorization can be performed adaptively for each new piece to analyze, or the dictionary can be pretrained from an extended corpus and kept fixed during testing. Dictionaries can also be constructed from the concatenation of isolated note spectra [3,4].

Another option is to predict each column of Y from the corresponding column of H using a general-purpose multi-label classifier or a set of binary classifiers, one for each label (note) in the designated range. This obviously requires the use of a fixed dictionary and the availability of annotated pieces to train the classifiers. In this work, we will exclusively employ pretrained dictionaries and we will consider both dictionary inspection and multi-label classification with linear support vector machines (SVM) [17].

4. DISCRIMINATIVE CRITERIA

The simple interpretation of the activity matrix as an approximate transcription usually deteriorates when we increase instrumental diversity, pitch range or polyphony. In this section, we introduce two discriminative criteria exploiting the aligned score information Y to ensure that NMF extracts meaningful features into W and H.

The first criterion is inspired from linear discriminant analysis in that we aim to maximize the inter-class scatter of the H_{ij}, where the classes here refer to the presence or absence of a given pitch label at a given time. We encourage the activities associated with a given basis spectrum to be maximal when its pitch is present in the score and minimal otherwise, such that a unidimensional decision threshold is sufficient to estimate the presence of a note. We first assign a pitch label $L(i)$ to each column i of W, or set $L(i) = -1$ to denote an unpitched basis spectrum. Due to the invariance of WH under the column permutation of W and the equivalent row permutation of H, this assignment can be done arbitrarily as long as the number of basis spectra describing each pitch (q) and the number of unpitched spectra ($\bar{q}$) remain constant. More precisely, this criterion has the form:

$$C_d(H) = \sum_{ij} \begin{cases} -\beta^+ H_{ij} & \text{if } Y_{L(i),j} = 1 \\ \beta^- H_{ij} & \text{if } Y_{L(i),j} = 0 \\ 0 & \text{if } L(i) = -1 \end{cases} \tag{10}$$

where the β^+ and β^- parameters quantify respectively the importance of presence and absence of an H_{ij} element. Note that the limit $\beta^- \to \infty$ corresponds to setting $H_{ij} = 0$ for $Y_{L(i),j} = 0$.

The second proposed criterion does not impose a predetermined structure on the activity matrix, but rather attempts to determine whether H is a good predictor for Y. We introduce a stage of logistic regressors with weight matrix V and bias vector b using H as input:

$$p_{kj} = \sigma((VH)_{kj} + b_k) \tag{11}$$

where $\sigma(x) \equiv (1+e^{-x})^{-1}$ is the element-wise logistic sigmoid function and p is an output matrix of note probabilities, or *probabilistic piano-roll*. We use the cross-entropy as a discriminative criterion for H:

$$C_l(H) = -\alpha \sum_{kj} Y_{kj} \log p_{kj} + (1 - Y_{kj}) \log(1 - p_{kj}) \tag{12}$$

where α is a weighting coefficient. Adding our criteria to the total objective yields the DNMF model:

$$C = C_{LS} + C_S + C_d + C_l. \tag{13}$$

It is easy to show that the Hessian matrices $\nabla^2_H C_d(H)$ and $\nabla^2_H C_l(H)$ are both positive semi-definite and that the DNMF objective remains convex in W or H separately. The multiplicative update rule for H (equation 6) becomes:

$$H \leftarrow H \circ \frac{W^T X}{W^T \Lambda + \lambda + \frac{\partial C_d(H)}{\partial H} + \frac{\partial C_l(H)}{\partial H}} \tag{14}$$

where the gradients are given by:

$$\frac{\partial C_d(H)}{\partial H_{ij}} = \begin{cases} -\beta^+ & \text{if } Y_{L(i),j} = 1 \\ \beta^- & \text{if } Y_{L(i),j} = 0 \\ 0 & \text{if } L(i) = -1 \end{cases} \tag{15}$$

$$\frac{\partial C_l(H)}{\partial H} = \alpha V^T (p - Y). \tag{16}$$

The update rules for W are the same as for sparse NMF and are given by (7) and (8). The V and b parameters are optimized via stochastic gradient descent using the updates:

$$V \leftarrow V - \mu (p - Y) H^T \tag{17}$$

$$b_k \leftarrow b_k - \mu \sum_j (p_{kj} - Y_{kj}). \tag{18}$$

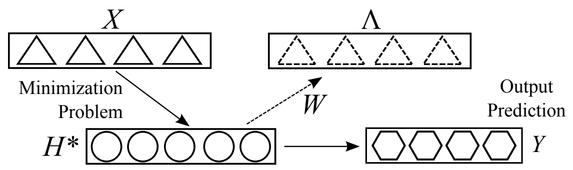


Figure 2. In the DNMF autoencoder model, the input is encoded via a deterministic minimization procedure. The code H^* is trained to reconstruct X and to predict Y.

5. AUTOENCODER MODEL

In the probabilistic latent variables model (LV) underlying NMF, the activities are regarded as hidden variables with joint negative log probability given by (13) and the use of equations (14) and (7-8) during training corresponds to the expectation and maximization phases of an EM algorithm [12]. A subtlety associated with this interpretation arises in testing conditions when the labels Y are unknown. We can resort to equation (6) to infer H, but it is possible to address this issue in a more principled manner with the autoencoder model (AE) presented in this section.

Let us consider the value of H obtained in testing conditions, denoted H^*:

$$H^*(W) \equiv \arg\min_H (C_{LS} + C_S) \tag{19}$$

and let us apply the same discriminative criteria $C_d(H^*)$ and $C_l(H^*)$ on that variable. Since H^* is a purely deterministic function of the input with W the only learned parameter, this model can be assimilated to an autoencoder with the encoding step consisting in a complex minimization problem (equation 19) and the decoding step is the usual linear input reconstruction (equation 1). In addition, the discriminative criteria encourage H^* to be a good predictor of Y. The overall model is depicted in Figure 2. The projected gradient descent update for W becomes:

$$W \leftarrow W - \mu \frac{\partial C(H^*)}{\partial W} \tag{20}$$

$$W_{:i} \leftarrow \frac{W_{:i}}{||W_{:i}||} \tag{21}$$

Since $H^*(W)$ is the result of an optimization process, the gradient of $C(H^*)$ with respect to W is not trivial to compute. We can exploit the convergence guarantee of the multiplicative update (6) to express H^* as an infinite sequence truncated to K iterations:

$$H^* = \lim_{k\to\infty} H^k \simeq H^K \tag{22}$$

where:

$$H^{k+1} = H^k \circ \frac{W^T X}{W^T W H^k + \lambda} \tag{23}$$

from which the gradients are easily computed by backpropagation through iteration k in an efficient $O(K)$ time:

$$\frac{\partial C}{\partial H^k} = \frac{\partial C}{\partial H^{k+1}} \circ \frac{H^{k+1}}{H^k} - W^T W B^k \tag{24}$$

for $0 \leq k < K$, where the auxiliary variable B^k is:

$$B^k = \frac{\partial C}{\partial H^{k+1}} \circ \frac{H^{k+1}}{W^T W H^k + \lambda}. \tag{25}$$

The initial conditions are:

$$\frac{\partial C}{\partial H^K} = W^T(WH^K - X) + \lambda + \frac{\partial C_d}{\partial H^K} + \frac{\partial C_l}{\partial H^K} \tag{26}$$

where the two rightmost terms are given by (15) and (16) with $H = H^K$. The gradient with respect to W is then given by:

$$\frac{\partial C}{\partial W} = \sum_{k=0}^{K-1} \Big[X \Big(\frac{\partial C}{\partial H^{k+1}} \circ \frac{H^{k+1}}{W^T X} \Big) - W(B^k H^{kT} + H^k B^{kT}) \Big] + (WH^K - X)H^{KT}. \tag{27}$$

When computing $\partial C/\partial W$, the finite-sequence approximation (22) needs only be accurate in the vicinity of the current value of W, denoted W^0. We can increase efficiency without sacrificing precision by initializing $H^0 \equiv H^*(W^0)$ and keeping K small (< 10). Note also that this gradient may become infinite when W is rank deficient, a condition that arises when combinations of basis spectra momentarily align [8]. This optimization issue is alleviated in practice by two facts: the basis spectra are renormalized after each update (equation 21), and the use of a finite sequence to approximate the gradient tends to smooth out singularities.

6. EVALUATION

We use three datasets to evaluate our method:

RAND is a piano dataset of random chords part of the larger MAPS database [6]. Each chord contains from 2 to 7 notes sampled from the whole piano range with heterogeneous loudnesses. We randomly split the data into training, validation and test sets using a 4:1:1 ratio.

ORC is a random polyphonic dataset similar to RAND, but that includes common orchestral instruments such as violin, cello, trumpet, French horn, saxophone, oboe, bassoon, clarinet, flute and piccolo, in addition to piano and organ. Each of the 3000 tracks contains 5 instruments simultaneously playing in their respective range for 16 seconds and was rendered with the FluidR3 SoundFont [1].

MUS is a collection of classical piano pieces also included in MAPS [6], that contains nine sets created by high-quality software synthesizers (7 sets) and a Yamaha Disklavier (2 sets). Five synthesizer sets were selected for training, with the remaining two held out for validation to avoid overfitting the specific piano tones heard during training. We used the first 30 seconds of each piece from the Disklavier sets for test. The average polyphony for this dataset is 2.9.

The magnitude spectrogram was computed for all datasets by the short-term Fourier transform using a 93 ms sliding Blackman window at 10 ms intervals. Each spectrogram frame (column of X) was normalized and square root compressed to reduce the dynamic range. The ground truth Y was directly inferred from the MIDI files [6].

We evaluate multiple pitch estimation performance with the standard metrics of accuracy, precision, recall and F-measure [2]. Either dictionary inspection or linear SVMs using H^* or X as input serve to estimate the pitches. The SVMs can optionally be replaced by multilayer perceptrons (MLP) [16] for comparison. For each NMF model, the parameters are first selected to maximize accuracy on the validation set and we report the final performance on the test set. Parameters are optimized over predetermined search grids on the following intervals:

$q \in [1, 7]$	$\bar{q} \in [0, 12]$	$\eta \in [0, 20]$
$\beta^{\pm} \in [10^{-6}, 10]$	$\alpha \in [10^{-2}, 10^2]$	
$\lambda \in [10^{-7}, 2]$	$\mu \in [10^{-6}, 10^{-3}]$	

7. RESULTS

To illustrate the effectiveness of our approach, we first evaluate qualitatively the learned basis and pitch activities on polyphonic piano data. The dictionary matrices obtained on RAND via unsupervised NMF (Fig. 3(a)) and DNMF (Fig. 3(b)) are presented in Figure 3 after sorting the columns by increasing estimated pitch. From these results, it is clear that DNMF extracted basis spectra – from a purely polyphonic mix – that correspond much closely to the expected spectrum of individual piano notes. It is thus not surprising that applying those dictionaries to extract pitch activities H^* from an excerpt of the MUS test set (Fig. 4(a)) yielded

[1] `http://www.hammersound.net`

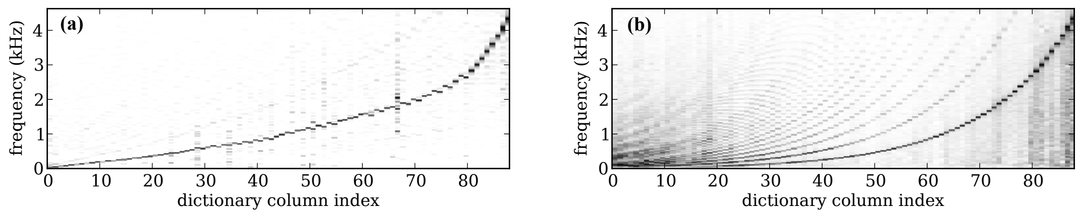


Figure 3. Dictionaries trained ($q = 1$, $\bar{q} = 0$) on the RAND dataset via NMF (a) and DNMF (b). Columns were sorted by increasing estimated pitch for visualization.

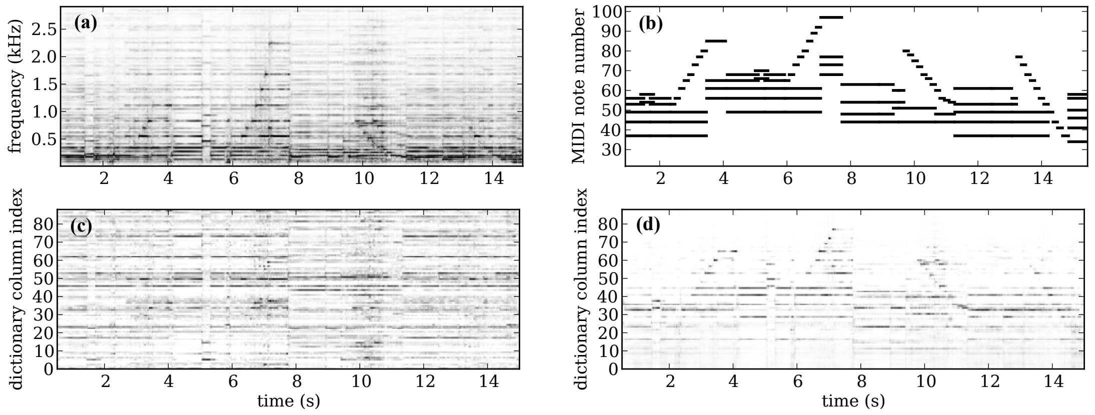


Figure 4. Spectrogram (a) and piano-roll score (b) for the first 15 seconds of an arpeggiated version of *Silent Night, Holy Night* from the MUS test set. Pitch activities H^* (c-d) were estimated for that signal using the pretrained dictionaries in Figure 3(a-b) respectively.

less noisy estimates much closer to the ground-truth score (Fig. 4(b)), as can be observed from Figure 4(c-d).

A more quantitative measure of the discriminative quality of the learned basis is the discriminative ratio r:

$$r(H) = \left(\sum_{i,j|Y_{L(i),j}=1} H_{ij} \Bigg/ \sum_{i,j|Y_{L(i),j}=0} H_{ij} \right). \quad (28)$$

According to this definition, we obviously favor higher ratios. While r can be made arbitrarily high in training conditions simply by increasing $\beta^{\pm}$, what we really care about is its value in testing conditions $r(H^*)$. Figure 5 shows a significant increase in the test discriminative ratio with our latent variable algorithm compared to the sparse NMF baseline, which indicates a much better pitch label separability. The additional improvement provided by the autoencoder model demonstrates that directly optimizing H^* is useful to increase discriminative performance.

In the next experiments we verify if the discriminative features learned by our models translate in good pitch estimation performance. Frame-level accuracies on the RAND and ORC datasets are presented in Table 1 using dictionary inspection and in Table 2 for multi-label classification. The proposed models outperform the baselines in all cases, especially DNMF-AE used in conjunction with SVMs. Table 3 shows frame-level precision, recall and F-measure results on the MUS test set for common existing NMF variants. Our approach surpasses adaptive unconstrained NMF and is competitive with NMF trained on isolated piano notes and NMF with spectral constraints [20].

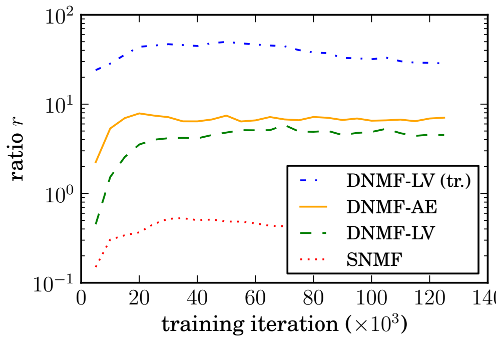


Figure 5. Evolution of the ratio $r(H^*)$ during training on the RAND dataset. “tr” stands for training conditions.

Method	RAND	ORC
NMF	27.6%	30.0%
SNMF	32.3%	43.8%
DNMF-LV	53.2%	**58.8%**
DNMF-AE	**53.4%**	58.6%

Table 1. Multiple pitch estimation accuracy obtained by dictionary inspection on the RAND and ORC datasets.

8. CONCLUSION

We have shown that by exploiting pitch information present in time-aligned musical scores to encourage the extracted features to discriminate against the pitch labels, we can improve the multiple pitch estimation performance of NMF on three datasets of polyphonic music. Interestingly, the

Features	RAND	ORC
Spectrogram	50.9%	55.9%
NMF	56.2%	59.4%
SNMF	55.5%	59.5%
DNMF-LV	60.4%	63.3%
DNMF-AE	**61.6%**	**65.5%**
Spectrogram (MLP)	52.7%	62.0%

Table 2. Multiple pitch estimation accuracy obtained on the RAND and ORC datasets via linear SVMs using the specified feature extraction technique.

NMF variant	Prec.	Rec.	F-meas.
No training			
Unconstrained †	58.9%	60.0%	57.8%
Spectral constraints [20]	71.6%	65.5%	67.0%
Pretrained dictionary			
Isolated note spectra †	68.6%	66.7%	66.0%
Proposed (DNMF-LV)	68.1%	65.9%	66.9%
Proposed (DNMF-AE)	66.8%	**68.7%**	**67.8%**
Other methods			
SONIC [16]	**74.5%**	57.6%	63.6%

Table 3. Average multiple pitch estimation performance of common NMF variants on the MUS (MAPS) piano dataset. †These results are from Vincent [20].

resulting basis spectra closely resemble the spectrum of individual piano notes, even though they were trained on purely polyphonic data without explicit harmonicity constraints. Once that discriminative basis is learned, relevant pitch activity features can be efficiently computed using only standard multiplicative updates.

9. ACKNOWLEDGMENTS

The authors would like to thank NSERC, CIFAR and the Canada Research Chairs for funding.

10. REFERENCES

[1] S.A. Abdallah and M.D. Plumbley. Unsupervised analysis of polyphonic music by sparse coding. *IEEE Trans. on Neural Networks*, 17(1):179–196, 2006.

[2] M. Bay, A.F. Ehmann, and J.S. Downie. Evaluation of multiple-F0 estimation and tracking systems. In *ISMIR*, 2009.

[3] A. Cont. Realtime multiple pitch observation using sparse non-negative constraints. In *ISMIR*, 2006.

[4] A. Dessein, A. Cont, and G. Lemaitre. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. In *ISMIR*, 2010.

[5] D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts. In *NIPS 16*, 2003.

[6] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Trans. on Audio, Speech, and Language Processing*, 18(6):1643–1654, 2010.

[7] D. Fitzgerald, M. Cranitch, and E. Coyle. Generalised prior subspace analysis for polyphonic pitch transcription. In *DAFX 8*, 2005.

[8] G.H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal on Numerical Analysis*, pages 413–432, 1973.

[9] N. Guan, D. Tao, Z. Luo, and B. Yuan. Manifold regularized discriminative nonnegative matrix factorization with fast gradient descent. *IEEE Transactions on Image Processing*, 20(7):2030–2048, 2011.

[10] P.O. Hoyer. Non-negative sparse coding. In *Neural Networks for Signal Processing*, pages 557–565, 2002.

[11] R. Kompass. A generalized divergence measure for nonnegative matrix factorization. *Neural computation*, 19(3):780–791, 2007.

[12] D.D. Lee and H.S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[13] D.D. Lee and H.S. Seung. Algorithms for non-negative matrix factorization. In *NIPS 13*, 2001.

[14] C.J. Lin. Projected gradient methods for nonnegative matrix factorization. *Neural computation*, 19(10):2756–2779, 2007.

[15] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *CVPR*, pages 1–8, 2008.

[16] M. Marolt. A connectionist approach to automatic transcription of polyphonic piano music. *IEEE Transactions on Multimedia*, 6(3):439–449, 2004.

[17] G.E. Poliner and D.P.W. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Applied Signal Processing*, 2007(1):154–164, 2007.

[18] P. Smaragdis. Polyphonic pitch tracking by example. In *IEEE WASPAA*, pages 125–128, 2011.

[19] P. Smaragdis and J.C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *IEEE WASPAA*, pages 177–180, 2003.

[20] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):528–537, 2010.

[21] Y. Wang and Y. Jia. Fisher non-negative matrix factorization for learning local features. In *ACCV*, 2004.

[22] C. Yeh and A. Röbel. The expected amplitude of overlapping partials of harmonic sounds. In *ICASSP*, 2009.

[23] S. Zafeiriou, A. Tefas, I. Buciu, and I. Pitas. Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification. *IEEE Transactions on Neural Networks*, 17(3):683–695, 2006.

A FEATURE RELEVANCE STUDY FOR GUITAR TONE CLASSIFICATION

Wolfgang Fohl **Ivan Turkalj** **Andreas Meisel**
HAW Hamburg University of Applied Sciences
{wolfgang.fohl | ivan.turkalj | andreas.meisel}@haw-hamburg.de

ABSTRACT

A series of experiments on the automatic classification of classical guitar sounds with support vector machines has been carried out to investigate the relevance of the features and to minimise the feature set for successful classification. Features used for classification were the time series of the partial tone amplitudes, and of the MFCCs, and the energy distribution of the nontonal percussive sound that is produced in the attack phase of the tone. Furthermore the influence of sound parameters as timbre, player, fret position and string number on the recognition rate is investigated. Finally, several nonlinear kernels are compared in their classification performance. It turns out, that a selection of 505 features out of the full feature set of 1155 elements does only reduce the recognition rate of a linear SVM from 82% to 78%. With the use of a polynomial instead of a linear kernel the recognition rate with the reduced feature set can even be increased to 84%.

1. INTRODUCTION

In the recent years musical instrument recognition has been extensively investigated. Primary research goals were automatic indexing of multimedia data bases, automatic musical genre classification and automatic music transcription systems. A less common topic is the quality assessment of musical instruments, which will be covered in the present paper. Currently the research efforts show several trends. One is the attempt to transfer the successful classification based on single notes to the analysis and classification of solo musical phrases. Joder, Essid, and Richard [9] describe a modification of Support Vector Machines with alignment kernels which they report to perform better than classifiers based on Gaussian Mixture Models or Hidden Markov Models. Barbedo and Tsanetakis [2] published their results on the even more challenging task of instrument classification in polyphonic recordings. Their method is the detection of partial tone structures that are unique to certain instrument groups, which are fed to a specialised decision-tree algorithm.

A second trend in instrument classification research is the systematic comparison of the performance of certain statistical methods and commonly used feature sets in order to reduce the feature space dimensionality and thus escaping the "curse of dimensionality" – the exponential increase of required training samples with increasing dimension of the feature space. A detailed general description of algorithms and procedures of feature space reduction is given by Guyon and Elisseeff [6]. Deng, Simmermacher, and Cranefield published a very good and complete survey on feature relevance for musical instrument classification [3]. Loughran, Walker, and O'Neill present a genetic algorithm approach to feature selection [11]. Genetic algorithms can even be employed for feature *generation*, as described by Mierswa and Morik [12], and by Pachet and Roy [13]. One outcome of these approaches could be the generation of meaningful features, that give an insight in the nature of the investigated sounds.

There is a paper on the quality assessment of musical instruments by Hsiao and Su [7]. They used an waveform-based feature set in conjunction with a multiclass Mahalanobis-Taguchi system to develop an automatic saxophone quality assessment system.

In this article we present a systematic study on the parameters influencing the classification performance of a support vector machine (SVM) classifier to distinguish single tones of three different classical guitars from each other. This continues an earlier work published on the 2008 DAFx conference [4].

Our research motivation is to pinpoint those acoustical features of high-quality instruments that are responsible for the perceived musical quality of the guitar. The classification experiments were conducted to further support our working hypothesis, that the acoustical quality of an instrument reveals itself at least partially already in a single tone. This hypothesis is motivated by the way professional guitarists assess an instrument: They test the acoustical properties of the guitar by carefully listening to single tones played on all strings and over the whole range of the fingerboard.

In the following section our experimental setup and the feature and sample selection strategy is described, followed by the presentation and discussion of classification performance results. The article is concluded by a summary and an interpretation of the results in terms of musical acoustics.

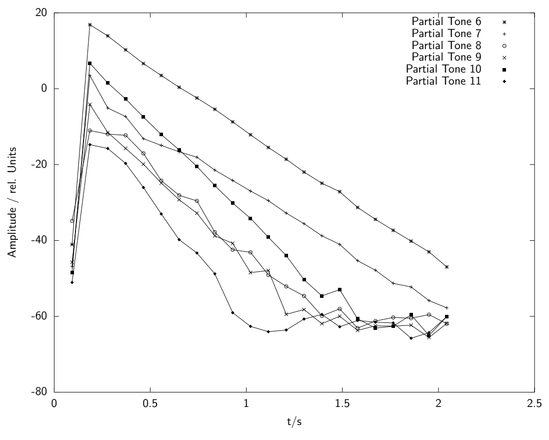


Figure 1. Time series of partial tones 6 – 11 of a guitar tone.

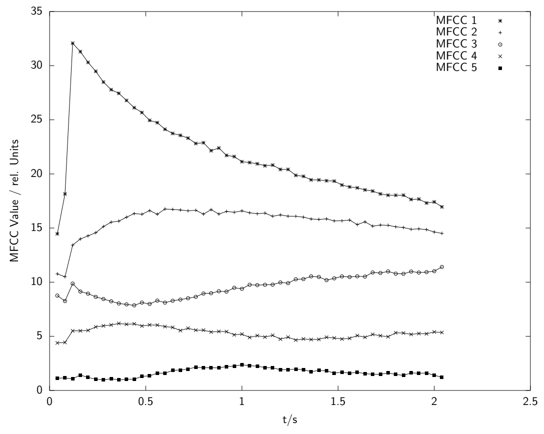


Figure 2. Time series of the first 10 MFCCs of a guitar tone. Curves shifted vertically for better overview.

2. EXPERIMENTS

Single guitar tones of three high-quality classical guitars (by the luthiers Hense, Marin, and Wichmann) played by four players on three different fret positions with the three different sound intentions *sonorous*, *sharp*, and *warm*, were recorded, resulting in ≈ 4000 tone samples, each with a duration of 2–3 seconds. These samples were normalised for equal maximum amplitude and three groups of features were extracted: The time series of the partial tone amplitudes (PT) (see figure 1 for an example), the time series of the mel frequency cepstral coefficients (MFCC) as shown in figure 2, and the power distribution of the *nontonal spectrum* (NT), see [5]. The partial tone time series was obtained by first taking the magnitude spectrum of the whole length of the sound sample, and f_0 was identified by cepstral analysis. Then the sound was split into frames of 4096 samples, each frame multiplied with a Blackman window, the FFT was calculated and the amplitudes of the first 16 partial peaks, i.e., the peaks in the vicinity of $n \cdot f_0$, were evaluated. The data of the first 40 frames was taken as the partial tone feature set.

The MFCC data was computed using the Matlab Auditory Toolbox by Slaney [14], with frame size of 1024 samples, and a frame frequency of 25 Hz. The time series of the first 10 MFCCs was evaluated, and the first 50 frames were taken as the MFCC feature set.

The calculation of the nontonal features starts with the magnitude spectrum of the whole sound sample, from which the tonal peaks have been removed as shown in Fig. 3. To obtain the nontonal features, the nontonal power spectrum P_k at frequency index k is computed by squaring the nontonal spectrum Y_k, and the accumulated power between $f_{\text{start}} = 0$ and $f_{\text{end}} = f_k$ is calculated to yield the fuction C_k:

$$C_k = \sum_{i=0}^{k} P_k \tag{1}$$

The logarithm of this monotonously increasing function is taken, and the range of $\log C_k$ from 1 to its final value

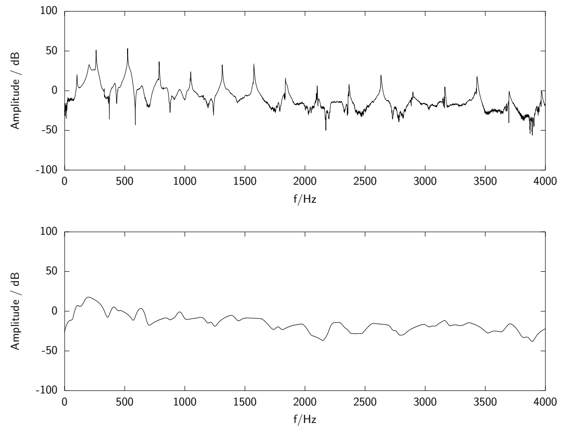


Figure 3. Magnitude spectrum and nontonal magnitude spectrum of a guitar tone.

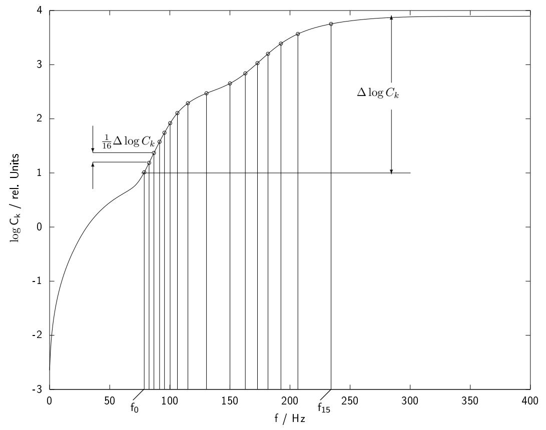


Figure 4. Calculation of nontonal frequency features. Plotted is the function $\log C_k$ from Eq. 1 and the frequencies, that divide the range $\Delta \log C_k$ from $\log C_k = 1$ to the maximum value of $\log C_k$ into 16 equally spaced regions. The corresponding frequencies $f_1 \ldots f_{15}$ are the nontonal feature values.

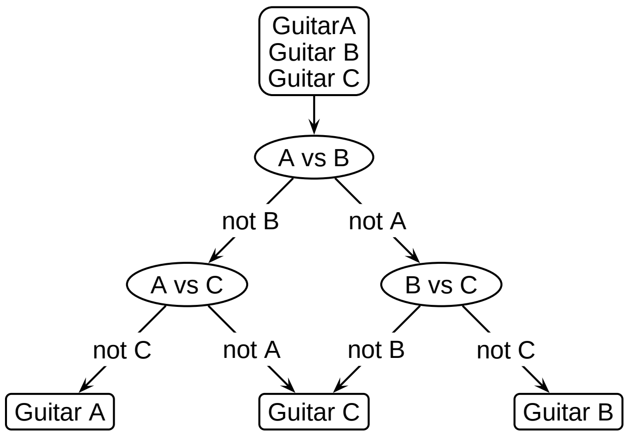


Figure 5. A directed acyclic graph for multi-class identification

is split into 16 equally spaced parts. The corresponding 15 boundary frequencies are taken as the nontonal features, as shown in figure 4.

The complete feature set consists of 1155 elements ($40 \cdot 16\,\mathrm{PT} + 50 \cdot 10\,\mathrm{MFCC} + 15\,\mathrm{NT}$).

These features or subsets of them were used for training the SVMs and for running the classification tests. All but the last group of experiments were performed with linear SVM kernels, the best classification performance with linear kernels was $82.0\,\%$, it was achieved using the full feature set of 1155 elements.

For each guitar, a SVM for a one-vs-rest classification was trained. For the multi-class identification, a directed acyclic graph (DAG) was constructed according to Fig. 5.

The *classification performance* is determined as the ratio of correctly classified test examples and the total number of examples

$$Performance = \frac{n_{\text{correct}}}{n_{\text{total}}} \tag{2}$$

A flexible data processing software has been written in GNU Octave [1], a free Matlab$^{\text{TM}}$ clone, for feature extraction and data selection. For the SVMs, the SVMLight implementation of Joachims has been used [8].

Several series of experiments were carried out to investigate which features are most important for a correct classification, how small the feature set can be made without degrading the classification performance, and if there are nonlinear kernels that perform better than the linear kernel.

In addition, the training and testing was conducted with different subsets of the guitar tone samples to investigate the role of the player, and to test, if a preselection of sounds according to several criteria can improve the classification result.

Two preliminary experiments were performed: A cross-validation of the classification was made by exchanging the sample sets for test and training. In the second preliminary experiment an attempt was made to reduce the feature space by performing a Principal Component Analysis on the training data set.

3. RESULTS AND DISCUSSION

3.1 Classification Performance With the Full Feature Set and a Linear Kernel

As a first step, the classification experiment has been carried out with the full data set of 1155 features, 978 training samples, 972 test samples, with a linear kernel.

The classification performance of $82.0\,\%$ is taken as the reference value for all subsequent experiments.

3.1.1 Cross-validation

The experiment was repeated with the test and training data exchanged. Table 1 compares the results of the two experiments.

Guitar	*Reference*	*Test / Train Data Exchanged*
Overall	$82.0\,\%$	$84.2\,\%$
Hense	$74.4\,\%$	$82.9\,\%$
Marin	$81.8\,\%$	$77.7\,\%$
Wichmann	$88.6\,\%$	$92.0\,\%$

Table 1. Cross-validation of guitar classification with exchanged train and test data

The deviations between the two experiments give an impression of the variances of the classification measurements.

3.2 Principal component analysis (PCA)

As a supporting study, an attempt was made to reduce the dimensionality of the feature space by applying a principal component analysis to the feature data. With the first 500 PCA-Eigenfeatures the classification rate is only $72.1\,\%$, which is significantly worse than the classification result of $77.7\,\%$, based on a manually selected 505-feature set shown in section 3.3.3. An explanation for this poor result might be the fact, that most of the features are *time series*, for which the differences of the neighbouring values carry important information. The preprocessing in the `princomp` function of the Octave/Matlab Statistical Toolbox removes these dependencies by calculating the mean and variance for each feature in the training set separately, and then for each feature value subtracts the mean and scales the variance to unity, thus eliminating the information of the relative magnitudes of the feature values.

To get a substantial reduction of the feature space, an adaption of the PCA method for time series, as described in chapter 12 of the book of Jolliffe [10], will have to be applied.

3.3 Relevance of Features

The experiment series to determine the relevant features were carried out with the full sample data set: 978 samples for training, and 972 samples for testing.

3.3.1 MFCCs

In this series of experiments only the time series of MFCC features were used for classification. In the first experiment set the MFCC coefficients were grouped into lower and upper half (coefficients 1 – 5 and 6 – 10), in the second experiment set the features were divided into three groups (coefficients 1 – 3, 4 – 7, and 8 – 10). All possible combinations of groups in the series were tested, these are the most important results:

MFCC coefficients	*Features*	*Performance*
1 – 10	500	75.5 %
1 – 5	250	71.2 %
6 – 10	250	60.2 %
1 – 7	350	72.0 %
1 – 3	150	65.9 %
4 – 7	150	62.6 %
8 – 10	150	56.8 %

Table 2. Classification performance of subsets of the MFCC features

The MFCC coefficient group 1 – 3 contains the most relevant third of the MFCC coefficients.

3.3.2 Partial Tones (PT)

In this series of experiments several selections of the first 16 partial tones have been made. Again a grouping in halves and thirds has been performed.

Partials	*Features*	*Performance*
1 – 16	640	52.8 %
1 – 11	440	57.2 %
1 – 5	200	44.0 %
6 – 11	240	54.3 % (!)
12 – 16	200	42.7 %

Table 3. Classification performance of subsets of the PT features

The medium third of the partial tones gives not only the best result of the one-third-selection, it is noteworthy, that this reduced feature set even performs better than the full set of partial tones.

3.3.3 Feature Combinations

Several combinations of the nontonal, MFCC, and partial tone features were tested. The best performance was achieved by combining nontonal features with MFCCs 1 – 5 and partial tones 6 – 11 i.e., the best-performing selections of the previous experiments:

Feature selection	*Features*	*Performance*
MFCC 1 – 5, PT 6 – 11 NT 1 – 15	505	77.7 % (!)
MFCC 1 – 5, PT 6 – 11	490	71.9 %
MFCC 1 – 5, NT 1 – 15	265	76.0 %
MFCC 1 – 7, PT 6 – 11 NT 1 – 15	605	77.7 %

Table 4. Classification performance of various feature combinations

Comparison of the first two lines in Table 4 shows, that the addition of the 15 nontonal features increases the classification performance by 5 percent points. It has to be stressed, that the MFCC and the PT features each represent a time series of 50 (MFCC) and 40 (PT) data points respectively, whereas the nontonal energy distribution is a global feature set that consists of only 15 single data values. So it can be concluded, that the nontonal features add new information to the feature set, that is not implicitly contained in the other feature data.

Another notable fact is, that the inclusion of the time series of MFCCs 6 and 7 does not at all affect the classification performance, as can be seen by comparing the first and the last line of Table 4.

3.4 Selections of Tone Samples

In the subsequent experiments certain selections of tone samples were made to further pinpoint the relevance of features. Since the number of training samples is reduced, also a reduced parameter set has to be used, so the most successful combination of the preceding experiments was taken: nontonal features 1 – 15, MFCCs 1 – 5, and partial tones 6 – 11. This is the 505-element feature set of section 3.3.3. In each overview of results the number of audio samples used for test and training is given.

3.4.1 Player

In the first set of the experiments with preselected tone samples, the influence of the player is investigated. In the first part, three out of four players are used for training, the remaining player is taken for testing. In the second part, the training is performed with all players, and again one of the players is used for testing.

Player (Test)	*Samples (Test)*	*Players (Train)*	*Samples (Train)*	*Performance*
1	324	2, 3, 4	654	59.6 %
2	162	1, 3, 4	816	69.1 %
3	324	1, 2, 4	648	65.4 %
4	162	1, 2, 3	816	64.2 %

Table 5. Classification performance, player in test set not included in training set

Player (Test)	*Samples (Test)*	*Players (Train)*	*Samples (Train)*	*Performance*
1	324	1 – 4	978	74.4 %
2	162	1 – 4	978	71.6 %
3	324	1 – 4	978	83.5 %
4	162	1 – 4	978	78.4 %

Table 6. Classification performance, player in test set **is** included in training set

As was to be expected, the players have quite a large influence on the produced sound, and so the classification performance decreases, when the testing player is not in the group of the training players. The classification performance might in this case be improved by a larger pool of players.

3.4.2 *Timbre*

In these experiments sound samples of the same timbre are used for training and testing. The term timbre here refers to the sound intention of the player. Usually, a warm timbre is produced by plucking the string above the sound hole of the guitar with the finger moving in an angle of approx. $45°$ to the string; a sharp timbre is produced by plucking near the bridge with the plucking finger moving perpendicular to the string.

Timbre	*Samples (Train / Test)*	*Performance*
sharp	326 / 324	80.6 %
sonorous	327 / 324	79.0 %
warm	325 / 324	82.7 %

Table 7. Classification performance for different timbres

It would have been expected, that the preselection of timbre would improve the classification performance, but the experiments show, that this is not the case. Obviously the influence of the different strings and the different positions on the fingerboard introduce too much inhomogeneity.

3.4.3 *String*

This series of experiments tests the influence of the string on the sound. Only sounds of the same string are taken for training and testing.

String (Note)	*Samples (Train / Test)*	*Performance*
1 (e')	165 / 162	96.3 %
2 (b)	163 / 162	92.6 %
3 (g)	162 / 162	80.9 %
4 (d)	163 / 162	79.6 %
5 (A)	163 / 162	84.0 %
6 (E)	162 / 162	83.3 %

Table 8. Classification performance for different strings

Obviously the preselection of the string does provide substantially more homogeneous sample sets. The trained SVMs are specialised to the sound of one particular string and perform substantially better than with the whole range of tone samples.

3.4.4 *Fret*

The last experiment series in this sections is devoted to the fret, i.e., the position on the fingerboard .

The observed performances of Table 9 are approximately the same as the overall performance given in Table 1.

Fret	*Samples (Train / Test)*	*Performance*
1	326 / 324	84.9 %
5	176 / 174	79.9 %
6	150 / 150	78.7 %
10	314 / 312	81.7 %

Table 9. Classification performance for different fret positions

4. NONLINEAR KERNELS

In a last series of experiments different nonlinear kernels were used for classification. Again the 505-element feature set of section 3.3.3 is used.

The only nonlinear kernel provided by SVMLight, that gave satisfactory results was the polynomial kernel. Table 10 shows the classification results for several polynomial degrees:

Polynomial Degree	*Samples (Train / Test)*	*Performance*
1	978 / 972	77.7 %
2	978 / 972	82.3 %
3	978 / 972	84.0 %

Table 10. Classification performance for polynominal kernels of degree 1–3

Other available nonlinear kernels (Sigmoid, RBF) and higher degree polynomial kernels performed very poor.

5. SUMMARY AND OUTLOOK

In this paper a detailed feature relevance study about the classification performance of SVMs for classical guitar sounds is presented. It is shown, that the original feature set of 1155 features with a classification performance of 82.0 % can be reduced to 505 features with an even better performance of 84.0 % when employing a third degree polynomial kernel.

Several experiments on the preselection of sound samples for testing and training have been carried out. A tentative interpretation in musical terms shall be tried in the following paragraphs.

The group of experiments with a pool of players used for training of the SVMs and one player for testing shows, that there is a large influence of the player on the sound. This conclusion can be drawn from the fact, that the classification performance is significantly increased, when the testing player is also member of the training players. From musical experience this is plausible. It is the interaction of player and instrument that produces the sound.

The preselection experiments, where the same timbre, string, and fret is used for training and testing can be explained in a technical way: the more uniform the samples are, the easier is the detection of differences arising from the acoustical properties of the guitars. The very good classification performance for the highest two strings (96.3 %) and 92.6 % is in accordance with the experience of luthiers and guitar players: A good guitar reveals its quality on the treble strings (b and e'), whereas even medium quality guitars may sound good on the lower strings.

It would be a promising approach to improve the overall classification result by introducing a two-step classification process: in the first step the string is determined, and in the second step the guitar is identified. Currently there are experiments going on to compare the reported results with other classification methods, in particular neural networks and a specialised form of principal component analysis to the classification problem presented by Wells and Aldam in [15].

The classification framework is currently being modified to apply to pieces of polyphonic solo guitar music. The robustness of the method has to be proven, when there is a mixture of tones to be analysed, and further features may show up in the musical context, that are not present in the single note analysis, especially the range of possible variations in amplitude, attack and decay times and various spectral properties.

6. REFERENCES

[1] Gnu octave. http://www.octave.org, visited 2011.

[2] J. G.A Barbedo and G. Tzanetakis. Instrument identification in polyphonic music signals based on individual partials. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 401 – 404, 2010.

[3] Jeremiah D. Deng, Christian Simmermacher, and Stephen Cranefield. A Study on Feature Analysis for Musical Instrument Classification. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, 38(2):429–438, 2008.

[4] K. Dosenbach, W. Fohl, and A. Meisel. Identification of Individual Guitar Sounds by Support Vector Machines. In *Proc. of the 11th Int. Conference on Digital Audio Effects (DAFx-08)*, Espoo, Finland, 2008.

[5] Dimitrios Fragoulis, Constantin Papaodysseus, Mihalis Exarhos, George Roussopoulos, Thanasis Panagopoulos, and Dimitrios Kamarotos. Automated classification of piano-guitar notes. *IEEE Trans. on Audio, Speech, and Language Processing*, 14(3), 2006.

[6] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[7] Y. H Hsiao and C. T Su. Multiclass MTS for saxophone timbre quality inspection using waveform-shape-based features. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(3):690 – 704, 2009.

[8] Thorsten Joachims. Svmlight. http://svmlight.joachims.org, 2008.

[9] Cyril Joder, Slim Essid, and Gaël Richard. Alignment kernels for audio classification with application to music instrument recognition. In *Proceedings of the European Signal Processing Conference*, 2008.

[10] I.T. Jolliffe. *Principal Component Analysis*, volume 2. Wiley Online Library, 2002.

[11] R. Loughran, J. Walker, and M. O'Neill. An exploration of genetic algorithms for efficient musical instrument identification. In *Signals and Systems Conference (ISSC 2009), IET Irish*, pages 1–6. IET, 2009.

[12] Ingo Mierswa and Katharina Morik. Automatic feature extraction for classifying audio data. *Machine Learning*, 58(2-3):127–149, 2005.

[13] F. Pachet and P Roy. Analytical features: a knowledge-based approach to audio feature generation. *EURASIP Journal on Audio, Speech, and Music Processing*, 2009(1), February 2009.

[14] Malcolm Slaney. Auditory toolbox. A MATLAB toolbox for auditory modeling work. version 2. http://cobweb.ecn.purdue.edu/ malcolm/interval/1998-010/, 1998.

[15] Jeremy J. Wells and Gregory Aldam. Principal component analysis of rasterised audio for crosssynthesis. In *Proc. Of the 12th Int. Conference on Digital Audio Effects (DAFx-09)*, volume 4, Como, Italy, 2009.

STRING METHODS FOR FOLK TUNE GENRE CLASSIFICATION

Ruben Hillewaere and **Bernard Manderick**
Computational Modeling Lab
Department of Computing
Vrije Universiteit Brussel
Brussels, Belgium
{rhillewa,bmanderi}@vub.ac.be

Darrell Conklin
Department of Computer Science and AI
Universidad del País Vasco UPV/EHU
San Sebastián, Spain
IKERBASQUE, Basque Foundation for Science
Bilbao, Spain
conklin@ikerbasque.org

ABSTRACT

In folk song research, string methods have been widely used to retrieve highly similar tunes or to perform tune family classification. In this study, we investigate how various string methods perform on a fundamentally different classification task, which is to classify folk tunes into genres, the genres being the dance types of the tunes. A new data set *Dance-9* is therefore introduced. The different string method classification accuracies are compared with each other and also with n-gram models and global feature models which have been proven to be useful in previous folk song research. They are shown to yield similar results to the global feature models, but are outperformed by the n-gram models.

1. INTRODUCTION

In the history of Music Information Retrieval (MIR), folk song databases have often been used as test collections to evaluate computational models, especially the Essen Folksong Collection [18] has been the test set for various MIR methods [19, 4]. The availability of large databases of labelled folk tunes and the fact that many of these contain mainly monophonic tunes, make it an attractive test bed for machine learning algorithms applied to musical sequences.

However, there is also a deeper interest in folk music from an ethnomusicological point of view, which is growing with the progression of advanced music data mining methods and the computational possibility of dealing with large folk song corpora. Archives of folk music are being handed over to computational musicologists to be analysed, clustered and subdivided into comprehensible subgroups. A self-organizing map is used to identify and analyse motive collections of 22 folk music cultures in Eurasia [10]. Automatic pattern discovery has been applied to Cretan folk songs, in order to describe the characteristic features of each song type and region [6].

Besides these descriptive tasks, a common task in computational folk music analysis is music retrieval, which is related to the concept of melodic similarity. Symbolic melodic similarity tasks have been proposed at the MIREX contests in 2005, 2006 and 2007, and were reintroduced since 2010, with the Essen folksong database as test set. Most of the methods applied at these contests rely on sequence alignment algorithms, which are shown to be succesful [7, 20, 21].

In this paper, however, we are interested in the the predictive task of folk music classification, where the goal is to predict the class label of an unseen folk tune. Sequence alignment methods have been used for the classification of folk songs into tune families, which are ensembles of tunes that all derive from the same initial tune [22]. It is shown that they outperform global feature approaches. In our previous work, we have shown that n-gram models outperform global feature models for the classification of European folk tunes into their geographic region [8]. Given the conclusions of these previous papers on the topic of folk tune classification, the question arises how sequence alignment methods and more generally string methods compare with n-gram models. We will thus pursue our comparative study by adding this third category of models, in order to shed some light on the existing folk music classification methods and their performance.

This paper investigates the performance of three string methods for the task of genre classification on a newly developed folk tune database *Dance-9*, containing 2198 dances of 9 different types, which we call the genres. String methods rely on a sequential music representation which views a piece as a string of symbols. A pairwise similarity measure between the strings is computed and used to classify unlabeled pieces. Obvious examples of string methods are the sequence alignment methods mentioned above, but also less standard approaches which have been used in the field of text classification, such as compression based techniques [13] or the string subsequence kernel method [12]. Compression based techniques have been applied to music classification [11] and clustering [5], but these methods have not been thoroughly compared with other existing methods for folk tune classification. String subsequence kernels have never been applied to music classification, but relate to the method presented by Pérez-Sancho

Dance type	number of pieces	relative number
Bourrée	59	2.7%
Hornpipe	108	4.9%
Jig	793	36.1%
March	76	3.5%
Polska	339	15.4%
Reel	453	20.6%
Schottische	119	5.4%
Strathspey	123	5.6%
Waltz	128	5.8%
Total	2198	

Table 1. The *Dance-9* collection: the number of pieces of each dance type.

et al. [16], where n-words are used to represent musical pieces as Boolean feature vectors, in order to classify MIDI files into jazz or classical music.

These string methods will be compared with both n-gram models and global feature models which we have studied in depth before [8], and the hypothesis of this study is that n-gram models will outperform both the global feature models and the string methods on the task of folk tune genre classification. It is unclear how the string methods will compare with the global feature models, since this classification task is essentially different than the tune family classification proposed by van Kranenburg [22]. Two folk dances, say for example two random waltzes, generally differ more than two tunes belonging to the same tune family.

The remainder of this paper is structured as follows. In the next section we discuss the data set and its representation that will be used for our experiments, then we describe the three string methods in detail, recapitulate the n-gram models and global feature models, before describing the experimental setup and reporting the results. We conclude with a discussion and future work.

2. DATA SET AND MUSIC REPRESENTATION

In this section we introduce a new folk tune database for our experiments, we illustrate two types of music representation and the features that will be used.

2.1 Data set : *Dance-9*

The corpus *Dance-9* is a large collection of European folk tunes which are subdivided into nine dance type categories, the largest ones being jigs, reels and polskas. An overview of the nine dance types and the class sizes is displayed in Table 1. The associated classification task is to predict the dance type of an unseen tune, which is what we call a *genre* classification task.

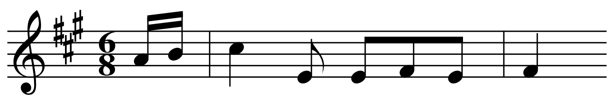
This corpus has been extracted from a much larger collection of approximately 14,000 folk songs transcribed in the ABC format, most of which are available on the web [1]. Many tunes contain metadata about their type of folk dance, and to construct *Dance-9* we only selected those with an unambiguous dance type annotation. Furthermore, we discarded all dance types that occurred insufficiently to have any statistical significance. To the remaining 2198 pieces, two preprocessing steps have been applied in order to end up with core melodies that fit for our research purpose: the first step ensures that all pieces are purely monophonic by retaining only the highest note of double stops which occured in some of the tunes, and in the second step we removed all performance information such as grace notes, trills, staccato, etc. Repeated sections and tempo indications were also ignored. Key and time signature information has been retained, even though they will not be explicitly used as musical features, as we explain in section 3.3. Finally, a conversion to clean quantized MIDI files is carried out with abc2midi. We removed all dynamic indications generated by the style interpretation mechanism of abc2midi.

2.2 Music representation

In MIDI format, the folk tunes are reduced to a list of music events which are specified by their onset time, their pitch and duration. For the purpose of music data mining, one can represent a piece in various ways based on this information, and the chosen music representation is associated to the type of model one intends to use. We will discuss two main types of representation:

- *global feature vector*: a global feature describes an aspect of the whole piece with one single value, such as the average pitch or the fraction of ascending intervals. With a collection of global features, one can represent the piece as a multidimensional feature vector. There is a wide range of standard machine learning techniques available in toolboxes to classify such vectorized data.
- *string representation*: a piece can also be viewed as an ordered sequence of events, and every event is represented by an event feature of one's choice. In our case, the music events are note objects, with pitch and duration as basic event features, from which one can for example derive the melodic interval between the current and the previous note. Other examples are "duration ratio" or "melodic contour". This event feature sequence can be used directly for modelling, or it can first be mapped onto an ASCII symbol string.

Figure 1 illustrates these types of representation on the first measures of the Scottish jig "With a hundred pipers". The two upper lines show two global features "average pitch" and "rel. freq. M2", which is the relative frequency of major seconds. Some event features are illustrated on the next three lines, "pitch" being a basic one from which "melodic interval" is derived. The "interonset interval" tells the time span between the onset times of two successive notes (given in MIDI ticks here), which is similar to note duration, except when there are rests in the piece. The lower two lines show a possible mapping into strings given



average pitch	67.125							
rel. freq. M2	0.714							
pitch	69	71	73	64	64	66	64	66
melodic interval	$\perp$	+2	+2	-9	0	+2	-2	+2
interonset int.	$\perp$	12	12	48	24	24	24	24
int-string	'ffqjfdf'							
ioi-string	'lltiiii'							

Figure 1. Excerpt of the Scottish jig "With a hundred pipers", illustrating the difference between global features, event features and the string representation.

two event features, the melodic interval feature is mapped to "int-string" and the interonset interval to "ioi-string".

Each type of representation allows us to describe the music pieces on different levels of abstraction. When dealing with global features, one can create a large collection of features, each of them capturing information about a different musical aspect. The entire collection will be used to vectorize the pieces, and classification is achieved with standard machine learning algorithms. In the context of event features or the string representation, only one event feature is chosen for modelling, and this choice entirely implies the musical aspect to model and its granularity.

In order to do a fair comparison between the methods, we will only consider features that directly derive from *pitch* on the one hand and *duration* on the other hand, regardless of the used method. More precisely, for the string methods and n-gram models, separate models are built with the event features "melodic interval" and "interonset interval". For the global feature models, we manually created two collections of global features, one containing features derived from the pitches, and the other with features derived from the note durations. Any attributes that make use of other information such as the key or time signature are not included.

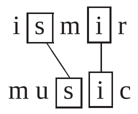
3. METHODS

3.1 String methods

In this section we give a detailed description of the string methods and the implementations that are used in our experiments.

3.1.1 Sequence alignment

The first category of string methods are the sequence alignment methods, which are very common in computational biology to compare protein sequences for example. Alignment algorithms define a similarity measure between two sequences of symbols, by estimating the minimal cost it takes to transform one sequence into the other by means of edit operations, such as substition, insertion and deletion. Therefore, this method is often referred to as "edit distance", which is in fact the Levenshtein distance. For example, the edit distance between the strings 'ismir' and 'music' is equal to 4, since the optimal alignment between them is given by



which means four edit operations are needed: two substitutions ('i' to 'm' and 'r' to 'c'), one insertion (the 'u') and one deletion (the 'm').

More advanced alignment algorithms and alignment scoring mechanisms have been developed depending on the application field. Mongeau and Sankoff [15] were among the first who designed a variant specifically for the alignment of musical sequences. For the purpose of our current research, we have used WEKA's implementation of the edit distance [2]. In a preliminary experiment, we tested this implementation on the melodic interval and interonset interval strings of the exact tune family database used by van Kranenburg [22]. We obtained one nearest neighbour classification accuracies of 94.2% and 80.6% respectively in comparison with his 92.0% and 74.0%, which shows that the general edit distance algorithm is sufficient for our comparative study at hand.

3.1.2 Compression based distance

The second type of string methods are compression based techniques, which also define a distance measure between two strings, by using the concept of information distance inherited from information theory. Ideally, this distance would be represented as

$$d(x, y) = \frac{\max(K(x|y), K(y|x))}{\max(K(x), K(y))},$$

where $K(x)$ is the Kolmogorov complexity of string x, and $K(x|y)$ is the conditional complexity of string x given string y. The underlying motivation behind this distance is to compute how much information is not shared between the two strings relatively to the information that they could maximally share. Since the Kolmogorov complexity $K(x)$ can not be exactly computed, it is approximated by the length of the compressed version of the string using a compressor C, denoted by $C(x)$. The information distance is thus approximated by the normalized compression distance [5]:

$$\text{NCD}(x, y) = \frac{C(xy) - \min(C(x), C(y))}{\max(C(x), C(y))},$$

where xy represents the concatenation of the strings x and y. Various types of compressors can be used to estimate the Kolmogorov complexity, in our experiments we used `bzlib`, which is a block sorting text compression algorithm. For instance, the normalized compression distance between 'ismir' and 'music' is computed as follows:

$$\begin{aligned} C(\text{'ismir'}) &= 344, \\ C(\text{'music'}) &= 336, \\ C(\text{'ismirmusic'}) &= 352, \end{aligned}$$

which are the compressed sizes in bits. So,

$$\text{NCD('ismir','music')} = \frac{352 - 336}{344} = 0.046512.$$

This number represents how different the two strings are, it is generally contained in $[0, 1]$.

3.1.3 *String subsequence kernel*

The third kind of string method is the string subsequence kernel method (SSK), which has been developed for text classification [12]. This approach computes a similarity measure between strings based on the number and form of their common subsequences. Given any pair of two strings, SSK will find all common subsequences of a specified length k, also allowing non-contiguous matches, although these are penalized with a decay factor $\lambda \in (0, 1)$. For example,

$$\text{SSK}(k = 2, \text{'ismir','music'}) = \lambda^5 + \lambda^6,$$

because there are two common subsequences 'si' and 'mi', and the lengths of the matches are the exponents of λ :

	'ismir'		'music'		
	match	l_1	match	l_2	$l_1 + l_2$
'si'	'**<u>s</u>m<u>i</u>**'	3	'**<u>si</u>**'	2	5
'mi'	'**<u>mi</u>**'	2	'**<u>m</u>us<u>i</u>**'	4	6

To speed up the algorithm, one can reduce the search space of subsequences by specifying a maximal exponent m of λ, which is called λ-pruning; it has been shown there is little quality loss due to this pruning. In our experiments, we looked for short subsequences ($k = 2, 3, 5$) allowing few or no non-contiguous matches. The parameter λ was set to a default value of 0.5.

The general idea behind these three string methods is to determine a similarity measure between two "stringified" music pieces x and y. A similarity measure between strings is either derived from a distance metric $d(x, y)$, such as the edit distance or the normalized compression distance (NCD), or else it is computed directly from the strings, which is what the string subsequence kernel (SSK) does. Given a distance metric $d(x, y)$, one can simply use a nearest neighbour approach to classify unseen test pieces, replacing the usual Euclidean distance with $d(x, y)$. In the case of the string subsequence kernel, the computed similarity measure is considered as the kernel function of a support vector machine, a state of the art classifier that learns non-linear decision boundaries between classes.

3.2 n-gram models

In this section we briefly recall how an n-gram model can be employed for classification of music pieces, for more details we refer to our previous work [8]. In a first stage, every piece of the music data is transformed into an event feature sequence according to a feature of choice. In the training phase, for each class the n-grams are counted to estimate the probability distribution of the musical "words" in that particular class. Given a test piece represented by its event feature sequence, the piece probability is then computed as the joint probability of the individual events in the piece according to the learned distribution, with the assumption that the probability of an event only depends on the $n-1$ previous events. Finally, the test piece is assigned to the class with the highest piece probability, which is the most likely to have "generated" the piece.

Note that the music representation is basically the same as for the string methods, but the essential difference between these methods is that an n-gram model aims to model the transitions for a given class, whereas a string method computes a pairwise similarity measure between pieces.

3.3 Global feature models

In this section, we describe what global features were chosen for our experiments. Two separate global feature sets were made, with features derived from the pitch on the one hand and from the note durations on the other hand. The features were chosen among the following (see Table 2) :

- The *Alicante* set of 28 global features, proposed by P.J. Ponce de Léon and J.M. Iñesta in [17] to classify a collection of 110 MIDI tunes in the genres jazz and classical. Among these, 7 are derived from pitch, e.g. "average melodic interval" and 12 from duration, like "duration range".

- The *Jesser* set, containing 39 statistics designed by B. Jesser [9], 31 of which are pitch-based features. Most of these are basic relative interval counts, like "dminthird", measuring the fraction of descending minor thirds, for all ascending and descending intervals in the range of the octave. This set also includes 6 features derived from the note durations.

- The *McKay* set of 101 global features [14], which were used in the winning 2005 MIREX symbolic genre classification experiment and computed with McKay's software package jSymbolic [3]. This set is composed of a wide range of features, since it was intended to classify orchestrated MIDI files. We retained 34 features based on pitch, for example "Direction of motion", i.e. the fraction of melodic intervals that are rising rather than falling, and 4 based on duration.

All features derived from pitch were joined to obtain a set of 73 features, since there are not many overlapping features. With the same procedure applied to the duration features a set of 22 features was formed. We recall that any features derived from the meter or the key signature have not been retained, since we want to compare the methods and representations on the basis of the same information.

Global feature set	pitch	duration
Alicante	8	12
Jesser	31	6
McKay	34	4
Total	73	22

Table 2. Global features that were selected for our experiments, divided into those derived from pitch and those from duration.

Since global features represent every instance as a multidimensional feature vector, any standard machine learning classifier can be applied to get a performance accuracy.

4. RESULTS

In this section we describe the experimental setup and the classification results on the folk tune data set *Dance-9*. Since we are interested in the relative performance of the string methods, the n-gram models and the global feature models, we have computed 10-fold cross validation classification accuracies for each of the methods. Care has been taken to use the exact same cross validation folds in all experiments, and the classifier parameters (if applicable) have always been set to standard values to do an unbiased comparison between the methods.

The string methods edit distance and NCD have been evaluated using a one nearest neighbour approach (1NN), whereas SSK implies one works with a support vector machine. Different lengths of short subsequences have been examined ($k = 2, 3, 5$), and it was found that the best performances were obtained with contiguous matches; only those will be displayed. For the n-gram models, we constructed trigram and pentagram models, in direct comparison with the SSK method. The global feature vectors have been classified with nearest neighbour approaches as well as with a regular SVM kernel with a Radial Basis kernel Function (RBF). For all experiments with SVM, the parameter determining the softness of the decision boundary has been set to its default value, after verifying this does not penalize any of the methods.

The results are reported in Table 3, which have to be compared to a baseline classification accuracy of 36.1% one obtains by always choosing the largest class “Jig”. The first column contains the results using only features related to pitch or melodic interval sequences, whereas the second column gives the results with the duration features and interonset interval sequences. It appears immediately that the latter leads to superior classification accuracies regardless of the method, with a difference of approximately 20% on average. This shows that the recognition of folk dance types on this corpus is easier to achieve with the duration representations than with melodic ones, which is not surprising since folk dances are commonly distinguished by their rhythmic patterns.

SSK appears to be the most powerful string method, especially with contiguous subsequences of length $k = 3$. However, when we increase the length to $k = 5$ the performance drops when using the melodic interval strings. NCD does not lead to any promising result, whereas the edit distance does reasonably well, especially if one keeps in mind its computation time is a lot shorter than for SSK.

String methods	melodic int.	interonset int.
EditDist (1NN)	50.0	70.0
NCD (1NN)	48.0	68.0
SSK ($k = 2$, $m = 4$)	54.0	71.2
SSK ($k = 3$, $m = 6$)	60.8	72.9
SSK ($k = 5$, $m = 10$)	38.4	68.9
n-gram models	melodic int.	interonset int.
$n = 3$	60.7	71.9
$n = 5$	**66.1**	**76.1**
Global feature models	pitch	duration
1NN	40.3	66.9
5NN	44.8	69.3
SVM, RBF-kernel	53.5	67.7

Table 3. The 10-fold cross validation classification accuracies with all methods using the interval and duration representation.

The comparison across all the methods reveals that the pentagram model clearly outperforms the other approaches, with both the melodic interval and the interonset interval features. The trigam model also outperforms most other methods with both representations, except for SSK with $k = 3$ that achieves very similar results. On this corpus, the string methods and global feature models yield similar results with the melodic features, but on the rhythmic features there is a slight advantage for all the string methods except NCD. For the global feature models, the SVM with RBF-kernel performs better than both nearest neighbour models with the melodic features, but with the rhythmic features there is no benefit in using the more sophisticated SVM classifier over the simple nearest neighbours approach.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we thorougly examined the classification performances of three string methods and compared them with well-known other classification methods on a large folk dance dataset with nine classes. We have described the difference between the underlying types of music representation and features, and have shown that features based on duration lead to better classification models than features based on pitch, no matter if they are used to represent the music with an event feature sequence or string, or with a collection of global features.

The comparison between the methods has revealed that the n-gram models outperform both the string methods and the global feature models, which is in agreement with our earlier survey [8]. This result proves the effectiveness of modelling the transitions within a musical sequence and supports our hypothesis that the n-gram model should be the default model for folk tune classification.

However, the string methods generally perform slightly better than the global feature models, particularly with the string subsequence kernel which obtains the highest accuracies among the string methods. This first result on music classification with the string subsequence kernel is encouraging for future work. The alignment methods which have been shown to be efficient in tune family classification [22] cannot measure up to the pentagram model on this genre classification task. We are currently doing more research on other folk song databases to get a broader view of the alignment method performance. In particular we are interested in discovering which models are most effective with respect to the precise classification task at hand.

6. REFERENCES

[1] `http://trillian.mit.edu/~jc/cgi/abc/tunefind`.

[2] `http://www.cs.waikato.ac.nz/ml/weka/`.

[3] `http://jmir.sourceforge.net/jSymbolic.html`.

[4] W. Chai and B. Vercoe. Folk music classification using hidden Markov models. In *Proceedings of International Conference on Artificial Intelligence*, Seattle, Washington, USA, 2001.

[5] R. Cilibrasi and P. Vitányi. Clustering by compression. *Information Theory, IEEE Transactions on*, 51(4):1523–1545, 2005.

[6] D. Conklin and C. Anagnostopoulou. Comparative pattern analysis of Cretan folk songs. *Journal of New Music Research*, 40(2):119–125, 2011.

[7] C. Gómez, S. Abad-Mota, and E. Ruckhaus. An analysis of the Mongeau-Sankoff algorithm for music information retrieval. In *Proceedings of the 8th International Conference on Music Information Retrieval*, pages 109–110, Vienna, Austria, 2007.

[8] R. Hillewaere, B. Manderick, and D. Conklin. Global feature versus event models for folk song classification. In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, pages 729–733, Kobe, Japan, 2009.

[9] B. Jesser. *Interaktive Melodieanalyse*. Peter Lang, Bern, 1991.

[10] Z. Juhász. Motive identification in 22 folksong corpora using dynamic time warping and self organizing maps. In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, pages 171–176, Kobe, Japan, 2009.

[11] M. Li and R. Sleep. Melody classification using a similarity metric based on Kolmogorov complexity. In *Sound and Music Computing Conference*, Paris, France, 2004.

[12] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *The Journal of Machine Learning Research*, 2:419–444, 2002.

[13] Y. Marton, N. Wu, and L. Hellerstein. On compression-based text classification. In *Advances in Information Retrieval*, volume 3408 of *Lecture Notes in Computer Science*, pages 300–314. Springer Berlin / Heidelberg, 2005.

[14] C. McKay and I. Fujinaga. Automatic genre classification using large high-level musical feature sets. In *Proceedings of the 5th International Conference on Music Information Retrieval*, pages 525–530, Barcelona, Spain, 2004.

[15] M. Mongeau and D. Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24(3):161–175, 1990.

[16] C. Pérez-Sancho, J. M. Iñesta, and J. Calera-Rubio. Style recognition through statistical event models. *Journal of New Music Research*, 34(4):331–339, 2005.

[17] P. J. Ponce de Léon and J. M. Iñesta. Statistical description models for melody analysis and characterization. In *Proceedings of the 2004 International Computer Music Conference*, pages 149–156, Miami, USA, 2004.

[18] H. Schaffrath. The Essen folksong collection in the Humdrum Kern format. *Stanford, California: Center for Computer Assisted Research in the Humanities*, 1995.

[19] P. Toiviainen and T. Eerola. A method for comparative analysis of folk music based on musical feature extraction and neural networks. In *3rd International Conference on Cognitive Musicology*, pages 41–45, Jyväskylä, Finland, 2001.

[20] A. L. Uitdenbogerd. N-gram pattern matching and dynamic programming for symbolic melody search. *Proceedings of the Third Annual Music Information Retrieval Evaluation eXchange*, 2007.

[21] J. Urbano, J. Lloréns, J. Morato, and S. Sánchez-Cuadrado. Mirex 2010 symbolic melodic similarity: Local alignment with geometric representations. *Music Information Retrieval Evaluation eXchange*, 2010.

[22] P. van Kranenburg. A computational approach to content-based retrieval of folk song melodies. *SIKS dissertatiereeks*, 2010(43), 2010.

A TURKISH MAKAM MUSIC SYMBOLIC DATABASE FOR MUSIC INFORMATION RETRIEVAL: SymbTr

M. Kemal Karaosmanoğlu
Yıldız Technical University
kkara@yildiz.edu.tr

ABSTRACT

Turkish makam music needs a comprehensive database for public consumption, to be used in MIR. This article introduces *SymbTr*, a Turkish Makam Music Symbolic Representation Database, aimed at filling this void. *SymbTr* consists of musical information in text, PDF, and MIDI formats. Raw data, drawn from reliable sources, and consisting of 1,700 musical pieces in Turkish art and folk music was processed featuring distinct examples in 155 diverse *makam*s, 100 *usul*s and 48 *form*s. Special care was devoted to selection of works that scatter across a broad historical time span and were among those still performed today. Total number of musical notes in these pieces was 630,000, corresponding to a nominal playback time of 72 hours. Synthesized sounds particular to Turkish makam music were used in MIDI playback, and transcription/playback errors were corrected by input from experts. Symbolic representation data, open to the public, is output from a computer program developed exclusively for Turkish makam music. *SymbTr* was designed as a wholesome representation of aforementioned distinct auditory and visual features that distinguish Turkish makam music from other music genres. This article explains the database format in detail, and also provides, through examples, statistical information on pitch/interval allocation and distribution.

1. INTRODUCTION

Turkish makam music is a genre drawing roots from a thousand year old tradition, featuring distinct melodic patterns called *makam* and rich rhythmic structures called *usul*. Since the number of tones per octave is greater in Turkish makam music, compared to Western music, several sharp and flat accidentals appear in printed scores. Additionally, one must take into consideration a multitude of idiosyncratic rhythmic structures. Although there exists only one version of the score, independent of the instrument or key, musicians perform improvised transpositions during performance, as permitted by the ranges of their instruments and the vocalist on hand. Probably the most prominent feature of Turkish makam music is its monophonic ─and incidentally heterophonic─ structure. Another characteristic is the number of notes in an octave: 17, 24, and, according to some musicologists, even a greater number of tones to the octave make up the pitch palette of Turkish makam music [12], [16]. Although displaying a higher pitch count compared to Western music, there is no one-to-one correlation between the fixed frequency values, music theory, implied in engraved scores and what is actually performed in practice [3].

Everything mentioned up to this point was to differentiate Turkish makam music from many other world music genres. It then follows; data structures and algorithms developed for other musical traditions are not directly applicable to Turkish makam music. On the other hand, there are only a handful of researchers working on computational models for Turkish makam music. There remains much to be done in areas related to data collection/compilation, algorithm development, and research. *SymbTr* is hopefully a likely candidate to be a pioneer in the field, since it is capable of accommodating and expressing information specific to makam music. Secondly, early studies ([9], [18]) have returned encouraging results. It is anticipated that *SymbTr* might provide a setting for scholars interested in makam music, potentially

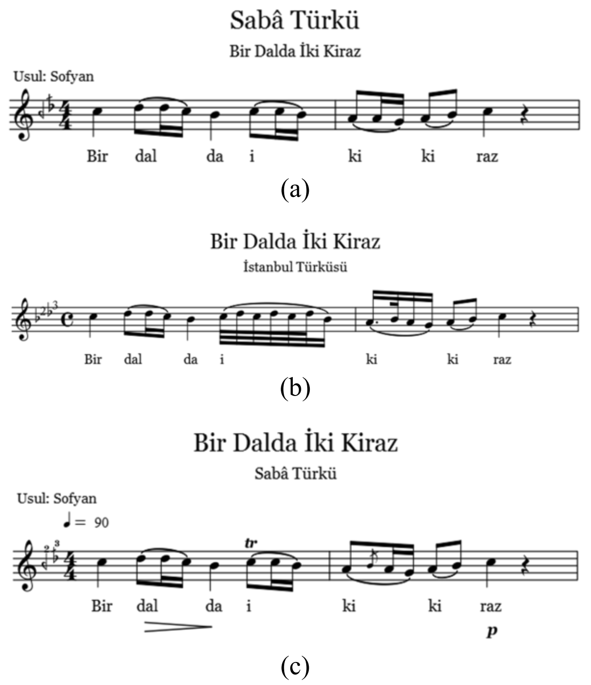


Figure 1. A Turkish folksong's scoring in (a) KTM, (b) THM, and (c) mixed format

stimulating further research at a global level.

2. STYLES OF TURKISH MAKAM MUSIC

Turkish makam music is viewed under two major headings:

1. Classical Turkish music (KTM),
2. Turkish folk music (THM).

Since both styles originate from the same cultural roots, their modal motifs and rhythmic structures are very similar in character [17]. Owing to political movements emerging at the turn of the 20th Century, a superficial bifurcation took place, which led to a divergence between the two styles resulting in two separate traditions. Today, these two traditions differ considerably when it comes to their respective theoretical models, notation systems, and terminology. Fig. 1 shows the first two measures from a folksong score, which is part of both the THM and KTM repertoires of TRT [1]. Scores are shown in three different notational systems.

As can be detected in the scores, accidental symbols, in particular, are different even though the melody is essentially the same: In the KTM version reverse and hooked flat signs (Fig. 1-a) represent the accidentals, while in the THM version superscripts over ordinary flat signs are used (Fig. 1-b). The mixed notation in (Fig. 1-c) displays a combination of the two. Reverse and hooked flats, by definition, lower a note by 1 and 4 Holdrian commas (Hc) [2] respectively. However, many of the measurements ([1], [4], [5]) evince that, printed scores for works in the Saba makam should carry a 2 comma flat sign for B and a 3 comma flat sign for D as the key signature. Indeed, these values are substituted in THM and mixed notations.

The difference between KTM and THM notation lies not only in the symbols representing the accidentals. In KTM notation there are almost no ornamentation symbols on the score. In THM, on the other hand, ornamentation is achieved by repeated use of notes with smaller rhythmic values, as shown in the trill in Fig. 1-b [17]. When such passages are converted into *SymbTr*, note clusters representing ornamentation are indicated by a single *core* note, with its type shown in the *Code* field.

Because of the fact that THM and KTM have mixed and intertwined traditionally, the *SymbTr* database naturally accommodates pieces from both categories. Format in the database was, therefore, designed to reconcile the artificial disparity between the two traditions. The most important design element for the database was the fundamental tuning selected. The Arel - Ezgi (AE) tone-system, which has been recognized and widely adopted as the official KTM system since the 1950s, has 24 notes in an octave. In contrast, THM has adopted a notation with 17 notes to the octave. Twelve of the said 17 notes are common with the AE system. Moreover, both tonal scales present a near-perfect subset of 53 tone equal temperament (53TET), with deviations less than 1 cent [19] (Fig. 2). Possibly due to this structural connection, Turkish makam music education has been built around 53TET, whether acknowledged by name (Ayomak, Sarısözen) or

THM	53TET	KTM
	Cb1	Dikbûselik
	B#2	
	B#1	
Si	B	Bûselik
	Bb1	Segâh
Si b2	Bb2	
	Bb3	
	Bb4	Dikkürdî/sünbüle
Si b	A#4	Kürdî/Sünbüle
	A#3	
	A#2	
	A#1	
La	A	Dügâh/Muhayyer
	Ab1	Dikzirgüle/şehnaz
La b2	Ab2	
	Ab3	
	Ab4	Zirgüle/Şehnaz
La b	G#4	Nîmzirgüle/şehnaz
	G#3	
	G#2	
	G#1	
Sol	G	Rast
	Gb1	Dikgeveşt/mâhur
	Gb2	
	Gb3	
Sol b4	Gb4	Geveşt/Mâhur
	F#4	Irak/Eviç
Fa #3	F#3	
	F#2	
	F#1	Dikacem/aşîran
Fa	F	Acem/aşîran
	E#3	
	E#2	
	E#1	
Mi	E	Hüseynî/aşîran
	Eb1	Dikhisar
Mi b2	Eb2	
	Eb3	
	Eb4	Hisar
Mi b	D#4	Nîmhisar
	D#3	
	D#2	
	D#1	
Re	D	Yegâh/Nevâ
	Db1	Dikhicaz
	Db2	
	Db3	
Re b4	Db4	Hicaz
	C#4	Nîmhicaz
Do #3	C#3	
	C#2	
	C#1	
Do	C	Çargâh

Figure 2. 17 tones in THM (left column), 24 tones in KTM (right column), and 53TET in between

[1] Turkish Radio/TV Broadcasting Corporation.

[2] Interval unit obtained by the division of the octave into logarithmically equal 53 parts: $Hc = 1200 / 53 \approx 22.5$ *cent*. In this article comma signifies the Holdrian comma.

not [13]. Hence, the term "comma", when describing *makam*s and preparing printed scores, refers to the Holdrian comma as the basic intervallic unit, obtained by equally dividing the octave into 53 equal parts. Selecting 53TET as the master underlying tuning in *SymbTr* also facilitates transpositions across *ahenk*s (pitch-levels). *Ahenk*s can be defined as 7 principal and 5 minor categories corresponding to 12 chromatic pitch-levels akin to what key transposing instruments of Western music accomplish. Detailed information about *ahenk*s and Turkish makam music in general can be found in [10] and [14].

3. MAKAM MUSIC AND SYMBOLIC DATA

SymbTr database is generated by using the output from a computer program *Mus2-Alpha*, developed by the author of this article. This software is the first notation and playback application for Turkish makam music to the best of our knowledge. All pieces in the database were entered manually using the said software. Printed scores and MIDI files were, then, prepared for every piece in the database. Initially, before the introduction of *Mus2-Alpha* and its sister applications (*Nota 2.2* [1], *Notist* [2]), scores were engraved either manually or using programs such as *Finale* or *Sibelius*, that were developed solely to transcribe Western music. Since these programs were not designed to notate flats and sharps specific to Turkish makam music, their standard output formats such as MusicXML and MIDI have not been useful in research on Turkish makam music [7].

The format for *SymbTr* described in this article was derived from *Mus2-Alpha*'s original format that was used initially to transcribe printable sheet music for pieces in Turkish makam music. Since this format includes reprise markings such as *segno* and *coda*, some modifications for scientific research are necessary. In *SymbTr*, notes are linearized just as they are performed. An advantage associated with *Mus2-Alpha* originating data is that pieces can be amended through consultation with experts, using listening tests based on synthesized sound output. An entry level version of this program, *Mus2okur* [3], has reached thousands of users, thereby resulting in a wide scale screening of possible errors in the database.

The main source of data in *SymbTr* is TRT and other trustworthy archives (Recollection of Turkish Music Culture [4]), where almost all of them were entered using the AE notation. To synthesize realistic intonations, however, it was necessary to use pitches not included in the AE tone-system. Five notes in the THM scale lie outside the AE scale (Fig. 2). As a courtesy for Turkish musicians, a composite system was adopted in the printout scores of *SymbTr*: Symbols for flats and sharps were taken directly from AE, and numerical superscripts were inserted to express comma-alterations for notes that were not available in the tone-system (Fig. 1-c).

Code	Note53	Comma53	NoteAE	CommaAE	Num.	Denom.	ms	LNS	VelOn	Syllable
9	Do5	318	C5	318	1	4	667	95	96	Bir
9	Re5b3	324	D5b4	325	1	8	333	99	108	dal
9	Re5b3	324	D5b4	325	1	16	167	99	96	
9	Do5	318	C5	318	1	16	167	95	84	
9	Si4b2	312	B4b1	313	1	4	667	95	72	da
12	Do5	318	C5	318	1	8	333	99	96	i
9	Do5	318	C5	318	1	16	167	99	96	
9	Si4b2	312	B4b1	313	1	16	167	95	96	
9	La4	305	A4	305	1	8	333	99	96	ki
8	Si4b2	312	B4b1	313	1	8	42	99	96	
9	La4	305	A4	305	1	16	167	99	96	
9	Sol4	296	G4	296	1	16	167	95	96	
9	La4	305	A4	305	1	8	333	99	96	ki
9	Si4b2	312	B4b1	313	1	8	333	95	96	
9	Do5	318	C5	318	1	4	1334	45	84	raz

Table 1. *SymbTr* representation of the score in Fig. 1.c

4. SymbTr FORMAT

Basic information such as *makam*, *form* and *usul* related to each piece in *SymbTr* is indicated in the filename. In this manner, any piece can be accessed directly from the file system:

$$\underbrace{\text{beyati}}_{\textit{Makam}}\text{--}\underbrace{\text{sarki}}_{\textit{Form}}\text{--}\underbrace{\text{aksak}}_{\textit{Usul}}\text{--}\underbrace{\text{karsidan\_yar}}_{\textit{Title}}\text{--}\underbrace{\text{dede\_efendi}}_{\textit{Composer}}\text{.txt}$$

Some fields in the *SymbTr* format consist of different representations of the same information. Therefore, one field can be easily converted into the other with the help of the relevant computer code. However, since this additional information requires very little extra storage space, it is provided separately for the convenience of researchers. These basic and readily derived fields are described under common headings below.

Code: Signifies a normal note (#9) or ornamentation. The most commonly used ornamentation codes are as follows: #7 for tremolos, #8 for acciaccatura, #12 for trills, and #23 for mordent.

NoteAE / **CommaAE**: A kind of scientific pitch notation [20]: Indicates note letter, its octave (for example, *G5* for *gerdaniye*), and its comma equivalent (349) (Fig. 3). Notes in THM sheets that do not exist in the AE system are represented by their closest equivalent AE note, e.g. *Mi b2 = Dikhisar* (*Eb1*) (Fig. 2). *C4* is the

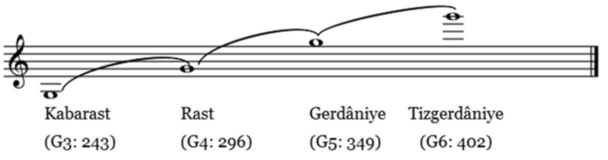


Figure 3. Triple octave operational range of *SymbTr* database and some of the comma numbers

[1] http://www.tulgan.com/Nota22/

[2] http://notist.org/

[3] www.musiki.org

[4] www.sanatmuziginotalari.com/ under http://devletkorosu.com
Please go to the second link 'http'first to reach the main site. Then, look for and click on the first address 'www'.

note with the frequency of about 262 Hz and numbered as 60 in the MIDI standard. All notes excluding *C*'s have a fractional *MIDI Nr*. The *MIDI Nr* corresponding to *CommaAE* can be computed by the following formula:

$$MIDI\_Nr = \frac{CommaAE \cdot Hc}{100} \qquad (1)$$

In order to represent flats and sharps in *Hc* units, the "b" and "#" prefixes were used respectively. For example, the *segah* note in AE tone-system is represented as *B4b1*; since, according to *AE* theory, it should sound one comma lower than the natural *B* (*Si - buselik*). Its comma equivalent is 313, and *MIDI Nr* is 70.87.

Note53 / **Comma53**: Indicates the code and the value of the note in 53TET. If there is no difference between the performance and the sheet music, *CommaAE* and *Comma53* values are the same. However, in some *makam* sequences such as *Uşşak*, *Hüzzam*, *Saba* and *Karcığar*, these two values often vary. For example, in some makams the pitch that corresponds to *B4b1* in AE is *Si4b2* in 53TET, since, in practice, this note should sound 2 commas lower then *Si* (*B*). Its comma equivalent is 312, and *MIDI Nr* is 70.64.

Numerator / **Denominator** and **ms**: Stands for the rhythmic value of the note, with its duration measured in milliseconds. When the tempo (quarter note beats per minute) of the piece is known, these two values can be converted to each other by the following formula:

$$ms = \frac{240\,000}{Tempo} \cdot \frac{\text{Numerator}}{\text{Denominator}} \qquad (2)$$

In Turkish makam music, changes in the tempo of a piece is a run-of-the-mill situation (e.g., the 4th section of sazsemaisi pieces are performed faster than other sections), and since the database can be used for rhythmic analysis purposes [9], it was found useful to enter these two strands of information in the same record.

LNS (Legato / Normal / Staccato): Indicates how tied or detached the notes are to be played. This information is extracted by listening to performances in synch with verses and syllables in the lyrics. The default value is *95*; that is, the last 5% of the duration time for normal notes is completed with silence. *50* means playback should be of *staccato*. Rest signs are determined using this value.

VelOn: Indicates the volume or strike of the note, making nuanced performance possible. Turkish makam music scores ordinarily do not contain dynamics markings like *piano* or *crescendo*. In *SymbTr* an attempt has been made to compensate, as much as possible, for this deficiency.

Syllable1: Indicates the syllable corresponding to a note. There is one space character at the end of the syllables that occur at word endings and two space characters at the end of the verses. This information was added to facilitate the tracking of the melody, as well as for its utility in studies of lyrics-based analyses [8]. In instrumental pieces, it is used to represent the beginning of sections such as "TESLİM", etc… In other places this field is left blank. Instrumental parts of vocal pieces contain a series of dots in this field. In the original *Mus2-Alpha* database, repetitive passages have a separate field for the second syllable. However, due to copyright considerations there is only one field in *SymbTr*.

The representation of the score of Fig. 1-c in *SymbTr* is listed in Table 1. The data starts immediately after the column headings. Fields are tab-delimited.

5. MAKAM MUSIC AND MIDI

It is impossible to produce makam music intonations using ordinary MIDI messages. Therefore, it becomes necessary to use pitch-bend techniques. To generate the needed feature, a pitch-bend message must be sent with the same delta-time value as the note, just before the 'Note on' message. The pseudo-MIDI messages for the first 5 notes in Fig. 1-c are as follows:

Delta Time	Pitch Bend	Note On
0	7 960	C4
4	9 429	D5b
2	9 429	D5b
1	7 960	C4
1	6 492	B

The anchor note is A (La). Therefore, pitch-bend is unnecessary for any A in all octaves. Bend is required for all other pitches. For example, the A – C interval is 13 Hc wide. This value is up to 5.7 cents narrower than the 12TET minor third. Taking into account that 100 cents = 4096 pitch bend units, bending for C is calculated as follows: *8 192 – 5.7 · 40.96 ≈ 7 960.*

MIDI files in *SymbTr* database are not for listening to music. They are included, so that the researchers may find it useful to hear the tune in its simplest *raw* form. To this end, even the instrument information has not been added. Voicing is done with the default MIDI instrument.

6. SOME STATISTICS

SymbTr has been created mainly for the purpose of education and scientific research, and hence, endeavored to be as rich as possible in the diversity of *makam*s, *form*s,

Nr.	Makams	# of Pieces	Usuls	# of Pieces	Forms	# of Pieces
1	Hicaz	118	Sofyan	251	Şarkı	677
2	Rast	88	Aksak	246	Türkü	285
3	Nihavent	85	Düyek	143	Seyir	169
4	Uşşak	85	Aksaksemai	101	Küpe	120
5	Segah	74	Curcuna	91	Peşrev	74
6	Hüseyni	72	Ağıraksak	83	Aranağme	72
7	Hüzzam	65	Yürüksemai	75	Sazsemaisi	66
8	Mahur	54	Nimsofyan	74	İlahi	32
9	Kürdilihicazkar	51	Semai	69	Yürüksemai	27
10	Muhayyer	51	Senginsemai	54	Beste	23

Table 2. The most used 10 *makam*s, *usul*s, and *form*s in *SymbTr*

Nr	AE Name	AE Code	Quantity %	Duration %
1	Neva	D5	16.1%	16.1%
2	Çargah	C5	11.0%	10.7%
3	Hüseyni	E5	9.4%	9.7%
4	Gerdaniye	G5	8.5%	9.1%
5	Dügah	A4	8.5%	7.9%
6	Segah	B4b1	6.9%	6.6%
7	Acem	F5	5.4%	5.6%
8	Muhayyer	A5	4.8%	5.3%
9	Eviç	F5#4	4.7%	5.0%
10	Rast	G4	4.0%	3.6%
11	Nimhicaz	C4#4	2.9%	2.8%
12	Hisar	E5b4	1.9%	1.9%
13	Kürdi	B4b5	1.8%	1.7%
14	Dikkürdi	B4b4	1.6%	1.5%
15	Buselik	B4	1.4%	1.4%
16	Nimhisar	D5#4	1.2%	1.2%
17	Dikhisar	E5b1	1.0%	1.0%

Table 3. The most commonly used 17 pitches

*usul*s, and so on. There are many examples such as *seyir* composed for educational purposes. One criterion in the selection of pieces has been music lovers' familiarity with them, as to whether a piece be average or above-average. We did not adopt random sampling (as in [2], [11], and [15]) as proper methodology when one considers 80% of the twenty five thousand pieces in the TRT repertoire have hardly ever been performed or have become obsolete. A musical piece, composed but almost never performed cannot be held equivalent to one widely known and frequently performed.

Some statistics about SymbTr as follows:

Total number of pieces: 1 700
Number of notes: ~ 630 000
Classical: 1 400
Folk: 300
Vocal pieces: 1 295
Instrumental pieces: 405
Religious: 49
The number of distinct *makam*s: 155
The number of distinct *usul*s: 100
The number of distinct *form*s: 48.

Interval (Hc)	Name, Direction	%
-9	Whole Tone (*Tanini*), descending	18.1
0	Unison	15.4
-5	Apotome (*Küçük Mücennep*), desc.	12.5
9	Whole Tone (*Tanini*)	11.4
5	Apotome (*Küçük Mücennep*)	8.6
-4	Limma (*Bakıyye*), desc.	6.3
-8	Minor Whole Tone (*B. Mücennep*),	4.2
4	Limma (*Bakıyye*)	4.1
8	Minor Whole Tone (*Büyük Mücennep*)	2.6
13	Augmented Second	2.1
-12	Augmented Second, desc.	2.0
-13	Augmented Second, desc.	1.8
22	Perfect Fourth	1.6

Table 4. The most commonly used 13 AE intervals

Highest ranking 10 *makam*s, *usul*s, and *form*s are shown in Table 2.

7. PITCHES AND INTERVALS

Of all the pitches in the database, 17 that are used over 1 per cent in quantity and duration are listed in descending order in Table 3. Percentages in quantity and duration exhibit slight variances but these do not affect the ranking.

Figure 4 shows a histogram of these pitches in the two octave range between *yegah* (D4: 274) - *tizneva* (D6: 380) using the note codes as given in Table 3.

It is interesting to note that 9 pitches in the 3 octave range (Fig. 3) have never been used (for example, *kabahicaz*, and *kabadikhicaz*). When we excluded the notes that were heard for less than one thousandth of the time, only 33 pitches remained, whereas there were 72 pitches defined in this range in the AE tone-system. These observations seem to support Can's results [6].

The most commonly used 13 AE intervals and their usage as quantity in percentages are listed in Table 4.

The SymbTr database can be accessed at the following address, open for public consumption:

http://compmusic.upf.edu

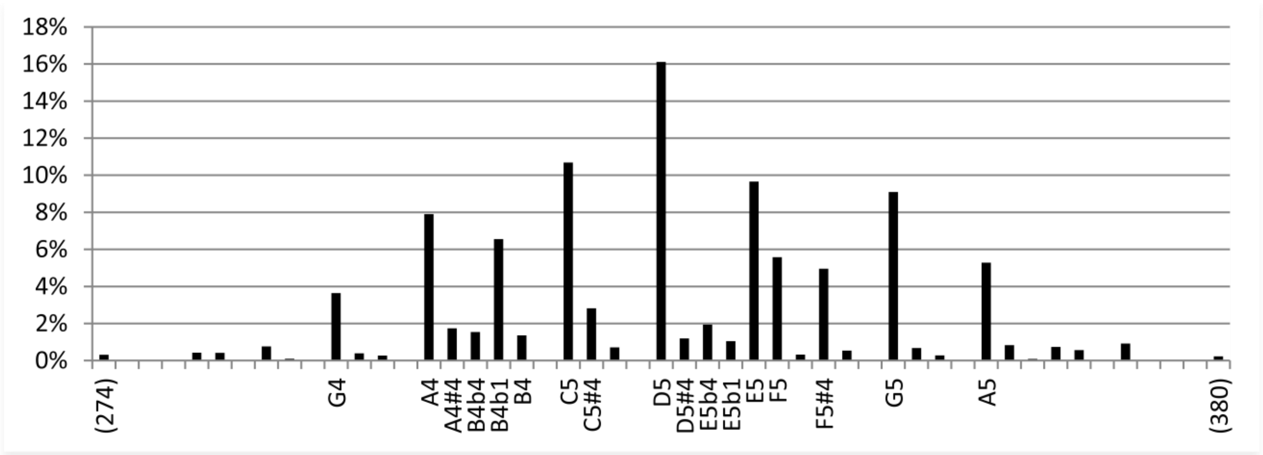


Figure 4. Usage of the notes in *SymbTr* as durations in percentages

8. SIMILAR DATASETS

In this article, we announce the availability of a new database called *SymbTr*, the most extensive machine readable database for Turkish makam music currently available. There is only one other compilation that would qualify to be called a database: the recently launched *TSM Corpus* [2] (TÜBİTAK [1] ref. is PN: 110K040) consisting of symbolic data that relate to 600 pieces. These two databases are far from adequately representing Turkish makam music. New data, however, is being continually added to the *SymbTr* database through various projects. In addition, *Mus2* (Turkish makam and microtonal music notation program) [2], which is still being marketed commercially, can produce output in the *SymbTr* format.

TSM Corpus project, supported by TÜBITAK, can be quite useful. However, the following deficiencies in database design need to be resolved:

- Presence of data belonging to various pieces in a single Excel format file makes usage difficult,
- Syllabized lyrics are not included in the database,
- Tempo information for musical pieces is not provided. Only one quantization information is included concerning durations: *1/4 meter note = 100 units*. This is a serious drawback for musical pieces that require, in particular, the inclusion of tempo and / or *usul* modulations throughout,
- It is not specified which engraved score variant is employed when entering symbolic data.

9. DISCUSSION

If MIR community members at large run their applications on the *SymbTr* database, making necessary small changes, it may lead to two-way improvements: Mysteries of makam music may be unraveled on a grand scale, at a global setting while scholars keep tapping into new structures and patterns, thus moving into uncharted territories of human cognition.

10. ACKNOWLEDGEMENT

This research was partly funded by the European Research Council under the European Union's Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583).

11. REFERENCES

[1] C. Akkoç, "Non-deterministic scales used in traditional Turkish music", Journal of New Music Research, vol. 31, no. 4, 2002.

[2] N.B. Atalay, S. Yöre: *Türk Sanat Müziği derlemi* [Data File]. Retrieved from www.tsmderlemi.com, Konya, 2011.

[3] G. Ay, L.B. Akkal: İTÜ Türk Musikisi Devlet Konservatuarı Türk müziğinde uygulama - Kuram sorunları ve çözümleri - *Uluslararası çağrılı kongre bildiriler kitabı*, İstanbul, 2009.

[4] R. Ayangil: *Türk makam müziği perdelerini çalabilen piyano imali projesi*. Yıldız Teknik Üniversitesi, İstanbul, 1998.

[5] B. Bozkurt, O. Yarman, M.K. Karaosmanoğlu, C. Akkoç: "Weighing Diverse Theoretical Models on Turkish Maqam Music Against Pitch Measurements: A Comparison of Peaks Automatically Derived from Frequency Histograms with Proposed Scale Tones", *Journal of New Music Research*, 2009.

[6] M.C. Can: "Geleneksel Türk Sanat Müziğinde Arel-Ezgi-Uzdilek Ses Sistemi ve Uygulamada Kullanılmayan Bazı Perdeler", *Gazi Üniversitesi, Gazi Eğitim Fakültesi Dergisi*, Ankara, 2001.

[7] M.C. Can: *Müzikte bilgisayar destekli istatistiksel analiz*, Ankara, 2004.

[8] H.T. Cheng et al. "Multimodal Structure Segmentation and Analysis of Music Using Audio and Textual Information". *EEE International Symposium,* 2009.

[9] A. Holzapfel, Y. Stylianou: "Rhytmic Similarity in Traditional Turkish Music". *ISMIR 2010* <http://www.csd.uoc.gr/~hannover/MMILab-Andre_files/HolzapfelIsmir09.pdf>.

[10] Y.F. Kutluğ: *Türk musikisinde makamlar*. YKY, İstanbul, 2000.

[11] N.O. Levendoğlu: "Türk Müziğinde Rast Makamının İstatistiksel Değerlendirmesi". *Selçuk Üniversitesi Fen Bilimleri Enstitüsü basılmamış yüksek lisans tezi*, Konya, 1996.

[12] G. Oransay: *"Das Tonsystem Der Türkei - Türkischen Kunstmusik", Musicforschung,* Germany, 1959.

[13] İ.H. Özkan: *Türk musikisi nazariyatı ve usulleri - Kudüm velveleleri*. Ötüken Neşriyat, İstanbul, 1998.

[14] K.L. Signell: *Makam – Modal practice in Turkish art music*. Da Capo Press, 1986.

[15] H.T. Sümbüllü, A. Albuz: "Türk Sanat Müziği dizilerinin, bilgisayar destekli makamsal analizi", *Uluslararası İnsan Bilimleri Dergisi,* cilt 8, sayı 1, 2011.

[16] A. Töre ve E. Karadeniz: *Türk musikisi nazariye ve esasları*, T. İş Bankası yayınları, İstanbul, 1965.

[17] Y. Tura: *Türk musikisinin meseleleri*. Pan Yayıncılık, İstanbul, 1988.

[18] E. Ünal, B. Bozkurt, M.K. Karaosmanoğlu, "N-gram based Statistical Makam Detection on Makam Music in Turkey using Symbolic Data", submitted to *ISMIR,* 2012.

[19] O. Yarman: "79-tone Tuning & Theory for Turkish Maqam Music", *PhD Thesis, Istanbul Technical University, Social Sciences Inst.,* İstanbul, 2007.

[20] R. W. Young: "Terminology for Logarithmic Frequency Units". *The Journal of the Acoustical Society of America* 11 (1), 1939.

[1] The Scientific and Technological Research Council of Turkey

[2] www.mus2.com.tr

POLYPHONIC MUSIC CLASSIFICATION ON SYMBOLIC DATA USING DISSIMILARITY FUNCTIONS

Yoko Anan, Kohei Hatano, Hideo Bannai, Masayuki Takeda
Department of Informatics, Kyushu University
{yoko.anan, hatano, bannai, takeda}@inf.kyushu-u.ac.jp

Ken Satoh
National Institute of Informatics
ksatoh@nii.ac.jp

ABSTRACT

This paper addresses the polyphonic music classification problem on symbolic data. A new method is proposed which converts music pieces into binary chroma vector sequences and then classifies them by applying the dissimilarity-based classification method TWIST proposed in our previous work. One advantage of using TWIST is that it works with *any* dissimilarity measure. Computational experiments show that the proposed method drastically outperforms SVM and k-NN, the state-of-the-art classification methods.

1. INTRODUCTION

Classification of music is one of the most fundamental problems in music information retrieval research and has been studied extensively (e.g., [4, 5, 7, 19, 20, 25, 28]). Music is usually polyphonic in the sense that more than one tone sounds simultaneously and thus a single time interval is made up of two or more simultaneous tones. Classifying polyphonic music pieces seems to be more difficult than classifying monophonic music pieces.

The difficulty in classifying polyphonic music stems from two issues. The first issue is in determining what kind of information needs to be extracted from polyphonic music. Many previous research (e.g., [13, 18]) reduce the classification problem of polyphonic music to that of monophonic music by converting the data in some way. For example, the so-called skyline method converts polyphonic music to monophonic music by choosing the highest pitches among multiple pitches. This approach is effective to an extent, but it does not fully exploit the information which can be obtained from multiple pitches.

The second issue is how to classify the preprocessed data. A major approach of machine learning techniques is to represent data as feature vectors and then applying learning algorithms. There are several known features such as performance worm [9], performance alphabet [26] and others [4, 19, 20, 28]. Then, it is non-trivial to construct effective features from data, since such construction requires much human resource such as experts' knowledge. If we employ kernel-based machine learning approaches such as SVM (e.g., [15–17, 21, 29]), we can avoid the problem of explicitly constructing features by using kernels since kernels implicitly define features. However, the kernel-based has a limitation that the kernel must be positive-semidefinite. Therefore many popular (dis)similarity measures such as the edit distance is not applicable since they are not proved to be positive-semidefinite.

In this paper, we propose a new method for classifying polyphonic music, which is a combination of polyphonic music preprocessing and classification techniques. For preprocessing data, our method employs chroma vector representation which is popular for audio data (e.g., [3]). Unlike previous approaches such as the skyline method, we preprocess the data so that information contained in the original data is kept as much as possible. An advantage of the chosen approach is that it captures concurrent behavior of pitches by encoding them into a new set of strings, and therefore, can extract more information from polyphonic music data than the monophonic music reduction approach.

For classification, we propose a multi-class version of our classification method named TWIST (Tug-of-War between Instances by Soft margin optimization Technique) proposed in [2]. TWIST is based on the theory of Wang et al. [31] for learning with (dis)similarity functions. The theory guarantees that under some mild assumptions, the final classifier constructed from dissimilarity functions is accurate enough for future data. Further, TWIST can use any dissimilarity function which might not be positive semi-definite.

By combining the two approaches described above, we significantly outperform the state-of-the-art methods such as k-nearest neighbor (k-NN) and SVM with string kernels for composer classification tasks of classical piano music and Japanese POP music given in MIDI format.

2. OUTLINE OF OUR METHOD

In this section, we explain the outline of our method. Our method consists of two parts: (i) First of all, we convert polyphonic music data into binary chroma sequences. In our experiments, the original polyphonic data is given as MIDI data. (ii) Next, given labeled binary chroma sequences and a dissimilarity function which measures the discrepancy between them, we use a multi-class version of TWIST to learn a classifier. The details are given in Section 3.

3. QUANTIFYING DISSIMILARITY BETWEEN MUSIC PIECES

The key to successfully using TWIST is in choosing how to quantify dissimilarity between music pieces. We consider dissimilarity measures d that are combinations of a preprocessing p of music pieces into particular sequential representation, and a string dissimilarity measure δ. That is, d is given by $d(x, y) = \delta(p(x), p(y))$, where x, y are music pieces.

One popular sequential representation of polyphonic music is the chroma vector sequence representation. A chroma vector is a twelve-element vector with each dimension representing the intensity in a very short time interval, associated with a particular semitone regardless of octave. Chroma vectors model important aspect of music audio and have been widely used in music retrieval [22,23], music classification [1, 10], and several other applications in music information processing [24].

On the other hand, many string (dis)similarity measures have been proposed, such as the edit distance [30], the longest common subsequence (LCS) length [14], the normalized compression distance (NCD) [6], which are used in many applications such as automatic spelling correction, information retrieval, gene information analysis and so on. String kernels such as the n-gram kernel [16], the mismatch kernel [15], the subsequence kernel [21] are also string similarity measures.

Consider applying such a string similarity measure to music pieces that are given in the form of sequences of binary chroma vectors (or pitch sets). A naive approach would be to use the so-called *skyline method*, where the highest pitch is chosen among multiple pitches in each time interval. It is essentially a reduction to monophonic music processing.

Another approach is a direct computation of dissimilarity regarding the binary chroma vectors as just symbols. There are $2^{12} = 4096$ symbols. For the edit distance, we have to define the weights associated with edit operations, namely, the weight $w(a, \varepsilon)$ of deleting a and the weight $w(\varepsilon, a)$ of inserting a for any symbol a, and the weight $w(a, b)$ of replacing a with b for any distinct symbol pair (a, b). The simplest way is to use the unit weight function such that $w(a, \varepsilon) = w(\varepsilon, a) = 1$ for any symbol a and $w(a, b) = 1$ for any distinct symbol pair (a, b) [1]. Another possible way would be to set $w(a, b) = 1 - \theta(a, b)$ where $\theta(a, b)$ is the angle between the vectors a and b for any distinct symbol pair (a, b), and $w(a, \varepsilon) = w(\varepsilon, a) = 1$ for any symbol a. An alternative way is to quantify resemblance between chroma vectors based on musical knowledge. Harte et al. [12] proposed such a method: It converts 12-dim. binary chroma vectors into 6-dim. real-valued vectors, called the tonal centroid vectors (TC vectors in short). They claim in [12] that close harmonic relations such as fifths and thirds appear as small Euclidian distances. Thus the Euclidian distance of two TC vectors or the cosine of the angle between them could be a good dissimilarity (similarity) measure between original chroma vectors.

[1] The edit distance with this weight function is often called the Levenshtein distance.

Using TC vector conversion Ahonen et al. [1] took another approach. The component values of TC vectors are quantized to 0 or 1 to produce 6-dim. binary vectors, which we call the *binary TC vectors*. The resulting sequences are thus strings over an alphabet of size 64, and NCD with `bzip2/PPMZ` is used to quantify their dissimilarity.

In Section 5 we compare by computational experiments the performance of all combinations of sequential representation and (dis)similarity measure mentioned above.

4. CLASSIFICATION METHOD

In this section, we briefly sketch TWIST [2] which is designed for binary classification. Then we show how to extend TWIST for multi-class classification tasks.

4.1 binary classification

TWIST employs a dissimilarity-based learning framework of [31], which we call TW (Tug-of-War). We first explain the TW framework below.

Let X be the instance space. We call a pair (x, y) of instance $x \in \mathrm{X}$ and label $y \in \{-1, 1\}$ an *example*. Suppose that we are given p positive examples $(x_1^+, +1) \ldots, (x_p^+, +1)$ and n negative examples $(x_1^-, -1), \ldots, (x_n^-, -1)$. We are also given a dissimilarity function $d(x, x')$ is a function from $\mathrm{X} \times \mathrm{X}$ to $\mathbb{R}^+$.

For each pair of positive instance x_i^+ and negative instance x_j^- ($i = 1, \ldots, p, j = 1, \ldots, n$), we define the base classifier $h_{i,j} : X \to \{-1, +1\}$ as follows:

$$h_{ij}(x) = \mathrm{sgn}(d(x_j^-, x) - d(x_i^+, x)),$$

where $\mathrm{sgn}(a) = 1$ if $a > 0$ and -1 otherwise. The base classifier h_{ij} classifies an instance x as positive if x is more dissimilar to the negative instance x_j^- than the positive instance x_i^+ (in other words, x is more similar to x_i^+ than x_j^-) and it classifies an instance x as negative, otherwise. The behavior of the base classifier seems like a tug-of-war, which is why we call the framework TW. In the TW framework, the final classifier is a weighted voting of base classifiers,

$$\mathrm{sgn}[\sum_{i=1}^{p} \sum_{j=1}^{n} w_{ij} h_{ij}(x)]$$

for some weights w_{ij}s. TW has a theoretical guarantee that, under some natural assumptions, there exist weights such that the associated final classifier is accurate enough for future instances. A heuristics is used to determine weights in the original paper [31].

Our previous work [2] uses a more robust method for finding weights than the above mentioned heuristics. We named the method of [2] TWIST, which is an abbreviation of "Tug-of-War of Instances by Soft margin optimization Technique". TWIST employs the 1-norm soft margin optimization to determine weights w_{ij}s. The 1-norm soft margin optimization is a standard formulation of classification problems in Machine Learning (see, e.g., [8, 32]), which is known to provide a robust classifier. In our case, the 1-norm soft margin optimization is formulated as follows:

$$\max_{\rho,b,\boldsymbol{w},\boldsymbol{\xi}^+,\boldsymbol{\xi}^-} \rho - \frac{1}{\nu}\sum_{k=1}^{p}\xi_k^+ - \frac{1}{\nu}\sum_{k=1}^{n}\xi_k^- \qquad (1)$$

sub.to

$$\sum_{i=1}^{p}\sum_{j=1}^{n} w_{ij}h_{ij}(x_k^+) + b) \geq \rho - \xi_k^+ (k = 1,\ldots,p),$$

$$-\sum_{i=1}^{p}\sum_{j=1}^{n} w_{ij}h_{ij}(x_k^-) + b) \geq \rho - \xi_k^- (k = 1,\ldots,n),$$

$$\boldsymbol{w} \geq \boldsymbol{0},\ \sum_{i=1}^{p}\sum_{j=1}^{n} w_{ij} = 1,\ \boldsymbol{\xi}^+,\ \boldsymbol{\xi}^- \geq \boldsymbol{0},$$

where each y_k is $+1$ or -1. An additional advantage of 1-norm soft margin optimization is that the resulting weights are likely to be sparse since we regularize 1-norm of the weights This property is also useful for feature selection tasks.

4.2 multi-class classification

We explain how to extend TWIST for multi-class classification. We employ the standard reduction method from multi-class to binary classification, one-versus-rest. The one-versus-rest method solves K-class classification by reducing it to K binary classification problems. For each class k ($1 \leq k \leq K$), the associated binary classification problem is constructed by assuming the label k is positive and other labels are negative. Then a learning algorithm is applied for each binary classification problem and it outputs the classifier $h_k : X \to \mathbb{R}$ for each class k. The final classifier of the one-versus-rest method is given as

$$\arg\max_{k=1,\ldots,K} h_k(x).$$

5. COMPUTATIONAL EXPERIMENT

We evaluated the performance of TWIST in composer classification of music pieces in comparison to those of the classification methods k-NN and SVM. A suitable dataset should contain enough number of music pieces for each composer, and the pieces for all composers have roughly the same conditions (the length, genre, instrument, etc. of the piece). Although various MIDI datasets are publicly available, datasets suitable for composer classification are rare. For example, the classical music dataset of the RWC Music Database [2] consists of 50 pieces written by 24 composers, only 2.08 pieces for each composer on the average.

The following datasets were available for our experiments:

Classical. The set of classical music MIDI files described in [27]. It consists of 5 sets of 25 pieces of keyboard music, written by Bach, Beethoven, Chopin, Mozart and Schumann, respectively.

JPOP. A set of Japanese POP (JPOP) music MIDI files for KARAOKE downloaded from a commercial site by YAMAHA. It consists of 5 sets of 25 pieces of JPOP, written by 5 composers (Tomoyasu Hotei, Tetsuya Komuro, Keisuke Kuwata, Takahiro Matsumoto, Kazumasa Oda).

From the MIDI files, we removed the MIDI events other than the NOTE ON/OFF events and quantized the NOTE ON/OFF times with unit time corresponding to the sixteenth note length. The tracks/channels of the MIDI files can then be viewed as sequences of sets of pitches that are "ON" in respective unit time intervals. Each MIDI file in **Classical** consists of two tracks, corresponding to the left and right hand parts. We extracted two pitch-set sequences and merged them into a single sequence. Each MIDI file in **JPOP** consists of a single track with several channels. We chose the channel 0 corresponding to main melody part and obtained a single pitch-set sequence from the channel. We then converted the obtained pitch-set sequences into (a) highest-pitch strings, (b) binary chroma vector sequences, and (c) binary TC vector sequences.

For quantifying dissimilarities between sequences, we adopted several (dis)similarity measures between strings. For TWIST, SVM and k-NN, we used the following two string kernels: n-gram kernel with parameters $n = 2, 5, 10$ and mismatch kernel with parameters $n = 5, 10$ and $m = 1, 2$, where m is the maximum number of errors allowed. For TWIST and k-NN, we also used the following (dis)similarity measures: edit distance, LCS, and NCD with compression programs `gzip` and `bzip2`.

For the edit distance between binary chroma vector sequences, we used the symbol-pair weight functions w of the three types: (i) the unit weight ($w(a, b) = 1$ if $a \neq b$ and $w(a, b) = 0$ if $a = b$); (ii) $w(a, b) = 1 - \cos\theta(a, b)$ for binary chroma vectors a, b; and (iii) $w(a, b) = 1 - \cos\theta(a', b')$ for TC vectors a', b' of binary chroma vectors a, b. For highest-pitch strings and binary TC vectors, we used only the unit weight. We used the cosine values for (ii) and (iii) in the case of LCS. We note that the compression programs `gzip` and `bzip2` used in NCD take data files of byte-sequences as input. We encoded the highest-pitch strings as one-byte-integer sequences, wrote them into data files and then applied the compressors to the files. For binary chroma vector sequences, we wrote them as data files of two-byte-integer sequences to be processed in a byte-wise manner by the compressors.

We evaluated the three classification methods by performing 5-fold cross validation. We used the values 0.05, 0.1, 0.2, 0.3, 0.4, 0.5 for the parameter ν of the 1-norm soft margin optimization formulation (1). For SVM, we used the ν-SVM implementation of LIBSVM (version 3.11) [11]. The values 0.05, 0.1, 0.2, 0.3, 0.4, 0.5 for the parameter ν were used. For k-NN, we used $k = 1, 3, 5$. Accuracies are obtained using the best value of ν for each method and each (dis) similarity measure.

[2] http://staff.aist.go.jp/m.goto/RWC-MDB/

Table 1. Comparison of classification accuracy for dataset **Classical** (in %).

(a) highest-pitch strings.

	edit distance	LCS	NCD		n-gram kernel			mismatch kernel			
	unit weight	unit weight	bzip2	gzip	$n = 2$	$n = 5$	$n = 10$	$n = 5$		$n = 10$	
								$m = 1$	$m = 2$	$m = 1$	$m = 2$
TWIST	70.40	78.40	80.80	**84.80**	76.00	77.60	72.80	72.80	81.60	79.20	75.20
1-NN	52.80	20.00	51.20	41.60	16.00	20.00	12.00	19.20	18.40	12.00	13.60
3-NN	60.00	18.40	62.40	48.00	17.60	41.60	44.80	28.00	19.20	46.40	39.20
5-NN	51.20	24.80	56.00	39.20	33.60	27.20	32.00	36.00	34.40	31.20	30.40
SVM	N/A	N/A	N/A	N/A	51.20	44.00	24.80	52.00	52.80	26.40	29.60

(b) binary chroma vector sequences.

	edit distance			LCS			NCD		n-gram kernel			mismatch kernel			
	weight			weight			bzip2	gzip	$n = 2$	$n = 5$	$n = 10$	$n = 5$		$n = 10$	
	unit	cosine	TC	unit	cosine	TC						$m = 1$	$m = 2$	$m = 1$	$m = 2$
TWIST	80.00	70.40	72.00	81.60	77.60	77.60	91.20	**92.00**	80.00	70.40	49.60	74.40	73.60	61.60	67.20
1-NN	46.40	50.40	44.80	29.60	30.40	27.20	63.20	53.60	20.00	22.40	24.00	20.80	21.60	18.40	17.60
3-NN	35.20	43.20	38.40	25.60	28.80	24.00	68.00	63.20	16.80	4.80	8.00	10.40	16.80	4.80	4.80
5-NN	25.60	40.00	37.60	23.20	23.20	21.60	65.60	56.00	25.60	1.60	9.60	3.20	11.20	4.80	4.00
SVM	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	57.60	40.00	24.80	44.80	51.20	24.00	23.20

(c) binary TC vector sequences.

	edit distance	LCS	NCD		n-gram kernel			mismatch kernel			
	unit weight	unit weight	bzip2	gzip	$n = 2$	$n = 5$	$n = 10$	$n = 5$		$n = 10$	
								$m = 1$	$m = 2$	$m = 1$	$m = 2$
TWIST	76.00	78.40	85.60	**88.80**	78.40	70.40	59.20	72.00	73.60	64.80	67.20
1-NN	42.40	28.00	56.00	59.20	20.00	20.00	17.60	19.20	19.20	16.00	17.60
3-NN	32.00	25.60	64.80	65.60	18.40	18.40	17.60	18.40	18.40	15.20	16.00
5-NN	28.00	25.60	57.60	52.00	22.40	1.60	6.40	3.20	12.00	1.60	0.80
SVM	N/A	N/A	N/A	N/A	58.40	46.40	25.60	52.00	54.40	25.60	29.60

The experimental results for **Classical** are summarized in Table 1. TWIST drastically outperforms the other classification methods in all combinations of sequential representation and (dis)similarity measure. TWIST shows the best accuracy when used with combination of the binary chroma vector sequence representation and NCD with `gzip`.

The experimental results for **JPOP** are summarized in Table 2. Again, TWIST defeats the other classification methods in most combinations of sequential representations and (dis)similarity measures. This time TWIST using NCD with `bzip2` shows good accuracies. In fact the best and the second best are achieved by NCD with `bzip2` combined with the highest-pitch strings and with the binary chroma vector sequence representations, respectively.

Now we discuss the effects of preprocessing for classification of **JPOP** and **Classical**. For **JPOP**, the classification accuracy with highest-pitch strings is better than that with chroma vector sequences. This might be due to the fact that **JPOP** is almost like monophonic music. More precisely, each music of **JPOP** is characterized with highest-pitch sequence mostly corresponding to the lead vocal line. On the other hand, each piano music of **Classical** is characterized with a succession of simultaneously sounding pitches. Therefore, chroma vector representation is more advantageous for classification of polyphonic music since the representation keeps more information in the original music.

6. CONCLUSION

In this paper we proposed a polyphonic music classification method as a combination of way of quantifying affinity between music pieces and the classification technique TWIST [2]. The method converts given music data into binary chroma vector sequences, and builds a classifier based on the similarity values between the sequences (or their converted sequences) using a string similarity measure. One advantage is that TWIST works with *any* similarity measure, not necessarily to be positive semidefinite. The results of computational experiments with classical music and Japanese POP music show that TWIST drastically outperforms the well-known classification methods k-NN and SVM with string kernels in all combinations of sequential representation and similarity measure.

Although the computational experiments were carried out on MIDI files, our classification method can, in theory, be applied to audio files, provided that an appropriate function that quantifies affinity between music audio data

Table 2. Comparison of classification accuracy for dataset **JPOP** (in %).

(a) highest-pitch strings.

	edit distance	LCS	NCD		n-gram kernel			mismatch kernel			
	unit weight	unit weight	`bzip2`	`gzip`	$n=2$	$n=5$	$n=10$	$n=5$		$n=10$	
								$m=1$	$m=2$	$m=1$	$m=2$
TWIST	78.40	80.80	**86.40**	73.60	38.40	31.20	39.20	28.80	48.80	26.40	28.00
1-NN	26.40	21.60	33.60	27.20	19.20	19.20	19.20	19.20	19.20	19.20	19.20
3-NN	37.60	45.60	58.40	44.00	58.40	58.40	20.00	20.00	58.40	58.40	20.00
5-NN	34.40	42.40	43.20	40.00	20.00	20.00	20.00	20.00	20.00	20.00	20.00
SVM	N/A	N/A	N/A	N/A	35.20	25.60	17.60	25.60	36.00	17.60	19.20

(b) binary chroma vector sequences.

	edit distance			LCS			NCD		n-gram kernel			mismatch kernel			
	weight			weight			`bzip2`	`gzip`	$n=2$	$n=5$	$n=10$	$n=5$		$n=10$	
	unit	cosine	TC	unit	cosine	TC						$m=1$	$m=2$	$m=1$	$m=2$
TWIST	80.00	83.20	83.20	76.80	81.60	81.60	**84.80**	79.20	60.80	48.80	28.80	44.00	50.40	36.00	34.40
1-NN	31.20	30.40	34.40	27.20	27.20	30.40	35.20	30.40	19.20	19.20	19.20	19.20	19.20	19.20	19.20
3-NN	44.80	44.80	49.60	53.60	53.60	55.20	51.20	44.80	57.60	56.80	58.40	58.40	49.60	57.60	58.40
5-NN	41.60	41.60	46.40	46.40	46.40	32.80	48.80	40.00	19.20	20.00	20.00	20.00	27.20	20.00	20.00
SVM	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	40.80	40.00	32.00	40.80	40.80	33.60	36.00

(c) binary TC vector sequences.

	edit distance	LCS	NCD		n-gram kernel			mismatch kernel			
	unit weight	unit weight	`bzip2`	`gzip`	$n=2$	$n=5$	$n=10$	$n=5$		$n=10$	
								$m=1$	$m=2$	$m=1$	$m=2$
TWIST	72.80	**83.20**	79.20	80.80	56.00	38.40	40.00	45.60	42.40	28.80	37.60
1-NN	28.80	28.00	31.20	30.40	19.20	19.20	19.20	19.20	19.20	19.20	19.20
3-NN	44.00	55.20	48.80	55.20	57.60	56.00	58.40	58.40	42.40	57.60	58.40
5-NN	44.00	48.00	45.60	40.00	20.00	21.60	20.00	20.00	27.20	20.00	20.00
SVM	N/A	N/A	N/A	N/A	38.40	39.20	32.00	41.60	43.20	33.60	36.00

is available. A future work is to develop such a function for music audio data.

7. ACKNOWLEDGMENTS

We thank anonymous referees for helpful comments and suggestions.

8. REFERENCES

[1] Teppo E. Ahonen, Kjell Lemström, and Simo Linkola. Compression-based similarity measures in symbolic, polyphonic music. In *Proceedings of the 12th International Symposium on Music Information Retrieval (ISMIR'11)*, pages 91–96, 2011.

[2] Yoko Anan, Kohei Hatano, Hideo Bannai, and Masayuki Takeda. Music genre classification using similarity functions. In *Proceedings of the 12th International Symposium on Music Information Retrieval (ISMIR'11)*, pages 693–698, 2011.

[3] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, 2005.

[4] James Bergstra, Norman Casagrande, Dumitru Erhan, Douglas Eck, and Balázs Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning*, 65:473–484, 2006.

[5] Rudi Cilibrasi, Paul Vitányi, and Ronald de Wolf. Algorithmic clustering of music based on string compression. *Computer Music Journal*, 28(4):49–67, 2004.

[6] Rudi Cilibrasi and Paul M. B. Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51:1523–1545, 2005.

[7] 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)*, pages 77–80, 2007.

[8] Ayhan Demiriz, Kristin P. Bennett, and John Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1-3):225–254, 2002.

[9] Simon Dixon, Werner Goebl, and Gerhard Widmer. The performance worm: Real time visualisation of expression based on langner's tempo-loudness animation. In *Proceedings of the International Computer Music Conference (ICMC'02)*, pages 361–364, 2002.

[10] Daniel P. W. Ellis. Classifying music audio with timbral and chroma features. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR '07)*, pages 339–340, 2007.

[11] Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin. Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research*, 6:1889–1918, 2005.

[12] Christopher Harte, Mark Sandler, and Martin Gasser. Detecting harmonic change in musical audio. In *Proceedings of 1st ACM Workshop on Audio and Music Computing Multimedia (AMCMM'06)*, pages 21–26, 2006.

[13] Ruben Hillewaere, Bernard Manderick, and Darrell Conklin. String quartet classification with monophonic models. In *Proceedings of the 11th International Society for Music Information Retrieval (ISMIR '10)*, pages 537–542, 2010.

[14] D.S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, 18(6):341–343, 1975.

[15] 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.

[16] Christina S. Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel: a string kernel for SVM protein classification. In *Proceedings of the Pacific Symposium on Biocomputing (PSB'02)*, pages 566–575, 2002.

[17] Christina S. Leslie and Rui Kang. Fast string kernels using inexact matching for protein sequences. *Journal of Machine Learning Research*, 5:1435–1455, 2004.

[18] Ming Li and Ronan Sleep. Melody classification using a similarity metric based on Kolmogorov complexity. In *Sound and Music Computing*, pages 126–129, 2004.

[19] 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.

[20] 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)*, pages 61–66, 2007.

[21] 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.

[22] Riccardo Miotto and Nicola Orio. A music identification system based on chroma indexing and statistical modeling. In *Proceedings of the 9th International Society for Music Information Retrieval (ISMIR '08)*, pages 301–306, 2008.

[23] 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 '05)*, pages 288–295, 2005.

[24] Laurent Oudre, Yves Grenier, and Cédric Févotte. Chord recognition by fitting rescaled chroma vectors to chord templates. *IEEE Transactions on Audio, Speech and Language Processing*, 19(7):2222 – 2233, 2011.

[25] Carlos Pérez-Sancho, David Rizo, and José Manuel Iñesta Quereda. Genre classification using chords and stochastic language models. *Connection Science*, 21:145–159, 2009.

[26] 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 (ECML'04)*, pages 384–395, 2004.

[27] Takuya Sawada and Ken Satoh. Composer classification based on patterns of short note sequences. In *Proceedings of the AAAI-2000 Workshop on AI and Music*, pages 24–27, 2000.

[28] 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)*, pages 293–302, 2001.

[29] S. V. N. Vishwanathan and Alexander J. Smola. Fast kernels for string and tree matching. In *Advances on Neural Information Processing Systems 15*, pages 569–576, 2002.

[30] Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.

[31] Liwei Wang, Masashi Sugiyama, Cheng Yang, Kohei Hatano, and Jufu Feng. Theory and algorithm for learning with dissimilarity functions. *Neural Computation*, 21(5):1459–1484, 2009.

[32] Manfred K. Warmuth, Karen Glocer, and Gunnar Rätsch. Boosting algorithms for maximizing the soft margin. In *Advances in Neural Information Processing Systems 20 (NIPS'08)*, pages 1585–1592, 2008.

SEMIOTIC STRUCTURE LABELING OF MUSIC PIECES : CONCEPTS, METHODS AND ANNOTATION CONVENTIONS

Frédéric BIMBOT
IRISA / METISS
CNRS - UMR 6074, France
frederic.bimbot@irisa.fr

Emmanuel DERUTY
INRIA / METISS
Rennes, France
emmanuel.deruty@gmail.com

Gabriel SARGENT
IRISA / METISS
Université Rennes 1, France
gabriel.sargent@irisa.fr

Emmanuel VINCENT
INRIA / METISS
Rennes, France
emmanuel.vincent@inria.fr

ABSTRACT

Music structure description, i.e. the task of representing the high-level organization of music pieces in a concise, generic and reproducible way, is currently a scientific challenge both algorithmically and conceptually. In this paper, we focus on *semiotic structure*, i.e. the description of similarities and internal relationships within a music piece, as a low-rate stream of arbitrary symbols from a limited alphabet and we address methodological questions related to annotation.

We formulate the labeling task as a blind demodulation problem, whose goal is to identify a minimal set of semiotic *codewords*, whose realizations within the music piece are subject to a number of *connotative variations* viewed as *modulations*. The determination of labels is achieved by combining morphological, paradigmatic and syntagmatic considerations relying respectively on (i) a morphological model of semiotic blocks in order to define their individual properties, (ii) the support of prototypical structural patterns to guide the comparison between blocks and (iii) a methodology for the determination of distinctive features across semiotic classes.

Specific notations are introduced to account for unresolvable semiotic ambiguities, which are occasional but must be considered as inherent to the music matter itself. A set of 500 music pieces labeled in accordance with the proposed concepts and annotation conventions is being released with this article.

1. INTRODUCTION

Music can be defined as "the art, process and result of deliberately arranging sound items with the purpose of reflecting and affecting senses, emotions and intellect" [1]. From a more operative viewpoint, music can be approached as a set of sounds organized by human composers for human listeners. From these definitions, the role of structure in the musical process appears as rather essential, as it is both a constituent and a support of the musical discourse.

In the domain of MIR, music structure is frequently considered as a central element to music description and modeling, but also as a scientific challenge, both algorithmically and conceptually [2]. This situation has triggered significant effort in the MIR community, towards the production of annotated resources [3][4] and the organization of evaluation campaigns [5].

At the scale of an entire piece, music structure is a concept which can be approached in several ways :

a. **The acoustic structure** which consists in describing the course and turns of active sources and/or timbral textures within the piece : singer(s), lead entries, instrumentation, etc…
b. **The functional structure** which is based on usual designations of the different parts in terms of their role in the music piece, for instance : intro – verse – chorus – bridge – etc… (cf. [6], for instance),
c. **The semiotic structure** which aims at representing, by a limited set of arbitrary symbols (called labels), the similarities (and interrelations) of structural segments within the piece.

These various views of music structure have influenced the design of methods and algorithms for the automatic analysis of audio data, for instance [7][8][9].

However, in spite of a need and an interest for methodological and operational concepts [10][11], there is no well-established principles for the structural annotation of music pieces, either in terms of problem statement, procedure, or annotation conventions, even in "simple" cases like pop songs.

In this context, some of our former work [12][13] has been focused on the definition of structural block boundaries. In this article, we address the labeling task, i.e. the determination of equivalence classes between structural segments so as to obtain a symbolic transcription of the piece's structure. Our methodology is primarily designed for audio data but can also be applied to written music.

By approaching a music piece as a "communication system", we formulate (section 2) the labeling task as the resolution of an ill-posed problem, for which the solution is seeked by assuming that recurring properties and systematic differences across structural blocks are more prone to be semiotically relevant than irregular and occasional variations.

Within this scope, semiotic analysis aims at ensuring a trade-off between :

- coverage : i.e. to encompass the largest possible number of musical properties in the semiotic description
- regularity : i.e. to obtain a transcription as regular as possible and relating to a simple prototypical pattern.
- accuracy : i.e. to account as faithfully as possible for the distinctive properties across semiotic elements.
- compactness : i.e. to limit the semiotic alphabet to a reasonable number (and distribution) of elements.

This trade-off is obtained by combining methodological criteria based on morphologic, syntagmatic and paradigmatic considerations (section 3).

In section 4, we introduce annotation conventions which cover the most typical situations but which also handle occasional semiotic ambiguities, i.e. segments undecidedly belonging to two classes at the same time.

We release, with this article, a set of approximately 500 music pieces, annotated with the proposed conventions, and accompanied with additional documentation.

2. CONCEPTUAL APPROACH

The semiotic annotation of a music piece consists in summarizing its high-level structure as a short sequence of arbitrary symbols drawn from a limited alphabet, for instance :

A B C D A B C D E C D C D D

In the scope of this work, we suppose that the elements thus indexed are structural segments (or *blocks*) of comparable size and at a typical timescale between 10 and 25 seconds.

2.1 A music piece viewed as a communication system

Assuming the existence of a semiotic description of music structure is intimately linked to the hypothesis of an underlying communication scheme which governs, at the structural scale, the global narrative organization of the piece.

It is rather commonplace to consider music in general as a means of communication, based on a set of rules and conventions (which clearly depend on the type of music under consideration). Here, we consider that *each music piece* can itself be viewed as the output of a particular communication system, with its own constituents : a *sender* (the composer), one (or several) *receiver(s)* (i.e. listeners), a transmission *channel* (which can take various forms), a message (the musical *narration*) and a *code*, namely the alphabet of *semiotic elements* (or *codewords*) with which is built and developed the narration.

From this viewpoint, the codewords are fundamentally piece-specific and they are discovered (and inferred) gradually by the listener while the piece unfolds, i.e. while the musical narration develops. Describing the semiotic structure of a music piece can thus be viewed as a deciphering task based on the observation of the output of a communication system (possibly with the help of more or less conscious knowledge of composition conventions).

2.2 Semiotic structure : an ill-posed inverse problem

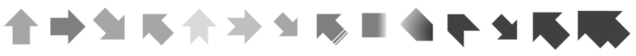
Semiotic structure description falls in the category of *ill-posed inverse problems*, and therefore cannot be solved unequivocally unless additional constraints are incorporated to condition the solution.

One option could be to try to formulate the conditioning constraints in terms of particular properties of the music substance, more or less specific to the genre of the piece. However, this approach would require sharp expertise, general concordance and a stable status of the compositional rules for all genres, which is difficult to imagine and which would certainly raise major problems of comparability and compatibility of the result across pieces (and annotators).

Our objective is to formulate the properties of semiotic elements (i.e. the conditioning constraints) as generically as possible. This is why, we propose that the structural description of the piece should be approached as some sort of *blind information demodulation problem*, i.e. the separation of a *carrier* (the sequence of codewords) and a *modulation* (the variations in the realization of the codewords), solely based on the *behavior and relationships* of musical properties, but not on their actual substance.

2.3 Inferring a semiotic code : an intuitive example

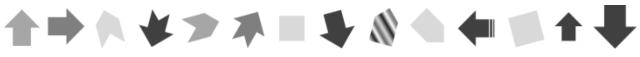
Let's consider the following sequence :



A careful study of this sequence reveals that :

- Almost all items are directional.
- Only 4 distinct orientations are observed (say N, E, SE and NW).
- Gray level varies but does not show any regular pattern, nor does the tail of the objects, nor their size.
- Orientation is driven by a strong syntax (for instance, SE is always followed by NW, NW never by E, etc…)
- The shape of item #9 is singular (could be interpreted as pointing SE *or* NW, though)

Let's now consider this second sequence :



Here, we observe that :

- Almost all items are (also) directional but
- Direction is taking all sort of random, unquantizable values
- Gray level takes only 4 different values (say 20, 40, 60 or 80 %)
- The tail of the objects (still) does not show any regular pattern, nor their size.
- Gray level is driven by a strong syntax (for instance 20 % is always followed by 80 %)
- The gray-level of item #9 is singular / undecided

The successive values or states taken by the various properties constitute *information layers* and, in both cases, a particular layer exhibits some systematic and organized behavior : "orientation" in the first sequence, "gray-level" in the second one. The knowledge of their behavior conveys, at a low explicative cost, significant information on the overall sequence. At the same time, the other properties appear mostly as creating different variants (or connotations) in the realization of these properties.

In both cases, the symbolic representation of the most organized information layer is :

A B C D A B C D E C D C D D

and in fact, considering any other property (or combination of properties) would create rather uninformative, much less regular or almost trivial descriptions, such as **ABCDE…KLMN, AABCBCDAEBADAA**, or **AAA…ABAAAAA**.

2.4 The carrier-modulation model

In the proposed approach, we assume that the sequence of structural elements can be decomposed into :

- **A carrier component** which is built on information layers whose behavior, is periodic, cyclic, regular, recurrent, repeated, correlated, quantized, organized…
- **A modulation component** which corresponds to information layers which appear as aperiodic, acyclic, irregular, occasional, erratic, sporadic, uncorrelated, isolated, continuous-valued, scattered, etc…

The sequence of semiotic labels describes the succession of *property values* taken by the carrier component. The modulation component represents circumstantial or incidental *variations* of the semiotic elements

2.5 Application to music

At the level of a short musical passage, successions of notes belonging to a given musical scale form an acoustic melody whose properties (amplitude, duration, attack, …) are modulated over time to convey expressivity.

Similarly, at the level of the whole piece, the succession of structural blocks forms some sort of *semiotic tune*, build on the "scale" of semiotic elements and whose modulation constitute *connotative variants* across different stages of the musical narration. However, as opposed to conventional music units, those "macro-notes" are piece-dependent and they are primarily inferred by detecting and comparing structural elements over the whole piece.

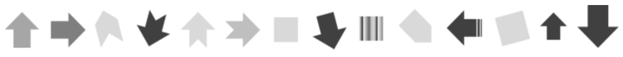
At this point, it is very important to note that the particular status of a property as being either semiotic (carrier) or connotative (modulation) cannot be *a priori* decided on the single basis of its nature.

Indeed, in a number of pieces, structural blocks are built on a few distinct harmonic progressions which recur throughout the piece together with a strong variability of the melodic line, whereas other pieces may be built on a unique harmonic cycle from the beginning to the end, the melodic line being the only distinctive feature between blocks. Some techno or electro pieces rather use the timbre or the texture to create structural patterns over a constant melodic-harmonic loop. In percussive pieces, the structural organization usually stems from rhythmic properties, etc…

Therefore, the analysis of semiotic structure primarily requires to identify, for each music piece, which are the *musical information layers* (melody, harmony, rhythm, etc…) taking part to the semiotic component.

2.6 Semiotic features

Let us now consider a third toy example, which we suppose to be an other realization of **ABCDABCDECDCDD** :



Here we have a slightly more complex situation than in section 2.3, in the sense that the structure of the sequence is based on a *switch* of the semiotic property on which the carrier component is built, as summarized in the table below :

Symbol	Orientation	Gray-level	Tail-shape	Size
A	**North**	*any*	*any*	*any*
B	**East**	*any*	*any*	*any*
C	*any*	**20 %**	*any*	*any*
D	*any*	**80 %**	*any*	*any*
E	**Indistinct**	**Multiple**	*any*	*any*

Such a situation is very common in music : for instance in a song, verses 1 and 2 may consist of two distinct melodies based on a same harmonic progression while choruses 1 and 2 may be based on two distinct chord loops (while the melodic line would be identical or almost). Indeed, in the case of music, semiotic structure relies on musical properties whose nature varies *across* pieces but which may also vary *within* a given piece.

Semiotic annotation therefore requires the determination of what we call *semiotic features*, i.e. not only the semiotic properties but also the particular values taken by these properties to form the carrier component describing the sequence of structural blocks. In the case above, the semiotic properties are *orientation* and *gray-level*, while the semiotic features are *North orientation*, *East orientation*, *20 % gray-level* and *80 % gray-level* (the other values of orientation and gray-level being irrelevant to the carrier component).

It is also worth noting that, in this example, the syntagmatic organization of the sequence plays an essential role in guiding the determination of the relevant semiotic features. Indeed, East is always followed by 20 %, itself always followed by 80 %, and more globally, the pattern N-E-20-80 is observed twice.

2.7 The three ingredients of semiotic structure

To sum up the underlying process at work behind semiotic analysis, we can distinguish 3 levels of reasoning which jointly participate to the determination of the semiotic description :

- Morphological analysis : *intrinsic features* of the structural elements composing the sequence (i.e, the properties and the values of these properties).
- Syntagmatic analysis : *local relations* that elements exhibit *with their neighbors* within the piece
- Paradigmatic analysis : *similarities* and *differences* which they exhibit *with other elements.*

The following sections investigate in details these three facets of semiotic structure analysis and how they interact with one another.

3. METHODOLOGICAL AXES

3.1 Morphological analysis

The morphological analysis of structural blocks is based on the *System & Contrast* (S&C) model [14].

Under this approach, each structural block is assumed to be built around 4 *morphological elements* (of typically 2 bars each) forming a *square carrier system.* These elements relate through a (usually 2x2) matrix of simple relationships. Structural blocks can be more complex, but they usually can be reduced to a square stem.

In general, on some musical information layers, the 4th element departs from the logical sequence formed by the first three (thus creating some sort of punctuation).

The morphology of a square S&C can be written as :

$$a \;\; f(a) \;\; g(a) \;\; \gamma(g(f(a)))$$

where a is the "seed" morphological element, f and g are the internal relations between the elements forming the carrier system and γ a *contrast* function that represents the (relative) disparity of the 4th element. S&Cs exist on several musical layers in different forms and at different timescales simultaneously. Their synchronization contributes to the musical consistency of the segment and to its autonomy [12]. Identifying S&Cs is thus very useful to locate, at the chosen timescale, the boundaries of the structural blocks.

A S&C can be summarized as a quadruplet : a, f, g, γ, which can be viewed as the "genetic code" of the structural block. Moreover, in many situations, either f or g (or both) are "identity" (*id*), resulting in well-identifiable morphological patterns such as *aaaa, abab, aabb* (for $\gamma = id$) or *aaab, abac, aabc* (for $\gamma \neq id$). These patterns can straightforwardly be extended to *"close-to-id"* or *"begins-like"* functions : *aaa'b, aba'c, aa'bc, ...*

The morphology of structural blocks can therefore be primarily characterized by the various systems followed by its (say *p*) active musical layers, i.e. as a multi-dimensional quadruplet $\{(a, f, g, \gamma)_i\}_{1 \leq i \leq p}$. In many cases, this quadruplet can be represented more simply as the morphological pattern governing each layer : for instance, $a_1 b_1 a_1 b_1$ for the melodic line, $a_2 b_2 a_2' c_2$ for the harmony, $a_3 a_3 a_3 b_3$ for the drum loop, etc…

3.2 Syntagmatic analysis

Moving back now to the timescale of the entire piece, we discuss how the position and context of structural blocks within the piece can be taken into account in order to guide semiotic labeling.

Indeed two structural blocks will be considered to be *a priori* more likely to belong to the same equivalence class if they appear in similar contexts in the piece, i.e. if they are located beside similar left and/or right segments within the piece. For instance, in the sequence **ABxDABy-DECDCDD**, **x** and **y** are more likely to belong to the same semiotic class than in **ABxDyBCDECDCDD.** This criterion partly relates to commutability, often used in semiotic analysis.

A second syntagmatic factor to take into account is that differences between two blocks should not be appreciated in the same way if the two blocks are immediately next to each other or if they appear at some distance in the piece : a slight difference observed between two successive similar blocks may be distinctive (especially if this opposition is recurrent in the piece) whereas a stronger difference at a long distance may just be a connotative variation, especially if the two blocks occur in similar contexts.

The guidance of a *prototypical structural pattern* (see Table I) is also an essential element of syntagmatic analysis, for weighing similarities and differences between and across blocks and interpreting them with respect to the global organization of the piece.

However, while the semiotic structure of music tends to be based on recurrent patterns, the actual *realization* of a structural pattern in a music piece generally shows irregularities (which are bound to increase when getting towards the end of the piece). For instance the structural description **ABCDABCDECDCDD** can be viewed as a realization of 4 cycles of an **(ABCD)** structural pattern with growing irregularities towards the end of the piece.

In practice, structural patterns can prove to be very efficient to guide the annotation for some musical genres, but they can also turn out to be totally useless for others.

3.3 Paradigmatic analysis

The goal of paradigmatic analysis is to determine the set of semiotic features within the population of structural blocks, i.e. what are the semiotic properties (and the values taken by these properties) which characterize the equivalence classes (cf. subsection 2.6).

A key concept of this process is that, rather than comparing the surface properties of the structural segments, the semiotic comparison of blocks is based primarily on the comparison of their *carrier system*, i.e. the triplet a, f, g (as defined in subsection 3.1) resulting from their morphological analysis. Note that the contrast function γ is treated separately, as a special form of modulation (see 4.1).

For a given music information layer, the carrier systems of two structural blocks **x** and **y** are considered as homologous, if there exist a *property* of that layer for which the triplets (a_x, f_x, g_x) and (a_y, f_y, g_y) are similar. For instance, if the musical layer is the melody, the property can be the melody itself, the support notes of the melody, the shape of the melodic line, etc…

This comparison is carried out for all musical layers which show some morphological organization and the subset of common properties that emerges from the comparison provides the characteristics of a potential class encompassing **x** and **y**. In particular, if the similarity of the systems holds for *all* music information layers active in **x** and **y**, it is considered that these segments should be grouped into a single semiotic class.

Semiotic features can thus be hypothesized as conjunctions of properties (*together with* their particular values) occurring in similar S&Cs and these features can be ordered (at least partly) according to their coverage of the various musical information layers.

A global solution is then searched (empirically) as a partition of the population of structural segments grouping those with *equivalent* carrier systems *in the subspace of* *their semiotic features*. In case of several possible solutions, the one yielding the most regular sequence of labels is chosen.

Finally, the set of *distinctive features* is established as the minimal subset of properties (and their particular values) which is necessary and sufficient to distinguish each semiotic element from all the others.

Of course, the trade-off between accuracy of the description and compactness of the semiotic set is an essential stopping condition. It is conjectured that a "good" *a posteriori* distribution of labels should follow some sort of Zipf law, or at least should not depart too much from it.

4. ANNOTATION CONVENTIONS

4.1 Primary notation of semiotic labels

Quite naturally, two blocks with non-equivalent carrier systems are denoted by 2 distinct alphabetic capital letters : **A** vs **B**.

When two blocks show equivalent carrier systems but different contrasts (γ), this difference is noted as a subscript : $\mathbf{A_1}$ vs $\mathbf{A_2}$. Blocks showing no (or extremely weak) contrasts are denoted $\mathbf{A_0}$ and conversely, blocks showing exceptional contrasts are denoted $\mathbf{A_*}$ (they usually tend to occur at the end of the piece).

When two blocks have equivalent carrier systems, but they significantly differ in their (surface) realizations (connotative variants), they are denoted with distinct superscripts, for instance : $\mathbf{A'}$ vs $\mathbf{A''}$. Optionally, the superscript may be chosen specifically to indicate the nature of the variant. For instance, the notations $\mathbf{A^+}$ and $\mathbf{A^-}$ are used to indicate more or less rich occurrences, and $\mathbf{A^\sim}$ can be used to denote exceptional variants of **A**.

When a property evolves gradually within a block, this is denoted as a fade-in or a fade-out. This is denoted as **<A, A>** and it may apply to surface properties such as the intensity, the instrumentation support, or to strengthening or vanishing properties of the carrier system.

If a block is realized only in a half-form, specific notations are used : **A/2** for a half-size block, **|A** and **A|** for left (resp. right) truncated half block, more general cases of incomplete blocks being denoted as **…A** or **A…**.

Table II summarizes these annotation conventions.

4.2 Composite labels for handling ambiguities

Inevitably, semiotic labeling leads to some situations which exhibit ambiguities, resulting from the *combination* or the *mutation* of semiotic items to create new hybrid ones, namely some sort of chord, at the level of the semiotic structure. This is indeed a natural process in music at many time-scales

Therefore, a set of additional notations were designed to render these ambiguities through *composite labels* (whose configurations are schematized on figure 1).

AB (*vertical hybrid*) : block showing undecided prevalence of properties of **A** and **B**, for instance, superposition.

A&B (*intrication*) : block showing intertwined portions of **A** and **B**. The size of block **A&B** is the total of that of **A** and **B**.

B|A (*horizontal hybrid*) : a specific system (**B**) is present in the first half of the block but the second half recalls the system of **A** (sometimes with a different contrast).

B<A (*kinship*) : block **B** is acceptable as autonomous at the current timescale but it exhibits strong cohesion with the previous block **A** via some common property or super-system. The sequence **[A][B<A]** could be described as a single morphological system at the immediately upper timescale.

B/A (*mutation*) : morphological system partly similar to **A** (*rooted* in **A**) but with a subset of properties whose system strongly departs from that of the other elements of class **A**. This is typically the case in some types of solos, where the harmony is common to some former block but the melody of the lead becomes freestyle. This situation also includes cases when the subset of properties is simply void, for example a passage (in particular, an intro) where the instrumental background is played alone, without any main lead.

A?B (*indetermination*) : ambiguous segment which cannot be annotated unequivocally. A typical situation like this is $\mathbf{B?A^\sim}$, when it is impossible for the annotator to conclude whether the segment is a specific semiotic element **B** or a very particular connotative variant of item **A**.

We also introduce notations for short segments that occasionally intervene in between regular ones :

A_B (*overlap*) : i.e. tiling segment corresponding to a partial superposition between **A** and **B**.

_AB_ (*connexion*) : *short segment* located between **A** and **B** which cannot decidedly be related primarily to **A** or **B**.

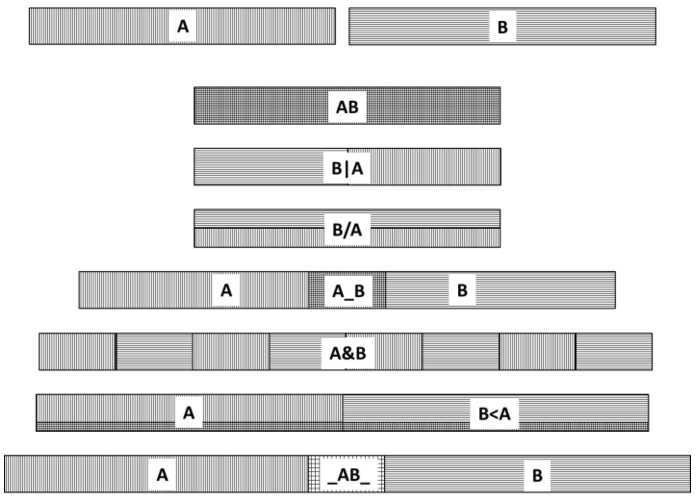


Figure 1 : composite labels (schematic configurations)

4.3 Proto-functional symbols

Even though semiotic structure description is distinct from functional structure description, we consider that it can be informative to choose the semiotic labels in such a way that they somehow reflect the *proto-functional* status of the block within the piece. We therefore propose to use, *as much as possible*, the alphabetic letters with the correspondence given in Table III.

4.4 Transcription example

Semiotic symbols can be put into brackets to facilitate visual parsing, especially when they are composite. Due to a lack of space, we leave it to the reader to "decode" the semiotic transcription represented below :

[I/A] [A_1] [A_2] [B] [C] [J/2] [A'_1] [|A'_2] [B'] [C] [X/C] [Y/2] [C_*]

4.5 Concluding remark

We want to underline that a primary objective of these notations is to provide a consistent *communication language* for describing the most obvious aspects of the semiotic structure of music, while also being able to reflect some of its subtle ambiguities.

5. ANNOTATION EFFORT AND FUTURE WORK

The set of approximately 500 music pieces, for which annotations in terms of structural boundaries have been released in 2011 [13], has been updated and complemented with semiotic labels obtained with the present methodology and is accessible at [15]. These annotations have been produced manually. They come with additional documentation and with the analysis of some difficulties met during the annotation process.

Future work will be turned towards the formalization of the concepts and methodology presented in this article in terms of information theory criteria, and their investigation for the design of models and algorithm for the automatic inference of music structure.

REFERENCES

[1] Mixture of several definitions from various sources.

[2] J. Paulus, M. Müller, A. Klapuri, Audio-based music structure analysis, Proc. ISMIR 2010.

[3] SALAMI Project : http://salami.music.mcgill.ca

[4] QUAERO Project : http://www.quaero.org

[5] MIREX 2011 : http://www.music-ir.org/mirex/2011

[6] Ten Minute Master n°18 : Song Structure - Music Tech Magazine, October 2003, pp. 62-63.

[7] J. Foote. Automatic audio segmentation using a measure of audio novelty. IEEE ICME, pp. 452–455, Aug. 2000.

[8] J. Paulus. Signal Processing Methods for Drum Transcription and Music Structure Analysis. *PhD Thesis*, 2009.

[9] M. Müller and F. Kurth. Towards structural analysis of audio recordings in the presence of musical variations. *EURASIP Journal on Advances in Signal Processing*, 2007.

[10] G. Peeters, E. Deruty : Is Music Structure Annotation Multi Dimensional ? LSAS, Graz (Austria) 2009.

[11] J.B.L. Smith, J.A. Burgoyne, I. Fujinaga, D. De Roure, J.S. Downie : Design and Creation of a Large-Scale Database of Structural Annotations. Proc. ISMIR 2011.

[12] F. Bimbot, O. Le Blouch, G. Sargent, E. Vincent : Decomposition into Autonomous and Comparable Blocks : A Structural Description of Music Pieces, Proc. ISMIR 2010.

[13] F. Bimbot, E. Deruty, G. Sargent, E. Vincent : Methodology and Resources for The Structural Segmentation of Music Pieces into Autonomous and Comparable Blocks, Proc. ISMIR 2011.

[14] F. Bimbot, E. Deruty, G. Sargent, E. Vincent : Complementary report to the article "Semiotic Structure Labeling of Music Pieces : Concepts, Methods and Annotations Conventions". IRISA Internal Report n° 1996, June 2012.

[15] http://musicdata.gforge.inria.fr

Acknowledgements

Part of this work was achieved in the context of the QUAERO project, funded by OSEO.

Prototype	Illustration	Codification
Trivial	AAAAAAAA…	(A)
Binary	ABABABABAB…	(AB)
Ternary	ABCABCABC…	(ABC)
Quaternary	ABCDABCDABCD…	(ABCD)
Alternate	AABCCDAABCCD…	(2A,B,2C,D)
Cyclic	ABBCDDDABBCDDD…	(A,2B,C,3D)
Acyclic	ABBCDDDEEF…	A,2B,C,3D,2E,…
Ergodic	ABCDBADAAACBCC…	{ABCD}

Table I : Most common prototypical structural patterns

	Regular	Specific
Semiotic variants	A_1, A_2, A_3, …, A_i, A_j	A_0, A_*
Connotative variants	A', A'', …, $A^{(i)}$, $A^{(j)}$	A^+, A^-, $A^\sim$

Fade-in / out	Non-square	Incomplete
<A A>	A/2 or (1/2)A (3/4)A, (5/4)A	|A, A| …A, A…

Table II : Main set of semiotic labels

	Intro	Pre-central	Central	Post-central	Relay	Other (recurrent)	Other (sporadic)	Outro
Primary set	I, J	A,B	C,D	E,F	J,K	M,N	X,Y,Z	K, L
Secondary set	G,H	P,Q	R,S	T,U	G,H	V,W		G,H

Table III : Proto-functional semiotic symbols

LARGE-SCALE COVER SONG RECOGNITION USING THE 2D FOURIER TRANSFORM MAGNITUDE

Thierry Bertin-Mahieux
Columbia University
LabROSA, EE Dept.
tb2332@columbia.edu

Daniel P.W. Ellis
Columbia University
LabROSA, EE Dept.
dpwe@ee.columbia.edu

ABSTRACT

Large-scale cover song recognition involves calculating item-to-item similarities that can accommodate differences in timing and tempo, rendering simple Euclidean measures unsuitable. Expensive solutions such as dynamic time warping do not scale to million of instances, making them inappropriate for commercial-scale applications. In this work, we transform a beat-synchronous chroma matrix with a 2D Fourier transform and show that the resulting representation has properties that fit the cover song recognition task. We can also apply PCA to efficiently scale comparisons. We report the best results to date on the largest available dataset of around 18,000 cover songs amid one million tracks, giving a mean average precision of 3.0%.

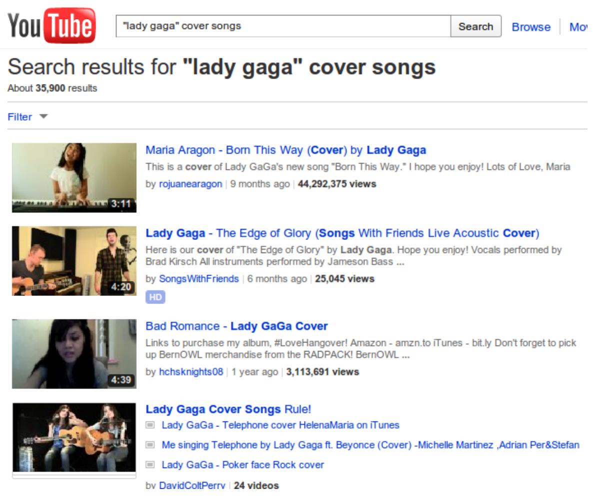
1. INTRODUCTION

Music videos are abundant on the web, and tracing the dissemination of a given work is a challenging task. Audio fingerprinting [26] can be used to find an exact copy of a given music track, but the same technology will not work for finding novel versions of the original work, i.e. "cover songs". Since cover songs are typically recorded by musicians other than those who originally commercialized the track, one motivation for identifying such "covers" is to ensure the correct handling of songwriting royalties. For instance, on YouTube, the copyright holder of the musical work can have the covers of her work removed, or she can receive part of the advertising revenue from the video [1]. Figure 1 shows how easy it is to find thousands of such covers online from the metadata alone, but many more are not identified as covers. Another reason to study cover song recognition is that finding and understanding transformations of a musical piece that retain its essential identity can help us to develop intelligent audio algorithms that recognize common patterns among musical excerpts.

Until recently, cover recognition was studied on a small scale (a few hundred tracks) due in part to the scarcity of generally-available databases. Most current algorithms are based on comparisons between chroma patterns, or a related feature, using some form of dynamic time-warping (DTW) [23]. Chromas are derived from the spectrogram and provide a coarse approximation of the musical score (see Section 2), which make them suitable for our task. Recently, the release of the Million Song Dataset [1] (MSD), which contains metadata and audio features (including chromas) for one million songs, has spurred the investigation of large-scale music information retrieval techniques. Linked to this dataset, the SecondHandSongs dataset (SHS) identifies approximately eighteen thousand cover songs. The task of finding these cover songs within one million tracks makes it much closer to a commercial-scale application.

Very large datasets constrain the amount of computation that can be devoted to individual comparisons, making DTW an increasingly infeasible choice. To work with millions of examples, we would ideally reduce each comparison to a simple operation in a low-dimensional space. Such a space may be defined via hash codes extracted from the signal in the spirit of fingerprinting algorithms [2]. Hash codes can be efficiently indexed, and finding a song that contains particular hash codes is extremely fast. Another option is to project the entire song into a small fixed dimension space in which nearest neighbors are our candidate covers. Nearest neighbor methods are easy to parallelize

[1] `http://www.youtube.com/t/faq`


"lady gaga" cover songs | Search | Browse | Mo

Search results for **"lady gaga" cover songs**
About **35,900** results

Filter

Maria Aragon - Born This Way (**Cover**) by **Lady Gaga**
This is a **cover** of Lady GaGa's new song "Born This Way." I hope you enjoy! Lots of Love, Maria
by rojuanearagon | 9 months ago | **44,292,375 views**

Lady Gaga - The Edge of Glory (**Songs** With Friends Live Acoustic **Cover**)
Here is our **cover** of "The Edge of Glory" by **Lady Gaga**. Hope you enjoy! Vocals performed by Brad Kirsch All instruments performed by Jameson Bass ...
by SongsWithFriends | 6 months ago | **25,045 views**

Bad Romance - **Lady GaGa Cover**
Links to purchase my album, #LoveHangover! Amazon - amzn.to iTunes - bit.ly Don't forget to pick up BernOWL merchandise from the RADPACK! BernOWL ...
by hchsknights08 | 1 year ago | **3,113,691 views**

Lady Gaga Cover Songs Rule!
Lady GaGa - Telephone cover HelenaMaria on iTunes
Me singing Telephone by Lady Gaga ft. Beyonce (Cover) -Michelle Martinez ,Adrian Per&Stefan
Lady GaGa - Poker face Rock cover
by DavidColtPerry | **24 videos**

Figure 1. Search result for self-identified cover songs of Lady Gaga on YouTube on November 22nd, 2011. This simple query produces about 35, 900 results.

and scale, and working with a fixed-dimensional representation (instead of a variable-length audio signal) is a great convenience.

Disappointed by the results presented in [2], we focus on the second option. Beat-synchronous chroma representations form a relatively compact description that retains information relevant to covers, and may be cropped to a constant size. Unfortunately, direct comparison of chroma patterns using common metrics is poorly behaved [3]. Summarizing a song by extracting a few chroma patches and comparing them with Euclidean distance gives unusable results. In order to obtain an efficient nearest-neighbor algorithm, we need a representation for the chroma patches with the following properties:

- representation vectors can be compared using a simple metric, e.g. Euclidean distance;
- representation vectors are compact, i.e. low-dimensional;
- representation must be robust to semitone rotations (musical transpositions);
- representation should be robust to different pattern offsets (time alignments).

The last condition would allow us to match patches without having to identify a universal time alignment which is very difficult in a large set. Our candidate representation is the two-dimensional Fourier transform magnitude (2DFTM), a simplified version of the representation used in [18] and discussed in Subsection 3.2. The Fourier transform separates patterns into different levels of detail, which is useful for compacting energy (as in image compression schemes such as JPEG) and for matching in Euclidean space. Discarding the phase component provides invariance both to transposition (rotation) in the pitch axes and skew (misalignment) on the beat (time) axis. Thus, taking the 2DFTM of a chroma patch, we obtain a transformation of chroma features that makes Euclidean distance quite useful, even after dimensionality reduction through PCA. Our method encodes each track as a 50-dimensional vector and provides a large improvement over the hash code-based method [2]. On the SHS, this approach gives the best result reported so far.

The rest of this work is organized as follows: In Section 2, we present the chroma feature and its variants, its metric issues, and the task of cover song recognition. In Section 3, we describe how we transform the chroma matrices and look at the resulting invariance properties. Section 4 details our experiments on large-scale cover song recognition, and we conclude in Section 5.

2. PREVIOUS WORK

2.1 Chroma feature and distance

Chroma features were introduced as pitch-class profiles (PCP) [9]. Many variants have been derived, including HPCP [10] and CENS [21]; an overview can be found in [15]. There is even evidence that chromas can be learned from a simple similarity task [12].

Unfortunately, chroma matrices (or chroma patches, our term for chroma matrices with a fixed number of time samples) are high-dimensional features that are difficult to compare with usual metrics [3]. Previous work has experimented with Euclidean distance [3, 23], cosine distance [23], Kullback-Leibler divergence [3,22] and Itakura-Saito divergence [22]. None of the results were fully satisfying for the task of cover song recognition.

2.2 Cover song recognition

Cover song recognition has been widely studied in recent years, including a specific task within MIREX since 2007 [6]. An early system is Ellis and Poliner [8] and a good overview is in Serrà's thesis [23]. A significant amount of work has been done with classical music [16, 19–21] but popular music can present a richer range of variation in style and instrumentation.

Most cover song works were tested on a few hundred or thousand songs, a size not comparable to commercial collections (Spotify [2] claims more than 15M tracks). However, some of the work was made to scale and could be extended to larger sets. Kurth and Müller [17] use a codebook of CENS features to encode songs, thus creating an index that can be searched efficiently. Casey and Slaney [5] use locally-sensitive hashing (LSH) to quickly compare chroma patches, or *shingles*. Yu et al. [27] also use LSH to compare different statistics about a song. Kim and Narayanan [16] look at chroma changes over time and suggest using these changes as hash codes. Finally, our previous method [2] uses hash codes inspired by the fingerprint system of [26], i.e., identifying peaks in the chromagram and encode their relative positions. This was the first result reported on the SHS.

The idea of using the magnitude of the two-dimensional Fourier transform has been explored in [14, 18]. As with the methods above, these were tested on a few hundreds or thousands examples. The differences with the method used in this work are highlighted in Subsection 3.4

3. CHROMA FEATURE AND 2DFTM

Our feature representation is the magnitude of the two-dimensional Fourier transform of beat-aligned chroma patches. Below, we explain how they are computed and discuss their properties.

3.1 Chroma

We use the chroma features computed by the Echo Nest API [7] as described in [13]. They are available as part of the Million Song Dataset [1] and were used in [2].

A chromagram is similar in spirit to a constant-Q spectrogram except that pitch content is folded into a single octave of 12 discrete bins, each corresponding to a particular semitone (e.g., one key of the octave on a piano). For each song in the MSD, the Echo Nest analyzer gives a chroma vector (length 12) for every music event (called

[2] `http://www.spotify.com`

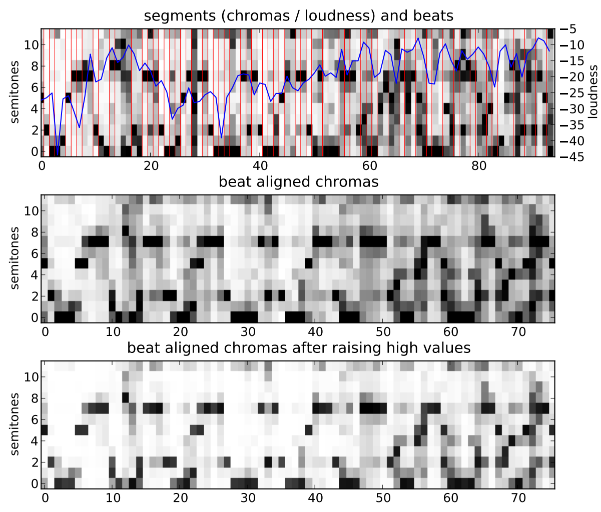


Figure 2. Beat-aligned chroma features.

"segment"), and a segmentation of the song into beats. Beats may span or subdivide segments. Averaging the per-segment chroma over beat times results in a beat-synchronous chroma feature representation similar to that used in [8]. Echo Nest chroma vectors are normalized to have the largest value in each column equal to 1.

Empirically, we found improved results from raising the highest values relative to the lowest ones by a power-law expansion. We believe this accentuates the main patterns in the signal. Figure 2 illustrates the stages in converting segment-aligned chroma features and their loudness to create beat-aligned chroma features for our task.

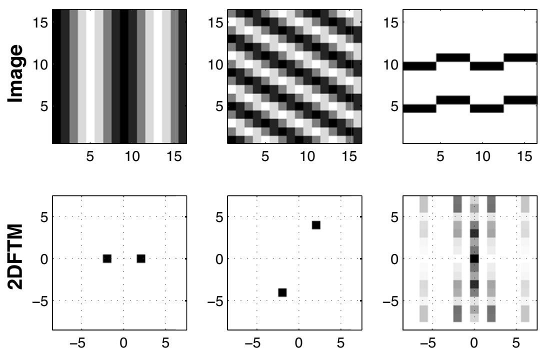


Figure 3. Examples of images and the corresponding 2DFTMs. FT images have the low-frequency values in the center.

3.2 Magnitude of the 2D Fourier Transform

Taking the two-dimensional Fourier transform is a common task in digital image processing where it is useful for denoising and compression, among other things [11]. As illustrated in the examples of Figure 3, a single point in the 2D Fourier transform corresponds to a sinusoidal grid of a particular period and orientation in the image (transform) domain; more complex images are built up out of multiple sinusoid grids.

We skip the definition of the 2D Fourier transform and its magnitude since it is widely known. Note that for visualization purposes, the bins are shifted so that the center of the axis system is in the middle of the image [3] . In that representation, the transformed image is symmetrical about the origin. Figure 4 gives an example of the transform applied to the chroma patch from Figure 2.

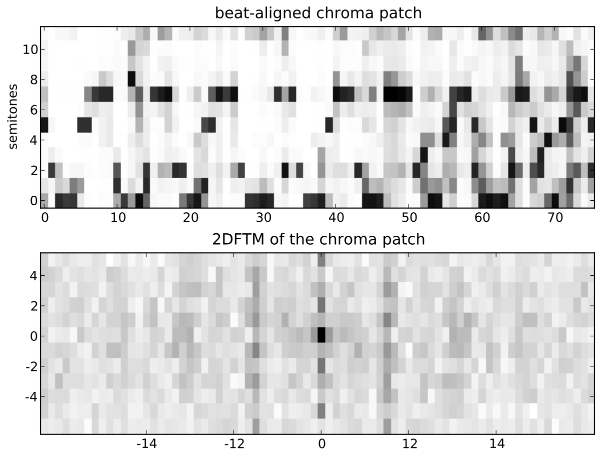


Figure 4. 2DFTM over a beat-synchronous chroma patch. Bins have been shifted so the maximum energy is in the center and magnitude values have been raised to the power 0.5 for visualization purposes. In the 2DFTM, the darker columns at -9 and $+9$ can be explained by the repetitive pattern in time in the chroma patch (bottom semitone) whose period is approximately 9 beats.

3.3 Rotation invariance

Analyzing nearest neighbors or clusters of chroma patches helps understand the properties of a given representation. Quantizing the chroma vector can be a useful step in an algorithm [20], but it can also be a goal on its own with the hope of seeing meaningful music patterns emerge [4]. The use of the 2DFTM introduces an interesting invariance, not only in key as in [4], but also in time, since small time shifts within a large patch correspond primarily to phase shifts in the Fourier components, with only slight changes in magnitude arising from edge effects. Figure 5 shows, for a randomly-chosen chroma patch at the top-left of the figure, the nearest neighbors obtained from the 2DFTM representation. For visualization purposes, we used 16-beat patches for this experiment. The result is noisy, but we see a clear square wave pattern that is repeated with different onsets in the first three neighbors.

3.4 Comparison with Similar Previous Approaches

Our method resembles those of [14,18], but some steps are simplified in view of the size of the collection at hand. In [14], instead of beat-aligned chromagrams, the authors use two-dimensional autocorrelation, then, for each semitone, take 17 samples spaced logarithmically in time (to normalize tempo) building a 12x17 feature matrix for each song. Autocorrelation is the Fourier transform of the Fourier Transform magnitude we use.

[3] *fftshift* in MATLAB, *scipy.fftpack.fftshift* in Python

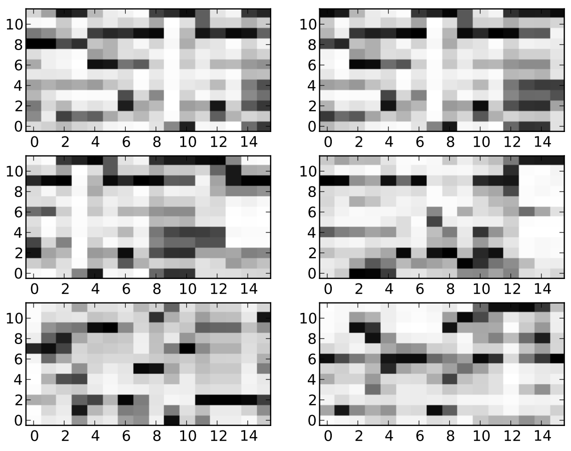


Figure 5. Neighbors of the upper left patch using 2DFTM representation (including raising values to power 1.96). The square signal shape, shifted in time, is visible in the first two neighbors (upper right, middle left) and partially in the middle right neighbor.

Compared to [18], we do not extract melodic lines, and we represent each track with an aggregate patch rather than storing many overlapping patches. We use PCA as the domain for comparison to improve generalization. Marolt describes time-shift invariance of the 2DFTM as an issue, but for us it is the main attraction. We found 75 beat patches to be most useful compared to the optimum at 32 in [18], perhaps because of PCA. A main emphasis of his work is the use of LSH, but reducing each track to a single pattern makes such indexing unneccessary, even for our much larger database.

4. COVER SONG RECOGNITION

Our goal is to find cover songs among mostly western pop music. The experiments are conducted on the SecondHandSongs dataset (SHS). The SHS was created by matching the MSD with the online database[4] that aggregates user-submitted cover songs. The SHS training set contains $12,960$ cover songs from $4,128$ cliques (musical works), and the testing set has $5,236$ tracks in 726 cliques. Note that we use the term "cover" in a wide sense. For instance, two songs derived from the same original work are considered covers of one another.

4.1 Method

We will represent each song by a vector of fixed-length, defining a point in Euclidean space. The closer two feature vectors lie in this space, the more likely the songs are covers. The steps to compute this feature vector for a song are summarized as follows:

1. obtain chroma from the MSD
2. resample onto beat grid
3. apply power-law expansion
4. extract FFT patches and keep magnitude
5. take the median
6. apply PCA

[4] www.secondhandsongs.com

The first step is done using the code provided with the MSD. One can also call The Echo Nest API [7] if starting from audio (see Subsection 4.4). Beat estimation is also provided in the MSD, and the second step is mostly a linear scaling. The third step, in which we enhance the contrast between strong and weak chroma bins by raising the values to an exponent, was determined to be useful in our experiments. In the fourth step, we take the 2DFTM as shown in Figure 4 for every 75-beat long chroma patch. In the fifth step we keep the median, within each bin, across all the transformed patches. Finally, in the sixth step, we use PCA to reduce the dimensionality. PCA is done without normalization, the data is simply centered to zero and rotated. In our experiments PCA was computed on 250K songs that were not identified as covers (i.e., in neither the SHS train or test sets).

Many of the parameters above were chosen empirically. We do not present the list of parameters we tried for lack of space and the little interest they carry. Simply note that the following parameters were chosen based on their ability to identify 500 train covers (see Subsection 4.3, first experiment):

- patches of size 75 beats, we tried number of beats ranging from 8 to 100;
- median as opposed to average;
- raising to the power 1.96, we tried values between 0.25 and 3.0.

The number of PCs we keep after PCA is also a parameter, but choosing the best one is more difficult. The number of PCs is a trade-off between accuracy and feature vector size (and hence speed). We believe 50 is the "best" trade-off, but we report results for other numbers of PCs. Still regarding PCA, since we use chroma patches of 75 beats, we have $12 \times 75 = 900$ principal components. Note that half of the components (450) are redundant due to the symmetry in the 2DFTM, and have a variance of zero associated to them.

4.2 Reference methods

We compare our algorithm to other methods, at different scales depending on the speed of the algorithm. We start with our previous work [2], the only reported result on the SHS to our knowledge. Also taken from that work, we report again a comparison with the method from [8].

We also test a DTW-based method based on [24] using code from S. Ravuri[5]. This is more of a sanity check than a full comparison; the authors in [24] used up to 36 semitones instead of the 12 we possess, we did not re-estimate the DTW parameters, etc. It is likely that the full system from [24] outperforms our method, the problem being the execution time which is prohibitive on a large scale. In our implementation, each DTW comparison takes on the order of 10 ms. One query on the MSD would therefore take

[5] http://infiniteseriousness.weebly.com/cover-song-detection.html

about 2.7 hours. Thus we do this comparison only on our 500 binary queries.

Finally, we compare with pitch histograms, a feature that was suggested for music identification in [25]. The pitch histogram of a song is the sum of the energy in the 12 semitones normalized to one. In our case, we compute it from beat-aligned chroma features. This feature is not powerful enough to perform cover recognition on its own, but it gives an idea of how much more information our method can encode in a $\sim$ 10-dimensional feature.

4.3 Experiments

Method	accuracy
random	50.0%
pitch hist.	73.6%
correlation	76.6%
DTW	80.0%
jcodes 1 [2]	79.8%
jcodes 2 [2]	77.4%
2DFTM (full)	82.0%
2DFTM (200 PC)	82.2%
2DFTM (50 PC)	**82.2%**
2DFTM (10 PC)	79.6%
2DFTM (1 PC)	66.2%

Table 1. Results on 500 binary tasks. PC is the number of principal components we retain after PCA. Empirically, 50 PC appear to be the best trade-off between accuracy and size.

We follow the methodology of [2] where the parameters are first tuned on a subset of 500 binary tasks created within the SHS training set (we use the same 500 queries). The goal is: Given a query song A and two songs B and C, find which of B or C is a cover of A. The result is the percentage of trials where the algorithm succeeds. We then present the results testing on the training set, mostly as a sanity check. Finally, we report result on the SHS test set using the full MSD.

In [2], the main reported result was the average rank of a known cover given a query. For instance, on 1M songs, picking at random would give 500K. We again report this measure to permit comparison, but for practical purposes this number may be misleading since it is dominated by the most difficult covers, of which there will always be a number, and hides differences in performance near the top of the ranking. We now prefer to report results in terms of mean average precision (meanAP), which puts emphasis on results that rank high, i.e., the covers that are in the top k results. present the recall curves for a few algorithms (see Figure 6).

As we see in Table 3, the method based on our 2DFTM representation provides a significant improvement over the method in [2] for both measures. In particular, based on meanAP, many more covers are ranked in the first few hundred results, which makes it much more valuable in a commercial system. Note that a second, slower step could be applied to the top k results, k being 1K or 10k, similar to the progressive refinement in [27]. A good candidate for this second step would be the full system of [24].

Method	average rank	mean AP
pitch hist.	4,032.9	0.01851
2DFTM (full)	3,096.7	0.08912
2DFTM (200 PC)	3,005.1	**0.09475**
2DFTM (50 PC)	**2,939.8**	0.07759
2DFTM (10 PC)	3,229.3	0.02649
2DFTM (1 PC)	4,499.1	0.00186

Table 2. Results on the training set (12,960 songs). For average rank, lower is better. For meanAP, higher is better.

Method	average rank	mean AP
random	500,000	$\sim$ 0.00001
pitch hist.	268,063	0.00162
jcodes 2	308,370	0.00213
2DFTM (200 PC)	180,304	**0.02954**
2DFTM (50 PC)	**173,117**	0.01999
2DFTM (10 PC)	190,023	0.00294
2DFTM (1 PC)	289,853	0.00003

Table 3. Results on 1M songs. For average rank, lower is better. For meanAP, higher is better.

Using the system with 50 principal components, Figure 6 shows us that more than half of the covers are ranked in the top 100k and more than a quarter of them are in the top 10k. For 112 queries, the first song returned was a known cover. We ignore here the songs that might have ranked second or third after other known covers. This includes the pairs *(Hayley Westenra, Kate Bush)* performing "Wuthering Heights" and *(The Boomtown Rats, G4)* performing "I don't like Mondays", matches considered *easy* in [2].

In terms of speed, with a 50-dimensional vector per song, ranking all one million for all 1,726 test covers in Python took 1h 46min on a machine with plenty of RAM. This is around 3-4 seconds per query without any optimization or parallelization. Compared to the 2.7 hours of the DTW method, that is a $\sim$ 2,500x speedup.

4.4 Out of collection data

Using audio as the query input with the SHS is a challenge, beat-synchronous features relies on a consistent beat tracker. Fortunately, The Echo Nest provides an open API which will convert any uploaded audio into the format provided in the Million Song Dataset. We experimented with this using cover songs found on YouTube. For instance, the song "Summertime" by *Ella Fitzgerald and Louis Armstrong* [6] was correctly associated with a cover version by

[6] `http://www.youtube.com/watch?v=MIDOEsQL7lA`

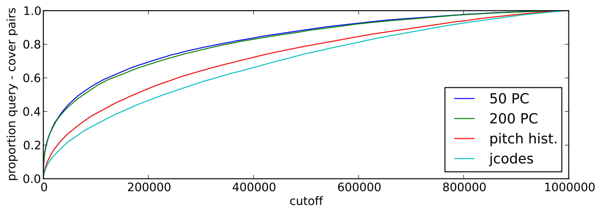


Figure 6. Recall using (*query, cover*) pairs on the full million songs for different systems. Legend is in order from best (upper left corner) to worst.

Joshua Redman (first match). The *Ella Fitzgerald and Louis Armstrong* version present in the SHS was found at rank 9. The fact that this was not the first match might be explained by the lower audio quality on YouTube, or it could be a different version.

5. CONCLUSION

The 2DFTM allows us to pose the search for cover songs as estimating an Euclidean distance. We show that this representation exhibits some nice properties and improves the state-of-the-art on large-scale cover song recognition. Furthermore, obtaining good results using patches of 75 beats suggests an easy way to include more time dependency in the harmonic representation. Future work will look into the usefulness of this representation for other tasks, such as music tagging and segmentation. Finally, all our code is available online[7].

6. ACKNOWLEDGMENTS

The authors would like to thank The Echo Nest and SecondHandSongs for making the MSD and the SHS available respectively. Thanks to S. Ravuri for his DTW code. TBM is a NSERC PGD scholar. This work has been supported by a gift from Google and by the NSF under grant IIS-0713334.

7. REFERENCES

[1] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proceedings of ISMIR*, 2011.

[2] T. Bertin-Mahieux and D.P.W. Ellis. Large-scale cover song recognition using hashed chroma landmarks. In *Proceedings of WASPAA*, New Platz, NY, 2011.

[3] T. Bertin-Mahieux, G. Grindlay, R. Weiss, and D. Ellis. Evaluating music sequence models through missing data. In *Proceedings of ICASSP*, 2011.

[4] T. Bertin-Mahieux, R. Weiss, and D. Ellis. Clustering beat-chroma patterns in a large music database. In *Proceedings of ISMIR*, 2010.

[5] M. Casey and M. Slaney. Fast recognition of remixed music audio. In *Proceedings of ICASSP*. IEEE Signal Processing Society, 2007.

[6] J.S. Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[7] The Echo Nest Analyze, API, `http://developer.echonest.com`.

[8] D. Ellis and G. Poliner. Identifying cover songs with chroma features and dynamic programming beat tracking. In *Proceedings of ICASSP*. IEEE Signal Processing Society, 2007.

[9] T. Fujishima. Realtime chord recognition of musical sound: a system using common lisp music. In *Proceedings of the International Computer Music Conference (ICMC)*, 1999.

[10] E. Gómez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, 18(3):294–304, 2006.

[11] R.C. González and R.E. Woods. *Digital image processing*. Pearson/Prentice Hall, 2008.

[12] Ö. İzmirli and R.B. Dannenberg. Understanding features and distance functions for music sequence alignment. In *Proceedings of ISMIR*, 2010.

[13] T. Jehan. *Creating music by listening*. PhD thesis, Massachusetts Institute of Technology, 2005.

[14] J.H. Jensen, M.G. Christensen, and S.H. Jensen. A chroma-based tempo-insensitive distance measure for cover song identification using the 2d autocorrelation function. MIREX cover song identification contest, 2008.

[15] N. Jiang, P. Grosche, V. Konz, and M. Müller. Analyzing chroma feature types for automated chord recognition. In *Proceedings of the 42nd AES Conference*, 2011.

[16] S. Kim and S. Narayanan. Dynamic chroma feature vectors with applications to cover song identification. In *Multimedia Signal Processing, 2008 IEEE 10th Workshop on*, pages 984–987. IEEE, 2008.

[17] F. Kurth and M. Müller. Efficient index-based audio matching. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):382–395, 2008.

[18] M. Marolt. A mid-level representation for melody-based retrieval in audio collections. *Multimedia, IEEE Transactions on*, 10(8):1617–1625, 2008.

[19] R. Miotto, N. Montecchio, and N. Orio. Content-based cover song identification in music digital libraries. In *Digital Libraries*, pages 195–204. Springer, 2010.

[20] R. Miotto and N. Orio. A music identification system based on chroma indexing and statistical modeling. In *Proceedings of ISMIR*, 2008.

[21] M. Müller, F. Kurth, and M. Clausen. Audio matching via chroma-based statistical features. In *Proceedings of ISMIR*, London, UK, 2005.

[22] L. Oudre, Y. Grenier, and C. Févotte. Chord recognition by fitting rescaled chroma vectors to chord templates. *IEEE Transactions on Audio, Speech and Language Processing*, 19(7):2222 – 2233, Sep. 2011.

[23] J. Serrà. *Identification of versions of the same musical composition by processing audio descriptions*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2011.

[24] 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(6):1138–1151, 2008.

[25] G. Tzanetakis, A. Ermolinskyi, and P. Cook. Pitch histograms in audio and symbolic music information retrieval. *Journal of New Music Research*, 32(2):143–152, 2003.

[26] A. Wang. An industrial strength audio search algorithm. In *Proceedings of ISMIR*, 2003.

[27] Y. Yu, M. Crucianu, V. Oria, and L. Chen. Local summarization and multi-level LSH for retrieving multi-variant audio tracks. In *Proceedings of the seventeen ACM international conference on Multimedia*, pages 341–350. ACM, 2009.

[7] `http://www.columbia.edu/~tb2332/proj/coversongs.html`

CREATING GROUND TRUTH FOR AUDIO KEY FINDING: WHEN THE TITLE KEY MAY NOT BE THE KEY

Ching-Hua Chuan
University of North Florida
School of Computing
c.chuan@unf.edu

Elaine Chew
Queen Mary, University of London
Centre for Digital Music
elaine.chew@eecs.qmul.ac.uk

ABSTRACT

In this paper, we present an effective and efficient way to create an accurately labeled dataset to advance audio key finding research. The MIREX audio key finding contest has been held twice using classical compositions for which the key is designated in the title. The problem with this accepted practice is that the title key may not be the perceived key in the audio excerpt. To reduce manual annotation, which is costly, we use a confusion index generated by existing audio key finding algorithms to determine if an audio excerpt requires manual annotation. We collected 3224 excerpts and identified 727 excerpts requiring manual annotation. We evaluate the algorithms' performance on these challenging cases using the title keys, and the re-labeled keys. The musicians who aurally identify the key also provide comments on the reasons for their choice. The relabeling process reveals the mismatch between title and perceived keys to be caused by tuning practices (in 471 of the 727 excerpts, 64.79%), and other factors (188 excerpts, 25.86%) including key modulation and intonation choices. The remaining 68 challenging cases provide useful information for algorithm design.

1. INTRODUCTION

The typical trend in technology development is for systems proposed later to outperform earlier ones, but this does not seem to have been the case for audio key finding, judging by the results of the MIREX audio key finding contest. The first MIREX audio key finding contest was held in 2005, and the second contest took place six years later. The same dataset was used in the two contests, and based on the numbers, the systems in MIREX 2011 seem to have performed worse than the ones in 2005 on average. This points to the fact that the statistics of the contest results alone have not provided sufficient information for future researchers to develop better systems. If the goal of the contest is to move the research community forward, a detailed examination of the results is required.

An effective way to improve audio key finding is to examine the cases in the dataset for which most systems have difficulties. For this paper, we constructed a dataset with 3324 music audio recording excerpts, 2.6 times the number in the MIREX dataset. It is worth noting that this dataset created from actual music recordings is distinct from the MIREX dataset synthesized from MIDI files. We implemented five existing audio key finding systems and tested them on the dataset. Using the title key as ground truth, let the confusion index, I, be the number of systems that disagree with this ground truth. We then extracted a subset of the data consisting of excerpts for which no more than two systems reported correct answers, i.e. $I \geq 3$. This subset, called the challenging set, was re-examined by three professional musicians and their keys manually labeled. By comparing the relabeled keys with the title keys, we observe reasons why the excerpt's perceived key might be different from the title key. We also present the musicians' comments about their annotations to show the factors that impact audio key finding. Finally, we describe some controversial audio key finding cases for which we received three conflicting answers.

The paper is organized as follows. Section 2 provides the background of MIREX audio key finding contests. Section 3 presents the five audio key finding systems implemented for the study. In Section 4, we describe the experiment design, followed by a detailed examination of the experiment results. We state our conclusions and suggestions for future work in Section 5.

2. BACKGROUND

In MIREX 2005, Chew, Mardirossian and Chuan proposed contests for symbolic and audio key finding [7]. For audio key finding, a wave file is given as input and one answer including a key name (for example, "C") and a mode (such as "major") is expected as output. The ground truth used in the contest is the key defined by the composer in the title of the piece, as is the practice in key finding research. Each output key is compared to the ground truth and assigned a score as follows: 1 point if the output is identical to the ground truth, 0.5 if they are a perfect fifth apart, 0.3 if one is the relative major/minor of the other, and 0.2 if one is the parallel major/minor of the other. In the contest, 1252 audio files synthesized from MIDI were used as the test dataset, consisting of symphonies from various time periods. The best system achieved an 87% correct rate, with a composite score of 89.55% [8]. The second audio key finding/detection contest was held in 2011 [9]. The same dataset was used and same evaluation method was employed. The best system achieved a 75% correct rate and weighted score of 82%.

3. AUDIO KEY FINDING SYSTEMS

In this section we first describe the general structure of an audio key finding system, then the five systems implemented in this study. Systems that rely on training data are not implemented because the title key (ground truth) may not be the key of the excerpt.

3.1 A General Structure

Figure 1 illustrates a general structure of an audio key finding system. The system can be divided to two major parts as shown in the dashed boxes. The components in the left dashed box are designed to transform low-level acoustic features such as frequency magnitude into high-level music information such as pitches or pitch classes. Some audio key finding systems start with some preprocessing of the data, such as the removal of noise and silence. The next two steps, windowing and Fast Fourier Transform (FFT), indicate a basic approach for spectral analysis. After spectral analysis, the step labeled Pitch Class Profile (PCP) generation converts the frequency spectrum information into a pitch class distribution called a pitch class profile. This is often the step where most audio key finding systems differ.

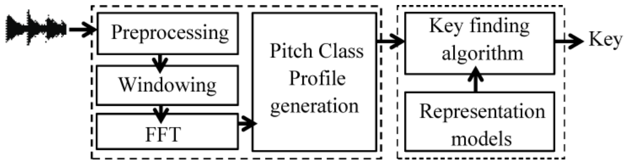


Figure 1. General structure of audio key finding systems.

After a pitch class profile is generated, it is then compared with 24 predefined key templates or profiles, 12 major and 12 minor, to determine the key in the audio excerpt. This step is shown in the right dashed box in Figure 1, consisting of two components: a key finding algorithm and a representation model of keys. A representation model provides a typical pitch class distribution for a specific key. The key profiles produced by the representation model are then used to determine the key in a key finding algorithm. For example, a simple key finding algorithm calculates the correlation between the pitch class profile of the audio excerpt and the 24 key profiles, and the key profile that reports the highest correlation value is selected as the key.

In the following sub sections, we describe the audio key finding systems implemented in this study in detail, emphasizing the uniqueness of each system.

3.2 Krumhansl's and Temperley's Key Profiles

Tonality has been studied extensively not only in music theory, but also from many other perspectives. In 1986, Krumhansl and Schmuckler [6] developed a widely accepted model called the probe tone profile method (referred to as the K-S model for convenience), which constructs pitch class profiles for major and minor keys by using user ratings from probe tone experiments. In 1999, Temperley [10] improved upon the K-S model by modifying the key profiles to emphasize the differences between diatonic and chromatic scales. Temperley also adjusted the weights of the forth and seventh pitches so as to differentiate between keys with highly similar pitch class signatures. The key profiles generated by the K-S model and Temperley are shown in Figure 2.

These key profiles can be directly used to build a symbolic key finding system. In this study, we added an audio signal processing module to these key profiles, and created two base audio key finding systems for comparison. The audio signal processing module consists of a silence removal preprocessing step, rectangular windowing with non-overlapped frames, FFT, and generation of a PCP using a uniform weighting function across the frequency range of a pitch and also for all pitches folded into 12 pitch classes.

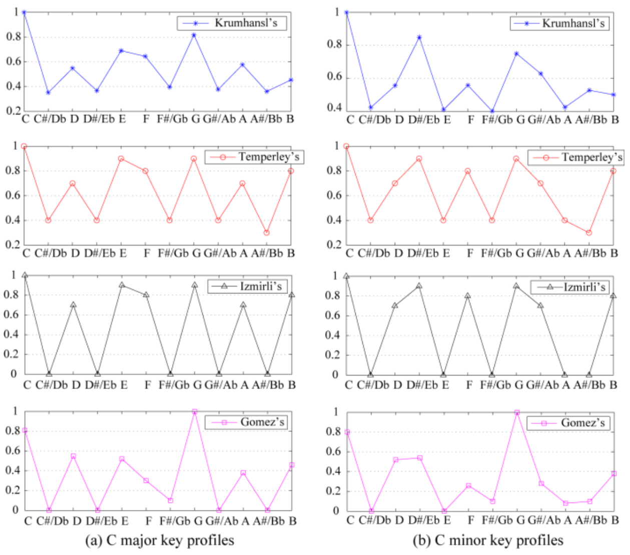


Figure 2. C major and C minor key profiles proposed by Krumhansl, Temperley, Izmirli and Gómez.

3.3 Izmirli's System

Izmirli proposed a template-based correlation model for audio key finding in [5]. During the PCP generation step shown in Figure 1, called chroma template calculation in Izmirli's system, pitches are weighted using a decreasing function that gives low-frequency pitches more weight. In Izmirli's system, he constructed key templates from monophonic instruments samples, weighted by a combination of the K-S and Temperley's modified pitch class profiles, as shown in Figure 2. Izmirli's system also tracks the confidence value for each key answer, and the global key is then selected as the one having the highest sum of confidence values over the length of the piece.

3.4 Gómez's HPCP

In [5], Gómez detected pitches using three times the standard resolution of the pitch frequency spectrum of the FFT method, and distributed the frequency values among the adjacent frequency bins using a triangular weighting function to reduce boundary errors. A Harmonic Pitch Class Profile (HPCP) is generated as input to the key finding algorithm, using a modified version of Krumhansl's key templates as shown in Figure 2.

3.5 Chuan and Chew's FACEG

In [3], Chuan and Chew proposed an audio key finding system called Fuzzy Analysis Center of Effect Generator (FACEG). A fuzzy analysis technique is used for PCP generation using the harmonic series to reduce the errors in noisy low frequency pitches. The PCP is further refined periodically using the current key information. The representation model used in the system is Chew's Spiral Array model [1, 2], a representation of pitches, chords, and keys in the same 3-dimensional space with distances reflecting their musical relations. The Center of Effect Generator (CEG) key finding algorithm determines key in real-time: an instantaneous key answer is generated in each window based on past information. It is the only model amongst the ones considered with pitch spelling.

4. EXPERIMENT DESIGN

In this section we describe the manner in which the dataset is prepared, and the experiment design for exploring the reasons for the challenging cases.

4.1 Data Collection and Selection

The dataset used in this study is provided by Classical KUSC, a classical public radio station. The entire dataset consists of over 40,000 audio recordings of classical performances. For this study, we selected compositions by Bach, Mozart and Schubert. We chose these three composers' work because (1) tonality is generally more clearly defined in these pieces than in more recent compositions; and (2) the composers represent three different styles with distinguishable levels of tonal complexity. We further refined the dataset by filtering out the recordings that do not have key information in the title. For multi-movement works, we used only the first and last movement, because they are generally in the title key.

As a result, the dataset we used in this study consists of 1662 recordings of varying lengths. Similar to the evaluation procedure in the MIREX 2005 and 2011 audio key finding contests, we extracted two excerpts from each recording: one containing the first 15 seconds and the other representing the last 15 seconds of the recording. We only reserved the beginning and end sections of a recording because these two sections are more likely to be in the key shown in the title. As a result, we ended up with a total of 3324 different excerpts in the experiment.

4.2 Evaluation Method

To improve the performance of existing audio key finding systems, some more detailed examination of the challenging cases is necessary. An excerpt is considered challenging if no more than two systems out of the five implemented reported the key identical to the one in the title. A challenging set was thus built from such cases.

The excerpts from the challenging cases were given to two professional musicians for key annotations. During the process, the two musicians were provided with the 15-second long excerpts instead of the entire pieces, to ensure that they have the same acoustic information as the systems. No key labels or titles were revealed to the musicians. The only information other than the audio excerpt provided is the name of the composer.

The musicians were asked to write down one answer as the global key for each excerpt, based on the 15 seconds they heard. They were also asked to comment on the reasons behind their answers, particularly for excerpts that they felt were difficult to annotate. When the key annotations by the two musicians differed, the excerpt was given to a third professional musician for relabeling. The final relabeled key was determined by majority vote from the three annotations.

5. EXPERIMENT RESULTS

5.1 Results Using Title Keys as Ground Truth

Out of the 3324 excerpts, there were 727 excerpts (21.87%) for which no more than two systems reported answers identical to the title keys, i.e. $I \geq 3$. We focused on these 727 excerpts, the challenging cases in this study, to examine the difficulties most key finding systems encounter. Table 1 shows the distribution of the challenging set, in absolute numbers and as a percentage of the number of excerpts we considered by each composer.

Composer	Total # of recordings	Challenging set (first 15 sec)	Challenging set (last 15 sec)
Bach	553	245 (44.30%)	244 (44.12%)
Mozart	873	75 (8.59%)	98 (11.23%)
Schubert	236	24 (10.17%)	41 (17.37%)

Table 1. Details of the entire data set and the challenging set by Bach, Mozart and Schubert.

We divided the reported key into 9 categories based on its relation to the title key: correct, dominant (Dom), subdominant (SubD), parallel major/minor (Par), relative major/minor (Rel), same mode with the root one half-step higher (Mode +1), same mode with the root one half-step lower (Mode – 1), same mode but not in the previous categories (Mode Others), and the rest of relations not included in any of the previous categories (Others).

Figure 3 shows the results for the Bach challenging set in (a) first 15 seconds and (b) last 15 seconds respectively. It can be observed that most of the incorrect answers reported by the systems fall into the last three categories, especially in the category (Mode – 1), indicating that tuning may be an issue in key finding for Bach's pieces.

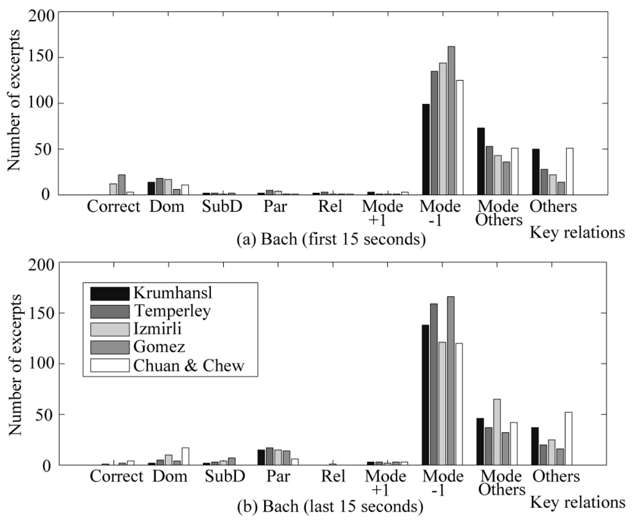


Figure 3. Key finding results for the challenging Bach dataset using the title key as ground truth.

Figure 4 shows the results for the Mozart challenging set in the (a) first and (b) last 15 seconds, respectively. In Figure 4 (a), similar to the results in Figure 3 (a), the last three categories account for the majority of the results in the first 15 seconds. However, unlike Figure 3 (b), the parallel major/minor (Par) category accounts for a significant proportion of the results in Figure 4 (b). The reported keys are also more evenly distributed than in Figure 3.

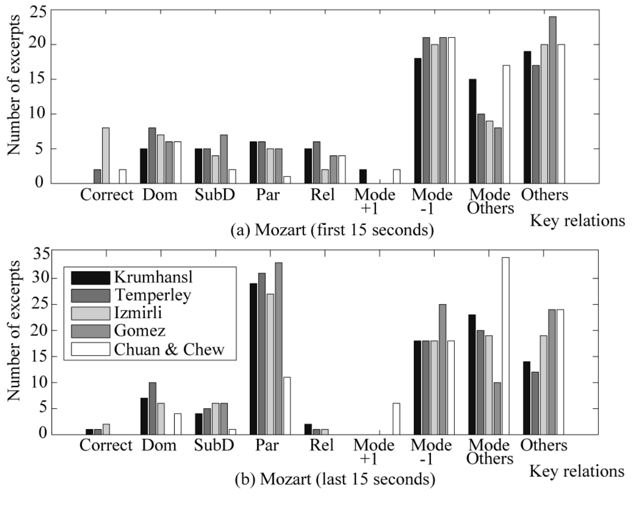


Figure 4. Key finding results for the challenging Mozart dataset using the title key as ground truth.

Figure 5 shows the results for the Schubert challenging set in the (a) first and (b) last 15 seconds, respectively. The results of Schubert challenging set are more similar to Mozart's than Bach's, as the results are more evenly distributed in the first 15 seconds and the parallel major/minor dominates in the last 15 seconds. One distinct feature observed in the Schubert results is that the (Mode – 1) category is much less significant.

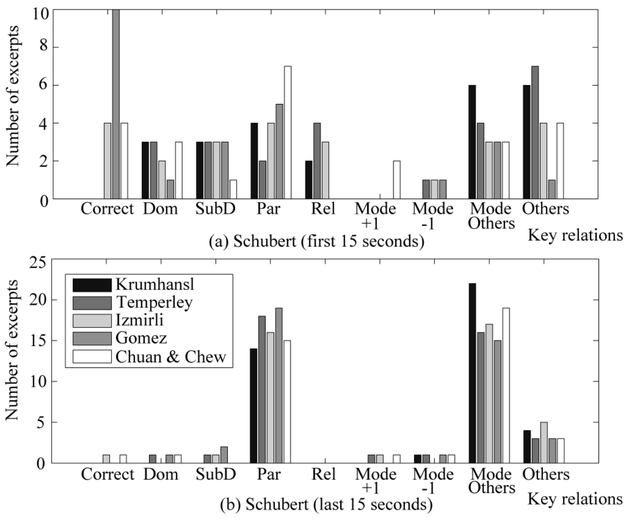


Figure 5. Key finding results for the challenging Schubert dataset using the title key as ground truth.

5.2 Results Using Re-labeled Keys as Ground Truth

Table 2 shows the statistics of re-labeled keys in relation to title keys. The tuning category consists of re-labeled keys one half step away from title keys, while the other category includes relabeled keys that are neither identical to title keys nor in the tuning category.

Composer	First 15 seconds		Last 15 seconds	
	tuning	other	tuning	other
Bach	183 (74.7%)	36 (14.7%)	182 (74.6%)	54 (22.1%)
Mozart	48 (64%)	16 (21. 3%)	55 (56.1%)	38 (38. 8%)
Schubert	1 (4.2%)	8 (33. 3%)	2 (4.9%)	36 (87.8%)

Table 2. Relations between title keys and re-labeled keys.

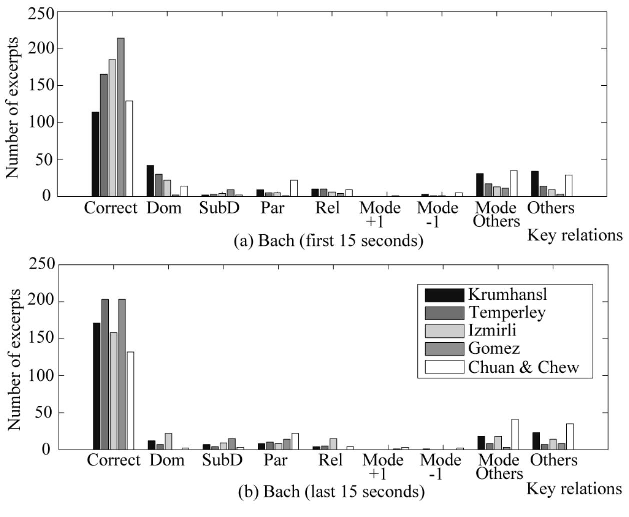


Figure 6. Key finding results for the challenging Bach dataset using the re-labeled key as ground truth.

Figure 6 shows the results for the Bach challenging set in the (a) first 15 seconds and (b) last 15 seconds using the relabeled keys as ground truth. Comparing the results in Figure 3 with those in Figure 6, it is clear that many of the recordings of Bach's compositions are not tuned to modern definitions of the title key. Pitches ranged from one quarter to one half-step lower than what one might expect in modern tuning. Therefore, the cases labeled as (Mode – 1) in Figure 3 could be considered correct. This also points to the importance of verifying the title keys manually for the audio key finding. However, it is debatable whether the key should be relabeled based on modern tuning. For example, a recording may be recognized as being in the key of B major according to modern tuning, but B major is a very uncommon key in Baroque music and some musicians still prefer to call it C major despite the flattened tuning.

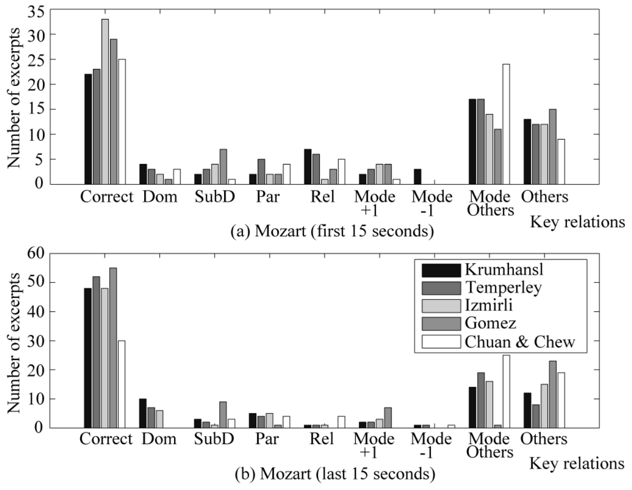


Figure 7. Key finding results for the challenging Mozart dataset using the re-labeled key as ground truth.

Figure 7 shows the results for the Mozart challenging set using the relabeled keys as ground truth in the (a) first 15 and (b) last 15 seconds. Observe that the number of correct answers is increased, indicating that the first and last 15 seconds of these pieces are actually in a key other than the title key. By comparing Figure 7 (a) and Figure 4 (a), we observe that the increase in correct answers in Figure 7 (a) mainly results from decreasing numbers in the four categories: Mode – 1, Others, Dominant (Dom) and Parallel (Par). This shows that for Mozart, a piece may start in a related parallel major/minor or even a foreign key. For the last 15 second excerpts, a piece may end in a parallel major/minor key. The tuning problem, indicated by the (Mode – 1) category, can still be observed in Mozart's recordings in both the first and last 15 seconds.

Figure 8 shows the results on the Schubert challenging set in the (a) first 15 and (b) last 15 seconds. The number of correct answers does not increase much in Figure 8 (a) comparing to Figure 5 (a), and the increment in Figure 8 (a) is the result of decrement in the Parallel major/minor (Par) category. When the relabeled keys are used as the ground truth, almost all the systems recognize the keys correctly in the last 15-second excerpts as shown in Figure 8 (b). This result shows that Schubert's pieces may end in the parallel major/minor key, or even some more distant keys in the same mode.

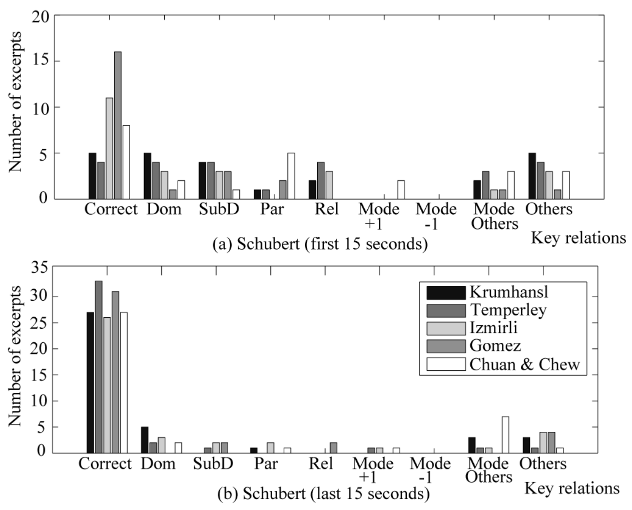


Figure 8. Key finding results for the challenging Schubert dataset using the re-labeled key as ground truth.

5.3 Musicians' Comments and Case Studies

In this section we present the musicians' comments alongside their answers, and excerpts where they disagreed with each other.

The musicians were encouraged to write down comments with their answers but were not restricted in terms of the words they can use. Table 3 shows the most frequently used keywords and their meanings.

Keywords	Meanings	Num. of occurrence
sharp/flat	Notes sound sharp/flat compared to modern tunings	79
easy	Clear V-I chord progression	66
picardy 3rd	A piece in a minor key that ends in the parallel major	28
modulation	A piece changes from one key to another	16
tough/tricky	Difficult to determine the key, mostly due to missing cadence	15
cadence	Cadence cut off; cadence spotted in the middle of the piece; misleading cadence	4

Table 3. Keywords with meanings and number of occurrences in musicians' comments.

Among 727 excerpts, there are only 8 excerpts in which all three musicians disagreed on the key. Table 4 gives the details of the 8 excerpts, the musicians' annotated keys, and our notes on why these excerpts might have confused the annotators.

6. CONCLUSIONS AND FUTURE WORK

In this paper we presented an approach to effectively and efficiently develop a well-annotated dataset for audio key finding. Having a well-annotated dataset is essential for any kind of algorithm testing and development, but it is very time-consuming to create one with numerous examples. In this paper we implemented five audio key finding systems, and used them to select the examples that require manual examination. Three professional musicians re-labeled the keys for these difficult cases.

Composer	Recording (excerpt)/ Performer	Title key	Relabeled keys
Bach	Cello Suite #3 BWV 1009 (last 15 secs)/Yo-Yo Ma	C major	C major, B major, C minor
	Cello. Briefly in minor mode at the beginning, but ends unequivocally in C major; pitches flat.		
Schubert	Moments Musicaux: #6 Op 94 (last 15 secs)/David Fray	Ab major	Eb major, G# major, A major
	Piano. In Ab major; tuning sharp. Annotator 1 misled by Bb's in beginning.		
Mozart	String Quartet #16 K428 (first 15 secs)/Quartetto Italiano	Eb major	F minor, C minor, Eb major
	String Quartet. Chromatic start and notes following led to ambiguity in mode; ends clearly in Eb major.		
Mozart	String Quartet #16 K428 (first 15 secs)/Quatuor Mosaiques	Eb major	B minor, D major, Eb minor
	Same piece as above; annotations completely different.		
Mozart	String Quartet #19 K465 (first 15 secs)/ Quatuor Mosaiques	C major	C minor, B minor, unsure
	String Quartet. In C but Eb in vln 2 and flat A in vln 1 (an intonation choice) led to perceived minor mode.		
Mozart	Gran Partita Serenade K361 (last 15 secs)/Octophorus	B major	Bb major, D major, A major
	Strings. Tuning flat, which explains the Bb and A.		
Mozart	Symphony #22 K162 (last 15 secs)/Amsterdam Baroque Orchestra	C major	B major, E major, D major
	Orchestra. Tuning flat, which explains the B.		
Mozart	Requiem K626 (last 15 secs)/Vienna Philharmonic	D minor	C major, unsure, F major
	Voices/Str/Winds/Perc. Flat; insufficent information.		

Table 4. Information of the excerpts where three musicians disagree on the key.

By examining the relabeled keys, we discovered potential causes for the difficulties and make the following observations and recommendations:

(a) **tuning:** In recorded performances, different tuning or intonation choices can cause confusion. Evaluations could either account for all possible categorical key name interpretations (e.g. flat C might be interpreted as B), or allow for tuning (and letter name) independent key finding, for example by requiring systems to locate the most stable tone.

(b) **modulations:** some excerpts may not be entirely in one key, modulating midstream or ending with a Picardy 3rd. These excerpts could either be removed, or more nuanced ground truth created with key change annotations.

(c) **missing cadences:** a key is theoretically established when a complete cadence confirms its identity. Many excerpts from the first 15 seconds of pieces may not have these cadences. Care could either be taken to make sure these cadences exist in the evaluation samples, or the scoring system could account for levels of difficulty of assessing key based on annotators' notes.

The established dataset and annotations, and the process of collecting them, will benefit the audio key finding community and MIREX contests. While we cannot publicly share the music files, we will post the results of the annotations online. Future plans include augmenting the dataset with automatically generated key labels.

7. ACKNOWLEDGEMENTS

We like to thank the annotators, Fred Bertino, Victor Hormilla, and Jaime Kaufman, for their time and effort.

8. REFERENCES

[1] E. Chew: *Towards a Mathematical Model of Tonality*, Doctoral dissertation, Department of Operations Research, Massachusetts Institute of Technology, Cambridge, Mass, USA, 2000.

[2] E. Chew: "Modeling Tonality: Applications to Music Cognition," in *Proc. of the 23rd Annual Meeting of the Cognitive Science Society*, pp. 206–211, Edinburgh, Scotland, UK, 2001.

[3] C.-H. Chuan and E. Chew: "Fuzzy Analysis in Pitch-Class Determination for Polyphonic Audio Key Finding," in *Proc. of the 6th International Conference on Music Information Retrieval*, pp. 296–303, London, UK, 2005.

[4] E. Gómez, "Tonal Description of Polyphonic Audio for Music Content Processing," INFORMS *Journal on Computing*, summer 2006, Vol. 18, No. 3, pp. 294–304, 2006.

[5] Ö. İzmirli, "Template Based Key Finding from Audio," in *Proc. of the International Computer Music Conference*, Barcelona, Spain, 2005.

[6] C. L. Krumhansl, "Quantifying Tonal Hierarchies and Key Distances," in *Cognitive Foundations of Musical Pitch*, chapter 2, pp. 16–49, Oxford University Press, New York, USA, 1990.

[7] MIREX 2005 Audio Key Finding Contest, www.music-ir.org/mirex/wiki/2005:Audio_and_Symbolic_Key

[8] MIREX 2005 Audio Key Finding Contest Results, www.music-ir.org/mirex/wiki/2005:Audio_Key_Finding_Results

[9] MIREX 2011 Audio Key Detection Contest, nema.lis.illinois.edu/nema_out/mirex2011/results/akd/index.html

[10] D. Temperley, "What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered," *Music Perception*, Vol. 17, No. 1, pp. 65–100, 1999.

UNDERSTANDING USER REQUIREMENTS FOR MUSIC INFORMATION SERVICES

Jin Ha Lee
The Information School
University of Washington
jinhalee@uw.edu

Nichole Maiman Waterman
The Information School
University of Washington
maiman@uw.edu

ABSTRACT

User studies in the music information retrieval and music digital library fields have been gradually increasing in recent years, but large-scale studies that can help detect common user behaviors are still lacking. We have conducted a large-scale user survey in which we asked numerous questions related to users' music needs, uses, seeking, and management behaviors. In this paper, we present our preliminary findings, specifically focusing on the responses to questions of users' favorite music related websites/applications and the reasons why they like them. We provide a list of popular music services, as well as an analysis of how these services are used, and what qualities are valued. Our findings suggest several trends in the types of music services people like: an increase in the popularity of music streaming and mobile music consumption, the emergence of new functionality, such as music identification and cloud music services, an appreciation of music videos, serendipitous discovery of music, and customizability, as well as users' changing expectations of particular types of music information.

1. INTRODUCTION

Understanding what kinds of music information services people use, how they use them, and what they expect from them is critical in designing successful services. We have seen a gradual increase in different types of user and usability studies in recent years. However, many of these studies are based on a limited number of subjects, and tend to employ analysis of qualitative research methods, like in-depth interviews or focus groups. While these kinds of studies can help uncover rich data about music users, large-scale user studies are also necessary in order to test the generalizability of results and to complement the insights obtained from smaller qualitative studies.

To fill this gap, we have conducted a large-scale user survey questioning people's music needs, uses, and music seeking and management behaviors. This survey is an extension of previous research conducted in 2004 by Lee and Downie [7]. The information we acquired through this new study can help improve our general understanding of music users and their behaviors, as well as how they have changed as compared to the 2004 survey results.

2. LITERATURE REVIEW

We conducted an extensive literature search in order to find out how many large-scale user studies exist in the MIR domain. Of the 87 studies discovered, only 6 involve more than 100 subjects (with the exception of studies analyzing user generated content such as queries/reviews). Ellis et al. [4] developed a web-based game named "MusicSeer," which collected over 6,200 responses; they found that "subjective artist similarities are quite variable between users," suggesting that the concept of a single ground truth may be problematic. Barrington et al. [1] studied 185 subjects and asked them to evaluate results from multiple music recommender systems. Both of these studies focused on highly specific ideas, such as responses about artist-to-artist relationships [4] or recommendation results [1], rather than general music behaviors.

Some studies dealt with particular organizations' user groups. Lai and Chan [5] surveyed 244 Hong Kong Baptist University Music Library users to improve understanding of their needs, usage patterns, and preferences toward various collections. The authors found that participants used scores and multimedia more frequently than other types of library materials, although they believed that electronic journal databases, books, and online music listening were also important to their academic and performance needs. In their survey of visitors to the Experience Music Project in Seattle, Maguire et al. [9] found that improving the user interface was the most important suggestion for changes to the museum's digital collection.

Other studies dealt with broader topics and more general user populations. Lesaffre et al. [8] collected 663 qualified survey responses to clarify the influence of demographics and musical background on how people describe music's semantic qualities. Their research listed several characteristics that average MIR system users likely would have, and found that gender had the most significant influence on music perception. Brinegar and Carpa [2] also surveyed 184 respondents on how they manage music across multiple devices, and provided empirical data on the sizes of user collections, the methods/reasons for synchronization, how users dealt with music loss, and so on. Lee and Downie [7] conducted a large-scale music survey in 2004 asking two groups of respondents (University of Illinois community, and general adult public) about their music information needs, uses, and search/browse patterns. Their analysis revealed the social aspects of music information seeking – that it can be a public and shared process, and many users felt positively towards reviews, ratings, and recommendations from other people. The authors also stressed the importance of providing context metadata (i.e., metadata on a music item's relationships with other items, and its associations with other works). Our study aims to add further insights into music users' behaviors; in particular, this paper focuses on discovering how people use currently available music services and why they favor them.

3. STUDY DESIGN

3.1 Study Population and Sampling

The design specifics of the 2004 and 2012 surveys are summarized in Table 1. For the 2004 survey, the candidate respondents for Group I were randomly selected from a list of students, faculty, and staff from the University of Illinois at Urbana-Champaign. For Group II, invitations to the survey were posted in music-related mailing lists/forums in order to recruit participants. For the 2012 survey, we posted invitations on mailing lists at University of Washington as well as music-related mailing lists. We also recruited participants through the authors' social media network such as Facebook, Twitter, and Google+.

	2004 Survey		2012 Survey
Study population	Group I: UIUC community	Group II: General population	UW community + General population
Sampling	Random	Convenience	Convenience
# questions	19	21	23
# responses	436	312	520

Table 1. Basic statistics of the surveys

In the 2004 survey we asked 19 questions for Group I and 21 questions for Group II (2 additional questions on job type and education level). The questions covered why, where, how, and how often users seek and obtain various kinds of music information; who they ask for help; how they use music information; what music-related websites/apps they use; and so on. The design of the 2012 survey was based on the previous survey to facilitate results comparison. The 2012 survey included 4 additional questions about how users manage physical and digital music collections, what devices they use to listen to music, and any comments related to the survey. In both surveys, there were follow up questions that were asked based on how users answered the main questions.

3.2 Limitation

One concern is that the different populations and sampling methods might affect the comparability of the results. For the 2004 survey, we were able to obtain a full list of all UIUC community members, and thus were able to randomly select participants. In the 2012 survey, it was not possible to obtain such a listing of the UW community for survey purposes, due to privacy concerns. When we compared the demographic information of respondents, [Table 2], there were in fact some differences. The average age was slightly higher for the 2012 survey respondents and the dominant gender was also different. However, most of the respondents did come from the United States for both surveys. In the article reporting the full survey results, we will be presenting the results controlling for these particular variables in order to see if there are significant differences between these sub-user groups. Nevertheless, it is important to be aware of this limitation in interpreting the findings and implications of this study.

	2004 Survey	2012 Survey
Age	Average: 30	Average: 37
Gender (excluding unanswered)	M (50.4%) F (46.1%)	Male (36.2%) Female (58.8%)
Geographic location	73.8% US	60.3% US

Table 2. Demographic information of respondents

4. DATA AND DISCUSSION

4.1 Overview

In this section, we present a detailed analysis of one of the open-ended questions, which asked about users' favorite music-related websites/applications. We also compare the responses we obtained for this question in 2004 and 2012, and present excerpts from users' responses and quantitative data on user responses from other relevant questions.

4.2 Summary of Results

4.2.1 Favorite Music-related Websites/Applications

The exact question asked was "*What are your favorite music-related websites or apps? What do you like about them?*" We received a total number of 237 responses from Group I, 229 from Group II in the 2004 survey, and 419 responses in the 2012 survey. Many users mentioned more than one website/application in their responses, so the total number of references to individual websites/apps added up to 1002 for the 2004 survey (combined) and 945 for the 2012 survey. Table 3 summarizes the services that received 5 or more responses from both surveys.

We can observe that a variety of different types of services were mentioned: Internet radio/streaming, music management and purchase, music identification, dictionary-type sources, reviews, etc. Many of our users seemed very savvy and knowledgeable, and specified multiple favorite websites and applications, explaining that they use each of them for very specific purposes.

When we compared the results from both surveys, we noticed a heavier concentration of responses with particular websites in 2012 survey. Only 5 websites were mentioned more than 5 times in both surveys and another 16 were new (in bold). There are a few completely new types of services, such as music identification (e.g., Shazam, Soundhound), and cloud music (e.g., Spotify, Grooveshark, Google Music). Peer-to-peer file sharing applications like Kazaa (in 2004) or general search engines like Google disappeared from the top list in 2012. The significant increase in the popularity of iTunes can probably be explained with the increasing use of mobile devices.

2004 Survey (combined)			2012 Survey		
Websites	#	%	Websites/Apps	#	%
Amazon	58	12.4	**Pandora**	149	35.6
All Music Guide	36	7.7	**YouTube**	68	16.2
Launch	25	5.4	**Spotify**	57	13.6
MTV	20	4.3	iTunes	56	13.4
Kazaa	19	4.1	**Shazam**	32	7.6
CD Now	18	3.9	Amazon	30	7.2
iTunes	17	3.6	**Naxos**	25	6.0
Mudcat Café	15	3.2	**Last.fm**	25	6.0
Rolling Stone	12	2.6	**Grooveshark**	25	6.0
Billboard	10	2.1	Pitchfork	20	4.8
Pitchfork	10	2.1	All Music Guide	20	4.8
Google	9	1.9	**NPR**	16	3.8
Lyrics.com	8	1.7	Grove Music Online	12	2.9
Grove Music Online	7	1.5	**Wikipedia**	11	2.6
eBay	6	1.3	**IMSLP**	11	2.6
Netscape Radio	6	1.3	**Soundhound**	10	2.4
Tower	6	1.3	**Rhapsody**	8	1.9
Andante	5	1.1	**Google Music**	8	1.9
CMT	5	1.1	**KEXP**	7	1.7
			Soundcloud	6	1.4
			ArkivMusic	5	1.2

Table 3. Services mentioned by 5 or more users.

We conducted a more thorough analysis of how these services were being used. Table 4[1] shows a list of how users specified they used different services, and how often those behaviors were mentioned in the surveys. We noticed a general trend of greater direct music consumption from these websites and applications, mostly due to the increased number of streaming and cloud music services. There was a significant drop among the several responses related to general music-information seeking, i.e., "To learn about the artists/bands." We conjecture that this has to do with the emergence and rising popularity of major music related websites and applications that serve purposes other than just providing music information. The existence and popularity of these websites now seem to heavily affect users' perception of what to expect from music services. Considering that 16 of the 21 top-rated services did not exist in 2004, this is not very surprising.

The data also suggest that the expectations from users regarding access to particular types of music information may have changed. For instance, websites providing lyrics information were sought by 7.3% of users in the 2004 survey, whereas in 2012, only 1.2% of respondents mentioned the need for lyrics information. Instead of visiting a particular website for lyrics, we suspect that many users are able to utilize a phrase search option in search engines like Google and are able to find links to numerous websites that provide lyrics. In addition, certain websites such as YouTube are not lyrics websites, but provide video content that often incorporates lyrics information. Thus, users might not even think of particular lyrics websites as one of their favorite music related websites. This may also be true for information on local events, which is much easier to find through social media in 2012.

We also saw a drop in responses indicating participation in or value for activities of social interaction [Table 5]. In 2004, online forums were extremely popular as a place to interact with other people. However, in 2012, social media such as Facebook, Twitter, and Google+ are now providing a space for users to discuss music, and users may not even think of these websites as specifically music-related, thus not showing up in the survey data.

The other category included: To track listening (new in 2012); save wish lists; find blogs; compare versions; etc.

Usage \ Response	2004 Survey		2012 Survey	
	#	%	#	%
To listen to music recordings	70	10.8	143	22.2
To discover new music/artists	14	2.2	80	12.4
To obtain/purchase music recordings	95	14.7	46	7.1
To obtain music information (general)	56	8.7	35	5.4
To identify/verify a particular song	6	0.9	34	5.3
To learn about the artists/bands	110	17.0	31	4.8
To read reviews	37	5.7	30	4.7
To search for/browse music recordings	22	3.4	23	3.6
To listen to samples before purchase	32	5.0	23	3.6
To get recommendations	11	1.7	22	3.4
To interact with other people	48	7.4	22	3.4
To obtain current news/information	26	4.0	21	3.3
To watch performances/music videos	14	2.2	18	2.8
To learn more about recordings	37	5.7	18	2.8
To obtain information for work/research	20	3.1	15	2.3
To obtain scores	29	4.5	13	2.0
To create playlists/stations	0	0.0	13	2.0
To store/manage music and metadata	0	0.0	9	1.4
To obtain lyrics	47	7.3	8	1.2
To find out about events	29	4.5	7	1.1
To share music recordings	2	0.3	6	0.9
To obtain ranking/rating information	11	1.7	4	0.6
Other	22	3.4	16	2.5

Table 4. How the websites/applications are used

4.2.2 Reasons for liking the Websites/Applications

From the user responses on why they like these websites/applications, we were able to infer what kinds of qualities users perceive to be important for these services. As shown in Table 5, there was a variety of different qualities mentioned by users in both surveys. We found it surprising that the quality mentioned most often was actually being exposed to new artists/music and serendipitous discovery, even more so than being free or inexpensive. The design aspects of the system (e.g., easy and convenient access to music; user-friendly system) were also perceived as important qualities. In fact, users' expectations on these aspects seem to be much higher compared to

[1] Table 4 and 5 are based on responses where the user specified the reason for liking the service. Some responses only specified the name or URL. 646 responses specified the reasons in 2004 and 644 in 2012.

how they were in 2004. Being able to customize or personalize the service was also highly appreciated. The responses for comprehensive coverage of music, including particular styles of music, and good music content that is updated frequently and matches users' interests/tastes, all dropped. We think it is unlikely that users do not believe these qualities are important anymore; rather, users probably just expect that current music services have these qualities to begin with. With the increasing use of mobile devices and a variety of applications, compatibility also surfaced as a new important quality for users.

The other category included: innovative, high quality recordings and writing, different purchase options, providing alerts, being able to listen to the whole album, not posting to Facebook, not hogging resources, directly paying artists, fewer bugs, etc.

Quality \ Response	2004 Survey		2012 Survey	
	#	%	#	%
Exposure to new things/Serendipity	18	2.8	80	12.4
Free/Inexpensive	50	7.7	68	10.6
Ease of access/Convenience	9	1.4	52	8.1
Customizability/Personalization	8	1.2	49	7.6
User-friendly/Ease of use	28	4.3	46	7.1
Comprehensive/Exhaustive coverage	64	9.9	37	5.7
Variety/Wide selection	51	7.9	36	5.6
Access to particular style of music	69	10.7	28	4.3
Compatibility/Use with other devices	1	0.2	25	3.9
Access to music samples	18	2.8	23	3.6
Good search/browse functions	8	1.2	23	3.6
Social/Ability to interact with others	52	8.0	22	3.4
Matches user's interest/taste	67	10.4	21	3.3
Good music/content	61	9.4	16	2.5
Quick/Instant service	7	1.1	16	2.5
Comparative data/Similar music	8	1.2	14	2.2
No rights management/restrictions	0	0.0	10	1.6
Fun/High entertainment value	2	0.3	9	1.4
Authority/Credibility of information	7	1.1	8	1.2
Does not require much user input	1	0.2	8	1.2
Rare/Obscure recordings/information	17	2.6	7	1.1
Familiarity/Set as default	8	1.2	6	0.9
Ability to store/archive recordings	0	0.0	6	0.9
New content/Updated frequently	48	7.4	5	0.8
Accuracy/Reliability of information	5	0.8	5	0.8
Access to local information	5	0.8	4	0.6
Good organization/design	11	1.7	3	0.5
No or fewer ads	6	0.9	3	0.5
Other	12	1.9	31	4.8

Table 5. The list of qualities valued by users

4.3 Discussion of the Trends in 2012

4.3.1 Popularity of Streaming Services

Analyzing the responses from both surveys clearly reveal the increasing popularity of Internet radio/music streaming services. With the rising use of various mobile devices such as tablets and smartphones, it is not surprising that streaming service is also becoming increasingly prevalent. Music is only one of many types of digital media users store on mobile devices, in addition to photos, videos, games, documents, etc., and numerous apps. This means that even though the storage space of these devices is always growing, the space allocated for music will always be limited. Listening to streaming music services rather than carrying one's own collection is one way to resolve that issue, as noted in comments below. Some comments also implied that there are songs users want to own vs. songs they just want to listen to now and then.

"I like them because I can still listen to music without cluttering up my phone or work computer with extra files."

"I also use things like spotify and pandora to listen to music that I don't necessarily want to own but have a hankering for now and again."

The quantitative data also support this trend. Table 6 shows various response statistics to questions related to Internet radio/streaming services and mobile music consumption. When we compare the frequency of users listening to these services from the 2004 and 2012 surveys (the first and second rows), we see a significant increase in the proportion of respondents (+30.7%) who use these services 5 or more times per month as well as a large decrease in the users (-16.5%) who never use these services. Two new questions were asked in the 2012 survey about how often people use music/music-themed apps on mobile phones (third row), and search for music heard through online streaming services (fourth row). 20.7% indicated they use music related apps "a few times a week" (8.4%) to "almost every day" (12.3%), implying a heavy mobile consumption of music by these users. Streaming music was also an important trigger for music searching; a total of 77% of respondents indicated that they search for music heard on streaming services at least once a month, and 22.8% do it "a few times a week" (12.0%) or "almost every day" (10.8%).

Source \ Response	Positive Frequency (times per month)			Never Total	Count Total
	1	2-4	5		
	%	%	%	%	#
Listening to streaming music/ online radio (old)	25.5	27.5	25.5	21.6	1066
Listening to streaming music/ online radio (new)	13.3	25.4	56.2	5.1	488
Using music or music-themed apps on mobile phone (new)	14.0	18.4	20.7	46.9	478
Searching for music heard from online streaming service (new)	24.6	29.6	22.8	23.2	501

Table 6. Various statistics related to streaming services and mobile music consumption

4.3.2 Emergence of Music Identification Services

Music identification services like Shazam and Soundhound also seem quite popular (42 responses combined). To provide a baseline for comparison, Table 7 shows the responses to different options for the question, "*How often to do you ask the following people/services for help* *when you search for music or music information?*" Friends and family members received the most positive responses. Over half (56.6%) of the responses indicated users consulted their social networks, and 43.5% of the 503 users said they have used music identification services. Overall the proportion of users who use this kind of service is still less than those who ask other people.

Responses from 2012 / Frequency	Friends and family	People on Social Network	Music ID service
Almost every day	2.4%	1.6%	1.0%
A few times a week	7.9%	6.3%	4.6%
About once a week	10.4%	6.5%	5.6%
2 or 3 times a month	20.0%	12.4%	5.6%
Once a month or less	37.7%	29.9%	21.1%
Never	21.6%	43.3%	56.5%
Total responses	509	508	503

Table 7. Frequency of users asking for help when searching for music or music information

However, when asked about how likely they would be to use the search/browse option of a music identification service, only 28.1% answered positively (Very likely + Somewhat likely), 62.4% answered negatively (Not very likely + Not at all likely), and 9.5% said "Don't know." We may infer that some people might use it out of curiosity but would not use it again. Considering that this type of service is still relatively new, we suspect that users' responses may change over time. We have observed similar results for a number of other search/browse options when we compared the responses from 2004 and 2012 surveys (e.g., "purchase patterns" (+20.6%), "recommendations from other people" (+14.9%), "mood/emotional state induced" (+7%) of positive responses in 2012).

4.3.3 Music combined with other Multimedia

It is interesting to note that YouTube was the second most preferred service by users in the 2012 survey, despite that the main objective of the website is not to provide music content. In addition to the benefit of being able to see music videos and concert/performance footage, the extensive coverage of YouTube was also highly valued by users as noted in the comments below:

"...gives an incredibly large choice of uploaded music to listen to (once again, including some specialized and rare items I wouldn't be able to find in my local library)."

"...I practically never searched for a song I didn't find on their servers."

YouTube was also seen as a place where many new artists post their work, where you can find "official" music videos, and hear a great deal of covers or different versions of songs. Some users also think of YouTube as an archive of old and rare music related materials. In addition, users appreciated that they do not need special hardware/software to use the service.:

"...the clips will play on any reasonable electronic platform (not restricted to certain brands or types, not requiring certain software beyond what it [sic] is likely to be on computers or smartphones already)."

4.3.4 Serendipitous Discovery of New Music

Exposure to and serendipitous discovery of new music/artists were very important to users in the 2012 survey [Table 5]. We conjecture that the popularity of services like Pandora or Spotify is greatly affecting user expectation. 32 responses specified that serendipitous discovery is the very reason why they like Pandora. Users gave split responses to the question asking how likely they would be to use the search/browse option "by recommendations from recommender systems": 50.8% positive, 46.9% negative, and 2.2% "Don't know." We saw a few responses that shed insight into why some people may not be pleased with current recommender systems:

"I've found a few new songs and artists I like through it, but I get frustrated with it sometimes when it thinks I like a whole genre because of one song, and it doesn't repeat the songs I REALLY like often enough."

"...even though it has stupid ads and plays music I don't like half the time... just because it's easy."

Recommender systems, of course, play a major role in the serendipitous discovery of music, but users mentioned employing other resources (e.g., YouTube, Pitchfork) to find new music, as well. By interviewing users about how they evaluate playlists, Lee [6] found that people definitely like learning new things, but still want them contextualized in familiar territory. We saw several comments that resonate with this finding:

"it exposes me to artists I'd never heard of before in genres I enjoy."

"they either play music I already like/know or introduce me to music that suits my tastes in an easy, unobtrusive way."

4.3.5 Customizability vs. Not Requiring User Input

As shown in Table 5, users' belief that a service is customized for them or that they are able to personalize it themselves seems very important and has a strong, positive effect on how they feel about that service. Some examples of comments include: "*stations tailored to my musical tastes and moods*," "*nice customization opportunities*," "*I like that you can say you like or dislike songs*," and "*I can adjust it to play music I like*." It is actually difficult to say how much these beliefs are objectively justified; for instance, do users understand the technical process of what happens after they like or dislike particular songs recommended by the service? We suspect that most users do not know how much these services actually incorporate their input to modify the results presented, except for the vague idea that they are somehow making it better to suit their tastes. This sense of control seemed to be what was important to them, rather than a set of perfect results [further discussion in Section 5].

It is also important to note that there exist a smaller number of users who prefer "not doing much." In order to appeal to these users, it will be important to provide an option to have an automatic algorithm learn their tastes and do the work on their behalf. Some of their comments include: "*I like that Pandora streaming radio lets me be lazy,*" "*making playlists is too much work most of the time*," "*they are free and do not require me to download or own anything. Streaming is key.*"

5. CONCLUSION AND FUTURE WORK

This work is part of a bigger research agenda that aims to provide an empirical basis for the development of music services reflecting the needs of real users. Our findings suggest several changes in the kinds of music services people like: an increase in the popularity of streaming services and mobile music consumption, an emergence of new types of services like music identification or cloud music services, an appreciation of music videos, serendipitous music discovery, and customizability, etc.

However, it also became apparent that many of the users' music information needs in 2012 did exist in 2004. The difference we see is that in 2012, a few dominant services are being used to fulfill those needs rather than a number of different websites. For instance, access to music videos/performances has always been important; users in the 2004 surveys were going to Yahoo! Launch, MTV, and VH1, and now seem to use YouTube for the same purpose. Users also told us that getting recommendations and discovering new music was important in 2004 [7]. In 2012, Pandora has become one of the major applications that serve those needs. The social aspect of music search was revealed in 2004 survey responses, where users said they were asking friends and family members about music and going to different forums to talk to other users [7]. Now we have Spotify and last.fm, where people can find out what their friends are listening to, and Shazam and Soundhound to help identify music.

Another interesting aspect of the survey was that many users recognize and accept the limitations of the services they like. As noted by the excerpts below, users seem willing to accept and forgive a few flaws if there are some other attractive aspects:

"...incredibly easy to use, awesome service, great wide-ranging library, integrated information, always being updated - glitchy at times and doesn't have everything but more than makes up for it in convenience and design."

"...it will do song identification, including humming/singing (still a little buggy, but a great idea),it pulls up lyrics for songs identified, gives you links to where you can purchase the music or to listen to it via the Slacker Radio app. It's really great!"

We believe that this is an important point with strong implications for developers of music systems and services. Much of the efforts in the MIR domain have been focused on improving the accuracy of particular algorithms, resulting in the "glass ceiling" problem where the effectiveness of techniques has reached its limits [3]. Maybe we should also start asking about what really matters to users; as the users in our survey told us, ease of use, a wide variety of music, innovative ideas, compatibility with other devices/apps, etc. are maybe as important as getting "accurate" results. We hope that the list of qualities valued by users of music related websites/applications will help inform system designers and developers in modifying existing services or creating new services.

A journal article reporting the detailed analysis of the 2012 survey and comparison of the results from 2012 and 2004 surveys is in preparation. For our future work, we plan to conduct additional user studies surrounding the expectations of specific music services, in particular, cloud music services. We are also interested in analyzing the failed cases, asking people what kinds of music related websites/applications they do not like and why.

6. REFERENCES

[1] L. Barrington, O. Reid, and G. Lanckriet: "Smarter than Genius? Human evaluation of music recommender systems," *Proceedings of the ISMIR*, pp. 357-362, 2009.

[2] J. Brinegar and R. Capra: "Managing music across multiple devices and computers," *Proceedings of the iConference*, 2011.

[3] J. S. Downie: "The music information retrieval evaluation exchange (2005-2007): A window into music information retrieval research," *Acoustical Science and Technology*, Vol. 29, No. 4, pp. 247-255, 2008.

[4] D. P. W. Ellis and B. Whitman: "The quest for ground truth in musical artist similarity," *Proceedings of the ISMIR*, pp. 170-177, 2002.

[5] K. Lai and K. Chan: "Do you know your music users' needs? A library user survey that helps enhance a user-centered music collection," *Journal of Academic Librarianship*, Vol.36, No.1, pp. 63-69, 2010.

[6] J. H. Lee: "How similar is too similar?: Exploring users' perceptions of similarity in playlist evaluation," *Proceedings of the ISMIR*, pp. 109-114, 2011

[7] J. H. Lee and J. S. Downie: "Survey of music information needs, uses, and seeking behaviours: Preliminary findings," *Proceedings of the ISMIR*, pp. 441-446, 2004

[8] M. Lesaffre, L. D. Voogdt, M. Leman, B. D. Baets, H. D. Meyer, and J. P. Martens: "How potential users of music search and retrieval systems describe the semantic quality of music," *Journal of American Society for Information Science and Technology*, Vol. 59, No. 5, pp. 695-707, 2008.

[9] M. Maguire, D. E. Motson, G. Wilson, and J. Wolfe: "Searching for Nirvana: Cataloging and the Digital Collection at the Experience Music Project," *Journal of Internet Cataloging*, Vol. 7, No. 1, pp. 9-31, 2004.

THE IMPACT OF MIREX ON SCHOLARLY RESEARCH (2005 – 2010)

Sally Jo Cunningham
University of Waikato
Hamilton, New Zealand
sallyjo@cs.waikato.ac.nz

David Bainbridge
University of Waikato
Hamilton, New Zealand
davidb@cs.waikato.ac.nz

J. Stephen Downie
University of Illinois
Urbana-Champaign, USA
jdownie@illinois.edu

ABSTRACT

This paper explores the impact of the MIREX (Music Information Retrieval Evaluation eXchange) evaluation initiative on scholarly research. Impact is assessed through a bibliometric evaluation of both the MIREX extended abstracts and the papers citing the MIREX results, the trial framework and methodology, or MIREX datasets. Impact is examined through number of publications and citation analysis. We further explore the primary publication venues for MIREX results, the geographic distribution of both MIREX contributors and researchers citing MIREX results, and the spread of MIREX-based research beyond the MIREX contributor teams. This analysis indicates that research in this area is highly collaborative, has achieved an international dissemination, and has grown to have a significant profile in the research literature.

1. INTRODUCTION

In this paper we report on the results of a study investigating the scholarly impact of the Music Information Retrieval Evaluation eXchange (MIREX), an annual formal evaluation of MIR systems and algorithms. A detailed examination of the structure of the MIREX trials and the results of the initial three years of the MIREX program is presented in [2]. In this present work, we look back on the MIREX publication literature to develop a rich picture of patterns of publication, collaboration, and dissemination of MIREX research (Section 3). Our analysis is based on a set of MIREX-related publications gathered via Google Scholar (Section 2). Issues encountered in building our MIREX document set indicate the existence of barriers to the dissemination of MIREX results. These issues are further explored in Section 4, where we also describe proposals to reduce these barriers—specifically, by providing a digital library of MIREX extended abstracts (thereby pulling the scattered abstracts together into a single repository that supports searching and browsing), and by recommending the development of referencing conventions for MIREX-related documents, datasets, and evaluation frameworks.

2. BIBLIOGRAPHIC DATA GATHERING

In this present paper, the impact of the MIREX trials is measured through both the number of MIREX-related papers published and the number of times that these papers have been cited. The MIREX publications include both the brief descriptions of the MIREX algorithms submitted to a given trial (referred to in the MIREX trials as ‘extended abstracts’) and the papers derived from the MIREX extended abstracts and MIREX results. As relying solely on sheer quantity of papers has obvious drawbacks, additional analysis focuses on the citation counts to round out the picture by indicating the degree to which each publication “has made a difference” [8] [9].

Three document sources have been commonly used in previous bibliometric studies: the ISI Web of Science (Thomson Reuters), Scopus (Elsevier), and Google Scholar (Google). The three have very different collection policies. The differences most significantly impacting this present study are that ISI restricts its computer science conference proceedings coverage more heavily than the other two; Scopus provides a more comprehensive coverage of both publishers and what they term ‘quality web sources’ than ISI; and Google Scholar includes the majority of the ISI and Scopus offerings as well as books, technical reports, and white papers.

In choosing Google Scholar as the source for this present study, we were influenced by issues of *coverage* and *user preference*. MIREX-flavored research is based strongly in computer science and engineering, two fields that place a greater emphasis on conference publications and technical reports than other sciences—and both ISI and Scopus do not include these publications types to the extent of Google Scholar [3] [4]. As the MIREX extended abstracts are not formally published, they are not included in the ISI and Scopus databases, and so their impact could not be measured through those resources. Further, we are specifically interested in exploring the documents most readily visible from the viewpoint of researchers interested in MIREX (rather than obtaining comprehensive coverage by hunting down MIREX related publications through all possible sources). For a given topic, Scopus, ISI, and Google Scholar are each likely to cover some content unavailable to the other two. Google and Google Scholar are the resources of preference for researchers in computer science and other science fields

[5]—and so by basing this study on documents drawn from Google Scholar, we build up a picture of the world of MIREX research that more closely resembles the viewpoint of MIREX researchers.

As citations must build up over time, we restricted the scope of this study to the years 2005 – 2010 rather than coming up to date (with the expectation that the 2009 and 2010 will show a 'Groos droop' [7]—the noticeable 'droop' in the right hand tail of the distribution—as citations are still accumulating for these later years). Each year was individually searched by using the date restriction facility in Google Scholar Advanced Search, and the search criterion used was "MIREX AND music" ('Mirex' is also a widely used insecticide, and so a further restriction to the music domain was necessary to filter out agricultural research).

Each paper in these initial results sets was then examined to gauge its relevance to this study. To be included in the study, the paper had to use / reference the results of a MIREX trial, a MIREX technique, MIREX data, or MIREX software. MIREX extended abstracts present in the Google Scholar results were also retained (extended abstracts not available through Google Scholar were not included in this study). Papers only tangentially related to MIREX were eliminated (for example, papers mentioning the MIREX trials as one example among many of retrieval evaluation exercises). Further documents were culled because they were not formal research papers (for example, undergraduate student assignments). Documents that were not publicly available were, when possible, downloaded for examination through the researchers' university library facilities (for example, the ACM publications). Some papers were not readily accessible, and for these the abstract and search snippet were examined; if these did not indicate a significant relationship to MIREX then they were also eliminated. Finally, duplicates were identified and merged (citation counts for copies were added together). Table 1 shows the document counts for both the raw and cleaned datasets.

	2005	2006	2007	2008	2009	2010
Raw	74	154	186	246	281	330
Cleaned	64	87	131	139	134	196

Table 1. Number of documents in the initial search results (raw) and final datasets (cleaned).

For each document retained, we recorded: author names, authors' institutional affiliations, title, abstract, publication type (journal article, book chapter, conference paper, thesis, technical report), abstract, source (eg, conference name), and citation count. As Google Scholar provides only the raw citation count, we were not able to filter for self-citations. Not all of this metadata was available for every document; specifically, a small number of institutional affiliations were absent and so the analyses of author geographic distribution and collaboration (Section 3.6) may be slight underestimates.

3. ANALYSIS OF MIREX PUBLICATONS

This section examines the impact of MIREX through publication and citation counts, the extent of collaboration within the MIREX research community, and the geographic distribution of MIREX research efforts.

3.1 MIREX Publication Set

For 2005 - 2010 we identified a total of 752 publications: 236 MIREX extended abstracts, and 516 more formal publications based on the MIREX trials and results (Table 2). Theses and dissertations are treated separately in Section 3.2. Note that this dataset does not provide exhaustive coverage of either category, and coverage of the MIREX abstracts in particular is patchy when viewed through the lens of Google Scholar. We return to this point in Section 4 with an explanation of this phenomenon and a partial solution to the relative invisibility of some MIREX documents.

Table 2 shows an overall increase in the number of MIREX-derived publications—a ten-fold increase in the first three years of the trials, and another large increase in 2010. The MIREX trials are clearly seen by the research community to have value, as expressed through the growth of literature that builds on MIREX.

However, MIREX extended abstracts can be seen to receive relatively fewer citations than the publications deriving from the MIREX trials (and even at that, the citation average for MIREX extended abstracts is in most years heavily skewed by one or two abstracts that received large numbers of citations). In contrast, a comparable analysis of the TRECVid (video retrieval) [8] [9] and ImageClef (image retrieval) [10] evaluations show the papers for those evaluation trials to have a similar citation profile to their respective derived literature. Again, in Section 4 we explore possible reasons for the lower citation counts and offer a tactic to counter this effect.

The h-index is a measure that attempts to encapsulate both the quantity and visibility of a set of publications [1]. It is calculated as the number h that is the largest number of papers in the set that have each received at least h citations. In Table 2 we see further evidence that the MIREX-derived publications have a far higher profile than the MIREX extended abstracts; in a given year the h-index for the derived publications is roughly three to four times higher than that of the extended abstracts.

3.2 MIREX-derived Publications: Publication Types

The derived papers are published formally as chapters in edited books, as conference papers, and in journals, and are less formally made available as technical reports. The publication venues follow the profile typical of computer science and engineering: there is a greater emphasis on conference than on journal publications, with a smaller

Year	MIREX extended abstracts				MIREX derived publications			
	No.	Citations	Mean citations	h-index	No.	Citations	Mean citations	h-index
2005	54	302	5.59	10	10	358	35.80	–
2006	36	226	6.28	6	51	1308	25.65	20
2007	33	242	7.33	9	98	1453	14.83	21
2008	38	99	2.61	6	101	1754	17.37	22
2009	33	34	1.03	3	101	802	7.94	14
2010	42	35	0.83	3	155	914	5.90	14

Table 2. Overview of citation data, 2005 – 2010**.**

(but not completely negligible) number of book chapters and technical reports (Table 3).

The MIREX annual results are reported through a special session in the ISMIR conference, and ISMIR is the focal conference for music retrieval research—so it is to be expected that ISMIR would be a significant publication venue for the MIREX-derived research. As Table 4 illustrates, once past the inaugural year over three quarters of the MIREX-derived papers are published outside of ISMIR, and that spread to other conferences and journals increased in the final year of this present study.

	2005	**2006**	**2007**	**2008**	**2009**	**2010**
Technical report	0	1	3	3	2	1
Book chapter	0	2	1	2	2	7
Conference	10	37	67	79	83	106
Journal article	0	11	17	17	14	41

Table 3. Publication type for MIREX-derived papers.

2005	2006	2007	2008	2009	2010
5 (50%)	14 (27%)	29 (30%)	26 (26%)	25 (25%)	28 (18%)

Table 4. Number and percentage of MIREX-derived papers that are published in ISMIR conferences.

3.3 MIREX Theses and Dissertations

Table 5 shows the number of research theses and dissertations that are based to some extent on the MIREX trials—typically by referencing MIREX annual results, by testing a novel algorithm against published MIREX datasets, or reporting more fully on the researcher's own MIREX entry. The uptake of MIREX as a degree focus bodes well for the future of research in this area, as Masters and PhD students move into research positions.

The theses and dissertations are cited less than the other MIREX-derived publications (Table 5), but that is to be expected—in the science fields, theses/dissertations are commonly re-worked into journal or conference publications, which are both more visible to other researchers and more visibly peer-reviewed (and hence more likely to be noticed and cited).

Year	**Degrees**	**No.**	**Total Citations**	**Mean citations**
2005	Masters: 1 PhD: 1	2	7	3.5
2006	Masters: 8 PhD: 5	13	90	6.92
2007	Masters: 10 PhD: 3	13	114	8.77
2008	Masters: 14 PhD: 9	23	90	6.92
2009	Masters: 4 PhD: 10	14	19	1.36
2010	Ugrad: 1 Masters: 9 PhD: 11	21	46	2.19

Table 5. MIREX-related theses and dissertations.

3.4 Collaboration in MIREX Research

The mean number of authors per paper is presented in Table 6 and the distribution of author numbers per paper is presented in Figure 2. The research teams submitting to the original MIREX trials were small—the vast majority comprised one or two researchers—but over the years the number of participants in a MIREX submission has grown. The number of co-authors for papers based on MIREX has shown steady growth to 2010. Both trends likely reflect the maturing of this area of research, as stable research groups develop from the interests of one or two key researchers.

The size of the collaborative teams for both categories of paper are larger than might be expected; typically the mean number of co-authors for a computer science or engineering paper hovers around two [6].

	2005	**2006**	**2007**	**2008**	**2009**	**2010**
Extended abstracts	1.75	1.75	2.39	2.47	2.85	2.79
Derived papers	2.3	2.31	2.62	2.96	2.95	3

Table 6. Mean number of authors per paper.

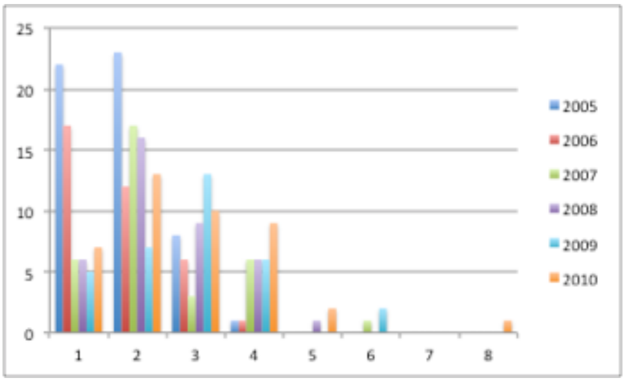


Figure 2a. Number of authors per paper for MIREX extended abstracts.

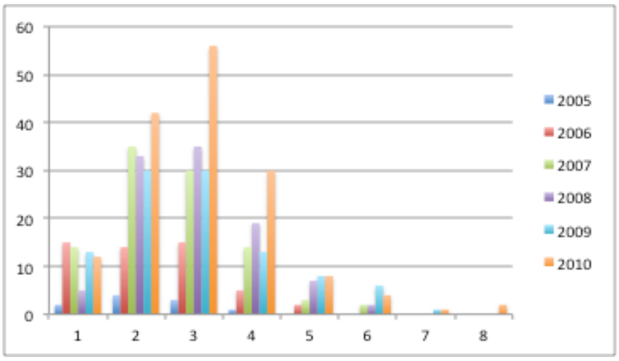


Figure 2b. Number of authors per paper for MIREX derived publications (excluding theses and dissertations).

3.5 Geographic Distribution of MIREX Researchers

Thirty-six countries have contributed at least one publication in the 2005 – 2010 MIREX document set (Table 7 presents the league table of the top contributors, and Figure 3 presents a map-based visualization of this geographic distribution). Participation in the MIREX evaluations is clearly not restricted to a small inner circle, and the MIREX results are seeing similarly widespread application.

Examining more closely the national affiliations for authors of the papers under study, we see that the research is surprisingly collaborative across national boundaries and between institutions within a single country (Table 8). The percentage of papers involving co-authors from two or more countries seems to have stabilized at 12% from 2007 – 2009, and then to have increased sharply in 2010 to 18%. The increases in these cross-boundary collaborations may reflect the increasing maturity of the field, as researchers move to new positions while maintaining research ties in their former institutions, or perhaps the personal connections made through ISMIR / MIREX conferences are encouraging greater collaboration outside the researcher's home institution. A further drill-down into the publications dataset (and likely follow-up survey of MIREX researchers) is necessary to clarify the factors contributing to this effect.

Country	MIREX abstracts	Derived papers	Theses	Total
USA	33	130	19	182
France	29	61	5	95
Spain	27	48	11	86
UK	22	50	11	83
Canada	14	32	7	53
Austria	16	23	6	45
Finland	16	16	2	34
Germany	15	19		34
China	14	19	1	34
Japan	8	22		30

Table 7. Number of publications by country for the top ten contributors, 2005-2010.

	Avg no. of countries per paper	% of multi-national collaborations	Avg no. of institutions per paper
2005	1.2	20.0%	1.13
2006	1.04	3.9%	1.08
2007	1.16	13.3%	1.22
2008	1.15	12.9%	1.32
2009	1.14	12.0%	1.43
2010	1.18	18.3%	1.46

Table 7. Summary of international and cross-institutional collaborations.

4. BUILDING A GREATER PROFILE FOR MIREX EXTENDED ABSTRACTS

Early in the data gathering process it became apparent that a substantial proportion of the MIREX extended abstracts were not being harvested by our Google Scholar searches—for example, a manual count of the 2008 extended abstracts on the MIREX wiki (http://www.music-ir.org/mirex/wiki/) yielded 51 submission abstracts, where our Google Scholar search identified only 38. Further, several extended abstracts appeared as multiple, but not identical, versions of the same intellectual content (obviously revised versions of a single submission). We later discovered that yet other extended abstracts were indeed present in the Google Scholar collection, but as they did not include MIREX in the document text or extracted metadata, they were not returned in our searches.

Perhaps more troublingly, Google Scholar was unable to extract meaningful bibliographic metadata for a number of the extended abstracts that did appear in the MIREX searches. For these latter extended abstracts, the researchers verified that they were indeed part of the MIREX trials by traversing backwards through the file hierarchy in which the document was stored, until we could determine that it was indeed a legitimate contribution to a MIREX evaluation cycle. For an extended abstract lacking metadata, a researcher unfamiliar with MIREX, but interested in the intellectual content of the paper, would not know the extent to which the results presented in the paper could be trusted—was this paper

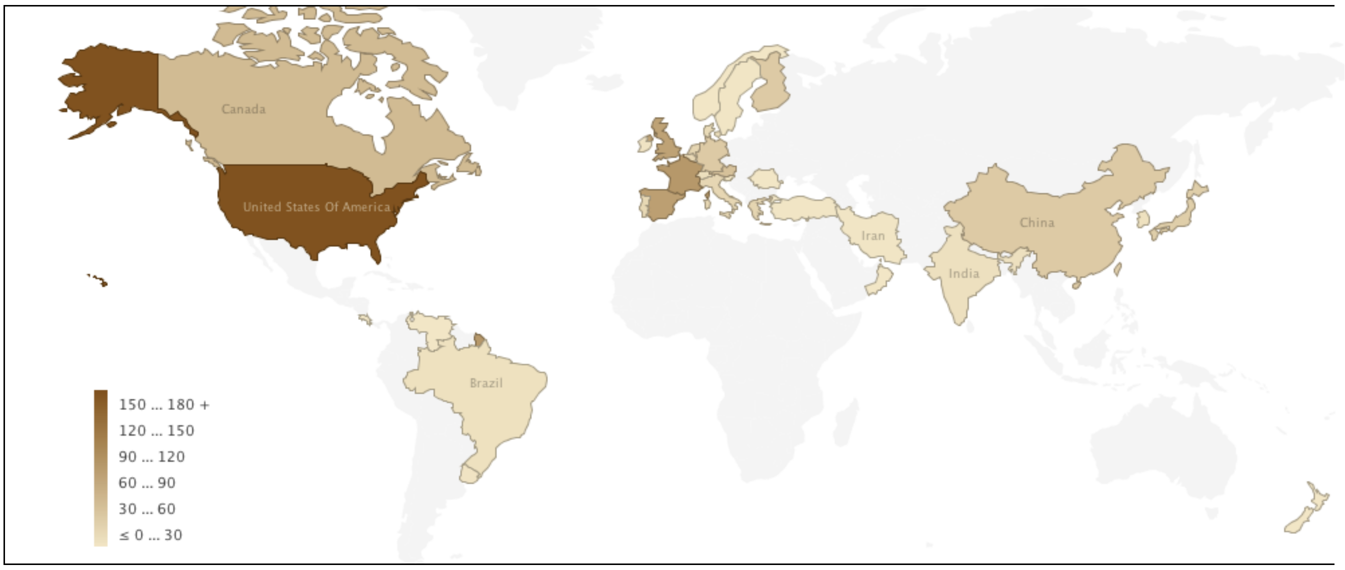
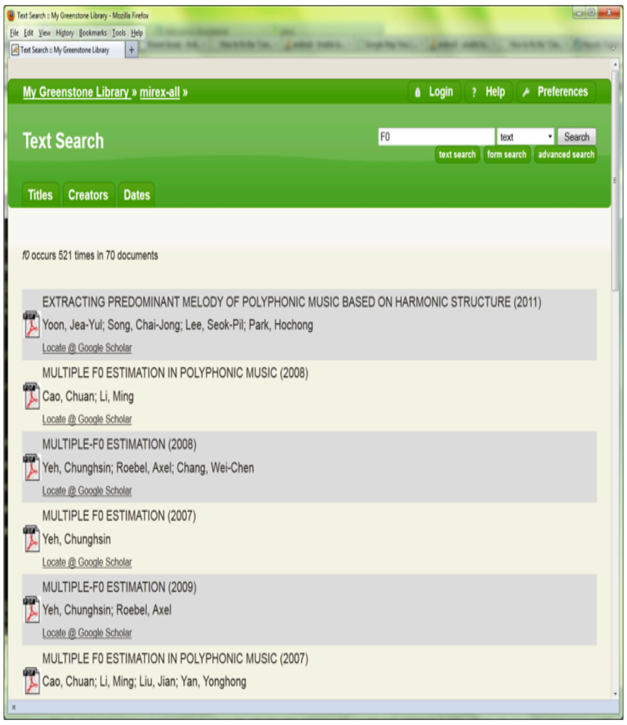


Figure 3. Geographic distribution of MIREX researchers.

peer reviewed? Was it a technical report, less formally 'published' but still endorsed by the authors' institutions? Or was it a student assignment accidentally harvested by Google Scholar?

These issues with identifying both the existence and provenance of MIREX extended abstracts in Google Scholar are likely explanations for the relatively low citation counts for the extended abstracts identified in this present study (Table 2). To mitigate these issues and, we hope, provide a mechanism for the MIREX evaluation documents to gain a higher profile, we have developed a digital library of the extended abstracts using the open source digital library software Greenstone [11]. Figure 4 shows a snapshot taken from this resource. The figure shows the result of searching for "F0" using the full-text index of the abstract texts. Each matching document displays the title, year of publication, and the authors, along with a link to the PDF document. Also provided for each document is a "Locate @ Google Scholar" link. Clicking on this takes the title of the paper and initiates a search for this on Google Scholar. While not guaranteed to find a match, we found it worked reliably well in practice, and a convenient way to locate citation information about the extended abstract. Features also include browsing by title, author and date, as well as search by these metadata fields. The resource can be accessed through http://music-ir.org/mirex-dl/library).

While this digital library provides improved access facilities to the extended abstracts, it is worth noting that the some of the metadata for each abstract may be provided through the digital library interface and is not apparent on the document itself. Searchers may stumble across an extended abstract via any number of mechanisms—a Google Scholar search, a general search engine query, a link from another website—and there is no guarantee that the specific path a particular user takes in locating a given document will provide any cues as to the document's provenance *beyond those included in the text of the document itself*. For this reason, we recommend that each extended abstract should include a header providing the citation for that abstract.


Text Search :: My Greenstone Library - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Text Search :: My Greenstone Library
My Greenstone Library » mirex-all »
Login
? Help
Preferences
Text Search
F0
text
Search
text search
form search
advanced search
Titles
Creators
Dates

f0 occurs 521 times in 70 documents

EXTRACTING PREDOMINANT MELODY OF POLYPHONIC MUSIC BASED ON HARMONIC STRUCTURE (2011)
Yoon, Jea-Yul; Song, Chai-Jong; Lee, Seok-Pil; Park, Hochong
Locate @ Google Scholar

MULTIPLE F0 ESTIMATION IN POLYPHONIC MUSIC (2008)
Cao, Chuan; Li, Ming
Locate @ Google Scholar

MULTIPLE-F0 ESTIMATION (2008)
Yeh, Chunghsin; Roebel, Axel; Chang, Wei-Chen
Locate @ Google Scholar

MULTIPLE F0 ESTIMATION (2007)
Yeh, Chunghsin
Locate @ Google Scholar

MULTIPLE-F0 ESTIMATION (2009)
Yeh, Chunghsin; Roebel, Axel
Locate @ Google Scholar

MULTIPLE F0 ESTIMATION IN POLYPHONIC MUSIC (2007)
Cao, Chuan; Li, Ming; Liu, Jian; Yan, Yonghong

Figure 4. Sample search results display in our prototype digital library of MIREX extended abstracts.

Close examination of the MIREX-derived literature also uncovered difficulties that some authors had obviously experienced in knowing how to cite the results of the MIREX trials (for example, the relative performance of specific algorithms). While an overview of the year's MIREX evaluations generally appears in the proceedings of the annual ISMIR conference, this document does not provide comprehensive results from all tasks. Exhaustive summaries of results are available on the MIREX wiki[1], but these are not provided in a form that is recognized as being suitable for indexing by Google Scholar—and no guidelines are given on the wiki as to how to cite these results. A straightforward solution would be to issue these results summaries as technical reports and store them in repositories indexed by Google Scholar and other scholarly indexing systems.

Similar difficulties were apparently experienced in providing formal acknowledgment of the MIREX trials, experimental setup, or datasets (the MIREX wiki does not provide a canonical reference form for these). While these papers did use the term "MIREX" in describing the results and datasets in the paper body (and so our Google Searches did return these papers), these mentions were not tied to entries in the papers' reference sections—and consequently no MIREX entity receives citation credit. Contrast this situation with that of the TRECVid evaluation series, which suggests standard references for many aspects of this programme (http://www-nlpir.nist.gov/projects/t01v/trecvid.citation.html). We encourage the MIREX organizers to develop similar referencing guidelines, and will include them in the home page of our extended abstracts digital library.

5. CONCLUSIONS

Our examination of the MIREX literature (the extended abstracts and papers referring to / referencing MIREX results, datasets, and evaluation trials) portrays a thriving international research community, characterized by collaboration. We have identified barriers to the accessibility of the MIREX extended abstracts, and present a prototype digital library for these documents that we believe can improve the MIREX profile in the larger research community. We also provide recommendations for modifications to the format of extended abstracts and the information presented in the MIREX wiki, to increase the visibility of MIREX to search engines and to make it easier for researchers to locate citation information for MIREX documents.

We believe that these small changes have the potential for a large payoff: MIREX can follow in the steps of the successful TRECVid and ImageCLEF series by providing the MIREX extended abstracts and citation information in formats that are readily indexed by Google Scholar and other resources, easily located by interested researchers, and easily cited in relevant publications.

6. REFERENCES

[1] L. Bornmann and H-D Daniel: "What do we know about the h index?", Journal of the American Society for Information Science and Technology, Vol. 58, No. 9, pp. 1381-1395, 2007.

[2] J. S. Downie: "The music information retrieval evaluation exchange (2005 – 2007): A window into music information retrieval research", Acoustical Science and Technology, Vol. 29, No. 4, pp. 247-255, 2008.

[3] J. Freyne, L. Coyle, B. Smyth, and P. Cunningham: "Relative status of journal and conference publications in computer science", Communications of the ACM, Vol. 53, No. 11, pp. 124-132, 2010.

[4] A.W. Harzing: "Citation analysis across the disciplines: the impact of different data sources and citation metrics", http://www.harzing.com/data_metrics_comparison.htm

[5] B.M. Hemminger, D. Lu, K.T.L. Vaughan, and S.J. Adams: "Information seeking behavior of academic scientists", Journal of the American Society for Information Science and Technology, Vol. 58, No. 14, pp. 2205-2225, 2007.

[6] M.E.J. Newman: "The structure of scientific collaboration networks", Proceedings of the National Academy of Sciences, Vol. 98, No. 2, pp. 404-409, 2001.

[7] R. Rousseau: "A bibliometric study of Nieuwenhuysen's bibliography of microcomputer software for online information and documentation work", Journal of Information Science, Vol. 16, pp. 45 – 50, 1990.

[8] C.V. Thornley, A.C. Johnson, A.F. Smeaton, and H. Lee: "The scholarly impact of TRECVid (2003 – 2009)", Journal of the Ame4rican Society for Information Science and Technology, Vol. 62, No. 4, pp. 613-627, 2011.

[9] C.V. Thornley, S.J. McLoughlin, A.C. Johnson, and A.F. Smeaton: "A bibliometric study of video retrieval evaluation benchmarking (TRECVid): A methodological analysis," Journal of Information Science, Vol. 37, No. 6, pp. 577-593, 2011.

[10] T, Tsikrika, A.G. Seco de Herrera, and H. Muller: "Assessing the scholarly impact of ImageCLEF", Multilingual and Multimodal Information Access Evaluation, Lecture Notes in Computer Science, Vol. 6941, pp. 95-106, 2011.

[11] I.H. Witten, D. Bainbridge, D.M. Nichols: How to build a digital library (2nd edition), Morgan Kaufmann, 2010.

[1] eg, "MIREX 2008 Overall Results Poster , http://www.music-ir.org/mirex/results/2008/MIREX2008_overview_A0.pdf

AN EMOTION MODEL FOR MUSIC USING BRAIN WAVES

Rafael Cabredo[1,2], Roberto Legaspi[1], Paul Salvador Inventado[1,2], and Masayuki Numao[1]

[1]Institute of Scientific and Industrial Research, Osaka University, Japan,
[2]Center for Empathic Human-Computer Interactions, De La Salle University, Philippines,
{cabredo,roberto,inventado,numao}@ai.sanken.osaka-u.ac.jp

ABSTRACT

Every person reacts differently to music. The task then is to identify a specific set of music features that have a significant effect on emotion for an individual. Previous research have used self-reported emotions or tags to annotate short segments of music using discrete labels. Our approach uses an electroencephalograph to record the subject's reaction to music. Emotion spectrum analysis method is used to analyse the electric potentials and provide continuous-valued annotations of four emotional states for different segments of the music. Music features are obtained by processing music information from the MIDI files which are separated into several segments using a windowing technique. The music features extracted are used in two separate supervised classification algorithms to build the emotion models. Classifiers have a minimum error rate of 5% predicting the emotion labels.

1. INTRODUCTION

Listening to music brings out different kinds of emotions. It can be involuntary and different for every person and primarily caused by musical content. A lot of research has been done identifying music features that are associated with affecting emotion or mood [3, 5, 17]. The work of [9] also investigates music features and discusses how changing these features can affect the emotions the music elicits.

With a good background of how different music features affect emotions, it is possible to automatically classify and predict what kind of emotions a person will experience. A survey of music emotion research by Kim et al. [6] report that the typical approach for classifying music using emotion is to build a database of ground truth of emotion labels by subjective tests. Afterwards, a machine learning technique is used to train a classifier to automatically recognize high-level or low-level music features.

A common problem encountered by previous work is the limitation of the annotation for emotion. It takes a lot of time and resources to annotate music. Lin, et al. [8] reviews various work on music emotion classification and utilize the vast amount of online social tags to improve emotion classification. However, a personalized emotion model for labelling music would still be desirable. Music that is relaxing for some people may be stressful for others.

Songs are also usually annotated with the most prominent emotion (i.e. only one emotion label per song). Multi-label classification [18] can be used to have richer emotion annotations. These annotations however are still discrete-valued emotion labels.

In our work, we are interested in learning how emotion changes throughout the song and identify music features that could have caused these changes. Because of this, continuous-valued emotion annotations are preferred. One method to do this is to use an electroencephalograph (EEG) in recognizing emotions similar to the work used to develop Constructive Adaptive User Interface (CAUI), which can arrange [7, 13] and compose [14] music based on one's impressions of music. In addition to collecting continuous-valued annotations for full-length music, we focus our work on considering individual emotion reactions to music as opposed to building a generalized emotion model.

2. DATA COLLECTION METHODOLOGY

We construct a user specific model by using supervised machine learning techniques to classify songs using music features. As mentioned earlier, this task requires songs that can elicit emotions from a listener and the music features of these songs.

For this research, we had a 29-year old female participant who selected and annotated songs. The music collection is a set of MIDI files comprised of 121 Japanese and Western songs having 33 Folk, 20 Jazz, 44 Pop, and 24 Rock music. By using MIDI files, the music information can be easily extracted to produce high-level features for the classifier. MIDI files also eliminate any additional emotions contributed by lyrics.

2.1 Emotion annotation

Music emotion annotation is performed in 3 stages. First, the subject listened to all songs and manually annotated each one. The subject was instructed to listen to the entire song and was given full control on which parts of the song she wanted to listen to.

After listening to each song, the subject gives a general impression on how joyful, sad, relaxing, and stressful each

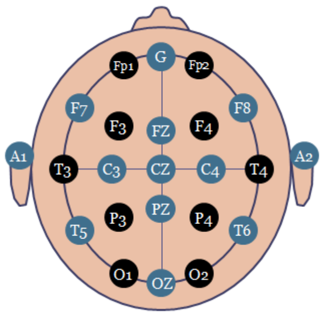


Figure 1. The EEG has 23 electrodes used to record electrical changes on the scalp. Each node is identified by a letter to indicate lobe position: F-Frontal lobe, T-Temporal lobe, C-Central lobe, P-Parietal lobe, O-Occipital lobe. 'Z' refers to an electrode placed on the mid-line

song was using a five-point Likert scale. Aside from the emotions felt, the subject was also asked to rate whether she was familiar with the song or not using the same scale. With this feedback, we chose the 10 most relaxing songs and 10 most stressful songs with varying levels of familiarity to the subject. The manual annotation was done in one session for approximately one and a half hours.

Since collection of the emotion annotations takes a lot of time and effort from the subject, it was decided to concentrate time and resources on a certain type of emotion. We opted to concentrate on relaxing music because these are normally the kind of music people would want to listen to on stressful days. The stressful songs are meant to serve as negative examples for the classifier.

In the second stage an EEG was used to measure brain activity while the subject listened to the 20 songs previously selected. The EEG device is a helmet with electrodes that can be placed on all scalp positions according to the International 10–20 Standard. Figure 1 shows the location of the different electrodes. Using the EEG, electric potential differences were recorded with a reference electrode on the right earlobe.

Work on EEG to recognize emotions find that different mental state produces a distinct pattern of electrical activity [1, 2]. The right hemisphere is responsible for negative emotions (i.e. stress, disgust, sadness) while the left hemisphere is responsible for positive emotions (i.e. happiness, gratitude, amusement).

The EEG device is very sensitive. As such, the subject was instructed to close her eyes and remain still while data was being collected. Listening sessions had to be limited to a maximum of 30 minutes or upto the moment that the subject begins to feel uncomfortable wearing the helmet. We had to ensure that the subject was comfortable and eliminate external factors that may contribute to changes in emotion. On average, EEG readings for 7 songs were recorded per session.

Prior to playing each music, we introduce a 10 second white noise to help the subject focus on the task at hand without stimulating a strong emotional response. After listening to one song, a short interview is conducted to determine if the subject particularly liked or disliked specific parts of the song. The interview also helped confirm the initial manual annotations of the subject.

In the final stage, continuous emotion annotations were obtained using EMonSys. This software [1] uses the emotion spectrum analysis method (ESAM) [12] to convert brain wave readings to emotion readings. Using data from 10 scalp positions at Fp1, Fp2, F3, F4, T3, T4, P3, P4, O1, O2, electric potentials were separated into their θ (5–8 Hz), α (8–13 Hz) and β (13–20 Hz) frequency components by means of fast Fourier transforms (FFT). Cross-correlation coefficients for each pair of channels are computed (i.e., 10 channels * 9 channels/2) and these are evaluated for every time step together with the 3 bands to obtain an input vector Y having 135 variables at each time step. EMonSys can evaluate the EEG readings at different time steps. We used the smallest available: 0.64 seconds.

Using an emotion matrix C, this 135-dimensional vector is linearly transformed into a 4-D emotion vector $E = (e_1, e_2, e_3, e_4)$, where e_i corresponds to the 4 emotional states, namely: stress, joy, sadness, and relaxation. Formally, the emotion vector is obtained by

$$C \cdot Y + d = E, \tag{1}$$

where d is a constant vector. The emotion vector is used to provide a continuous annotation to the music every 0.64 seconds. For example, if one feels joy, the emotion vector would have a value of $E = (0, e_2, 0, 0)$.

2.2 Extracting Music Features

A song having length m is split into several segments using a sliding window technique. Each segment, or now referred to as a window w has a length n, where one unit of length corresponds to one sample of emotion annotation.

MIDI information for each window is read using a module adapted from jSymbolic [10] to extract 109 high-level music features. These features can be loosely grouped into the following categories: Instrumentation, Texture, Dynamics, Rhythm, Pitch Statistics, and Melody. The feature set includes one-dimensional and multi-dimensional features. For example, *Amount of Arpeggiation* is a one-dimensional Melody feature, *Beat Histogram* is a 161-dimensional Rhythm feature, etc. All features available in jSymbolic were used to build a 1023-dimension feature vector. The category distribution of the feature vector is shown in Table 1. The *Others* category refers to the features *Duration* and *Music Position*. *Duration* is a feature from jSymbolic, which describes the length of the song in seconds. *Music Position* refers to the position of the window relative to duration of the song. Although it was known that not all of the features will be used, this approach allows utilization of feature selection techniques to determine which features were the most important in classification.

After extracting the features for one window, the window goes through the data using a step size s until the end

[1] software developed by Brain Functions Laboratory, Inc.

Category	Amount	Percentage
Dynamics	4	0.39%
Instrumentation	493	48.19%
Melody	145	14.17%
Pitch	174	17.01%
Rhythm	191	18.67%
Texture	14	1.37%
Others	2	0.20%

Table 1. Distribution of features used for the instances

of the song is reached. Each window was labelled using the average emotion values within the length of the window. Formally, the label for w_i is the emotion vector

$$E^i = \frac{1}{n}\sum_{j=i}^{i+n} E^j = \frac{1}{n}\sum_{j=i}^{i+n}\left(e_1^j, e_2^j, e_3^j, e_4^j\right), \qquad (2)$$

where $1 \leq j \leq m - n$.

3. EMOTION MODEL

Weka's [4] implementation of linear regression and C4.5 were used to build the emotion models for each emotion. The training examples were derived from the window given one emotion label, which results to four datasets. Each dataset has a maximum of 6156 instances using the smallest values for the sliding window (i.e. $n = 1$ and $s = 1$). The number of instances depends on the parameters used for windowing. During preliminary experiments we observed that the decrease of training data due to larger step sizes had too much of a negative influence on performance. As such, all features were extracted using the smallest size of $s = 1$ for all experiments.

Prior to training, all features that do not change at all or vary too frequently (i.e. varies 99% of the time) are removed. Afterwards, normalization is performed to have all feature values within $[0, 1]$.

3.1 Using Linear Regression

The linear regression used for building the emotion models uses the Akaike criterion for model selection and M5 method [15] to select features. The M5 method steps through the features and removes features with the smallest standardized coefficient until no improvement is observed in the estimate of the error given by the Akaike information criterion.

3.2 Using C4.5

C4.5 [16] is a learning technique that builds a decision tree from the set of training data using the concept of information entropy. Since this technique requires nominal class values, the emotion labels are first discretized into five bins. Initial work used larger bin sizes but we observed poorer performance using these.

3.3 Testing and Evaluation

We used 10-fold cross-validation to assess the models generated by the two methods using different values for the window length. We use the relative absolute error for evaluating performance of the classifiers. Weka computes this error measure by normalizing with respect to the performance obtained by predicting the classes' prior probabilities as estimated from the training data with a simple Laplace estimator. Figures 2 and 3 show the change in relative absolute error using linear regression and C4.5, respectively. Window length values were varied from 1 to 30 samples (i.e. 0.64 seconds to 19.2 seconds of music).

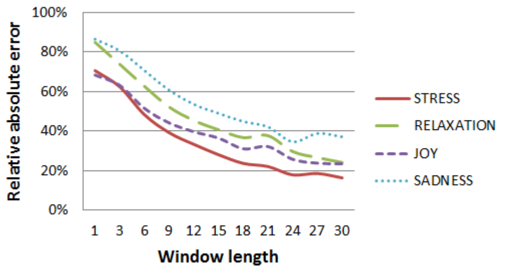


Figure 2. Relative absolute error using linear regression

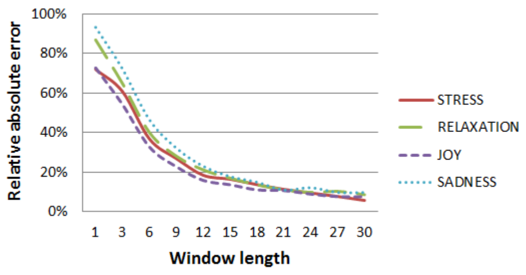


Figure 3. Relative absolute error using C4.5

4. RESULTS AND ANALYSIS

Increasing the window size increases accuracy of the classifiers. Further experiments were done to include window sizes upto 240 samples. Results of these are shown in Figures 4 and 5. From these results, we find the value of n which minimizes the average relative absolute error over $n = [1..20]$. For linear regression, using $n = 90$ gives the minimum average relative absolute error of 7.6% with a correlation coefficient of 0.8532 and root mean squared error of 0.1233. The average is taken from values for the four emotion model results.

Using C4.5, a smaller window length is necessary to obtain similar results. Using $n = 60$, the average relative absolute error is 5.1%, average root mean squared error is 0.0871, and average Kappa statistic is 0.9530. The Kappa statistic describes the chance-corrected measure of agreement between the classifications and the true classes.

When $n \geq 120$, we notice that some songs are no longer included in the training data as the window length becomes greater than the song length. As such, results using these window lengths may not be accurate.

	$n = 1$		$n = 30$		$n = 60$		$n = 90$		$n = 120$	
Class No.	S	R	S	R	S	R	S	R	S	R
1	84.0%	95.3%	56.5%	82.2%	52.3%	80.5%	51.0%	81.5%	49.1%	80.5%
2	13.3%	3.8%	31.6%	9.9%	28.6%	6.0%	26.1%	3.7%	25.7%	3.2%
3	1.9%	0.7%	8.7%	6.5%	15.4%	10.3%	18.4%	11.1%	20.7%	9.4%
4	0.5%	0.2%	1.8%	1.0%	1.8%	2.2%	2.3%	2.7%	1.6%	5.7%
5	0.3%	0.0%	1.4%	0.4%	1.9%	1.0%	2.1%	1.1%	2.9%	1.1%

Table 2. Class sizes for Stress (S) and Relaxation (R) data after discretization

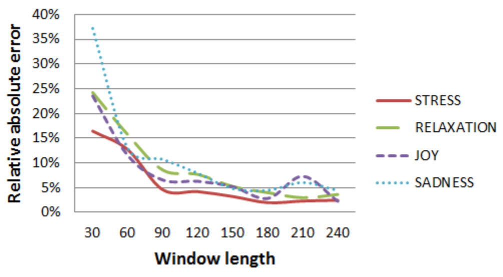


Figure 4. Relative absolute error using linear regression

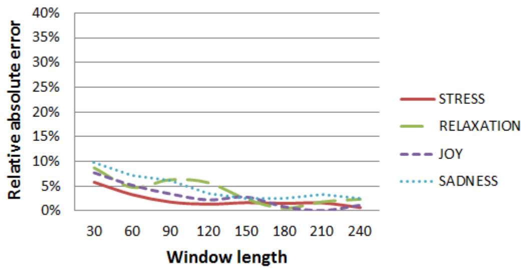


Figure 5. Relative absolute error using C4.5

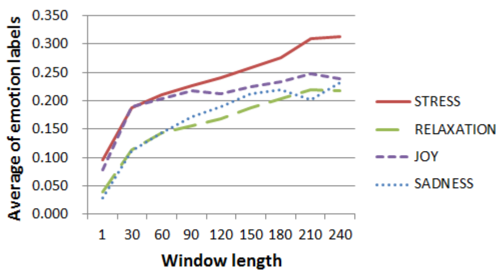


Figure 6. Average of emotion value for different window lengths

Category	Stress	Relaxation	Sadness	Joy
Rhythm	40.4%	32.4%	32.8%	34.0%
Pitch	21.3%	29.7%	28.4%	32.0%
Melody	10.6%	16.2%	19.4%	20.0%
Instrumentation	17.0%	10.8%	10.4%	8.0%
Texture	8.5%	5.4%	4.5%	2.0%
Dynamics	0.0%	2.7%	1.5%	0.0%
Others	2.1%	2.7%	3.0%	4.0%

Table 3. Distribution of features used in C4.5

4.1 Influence of window length

Model accuracy is highly dependent on the parameters of the windowing technique. Increasing the window length allows more music information to be included in the instances making each more distinguishable from instances of other classes.

Increasing the window length also affects the emotion annotations. ESAM was configured to produce emotion vectors having positive values. Since most of the emotion values are near zero, the average emotion values for the windows are also low. Figure 6 shows the steady increase of the values for the class labels as the window length is increased. The standard deviation also follows a linear trend and steadily increases from 0.091 to 0.272 for the same window lengths. Using larger window lengths diversifies the emotion labels as well which, in turn, contributes to better accuracy.

The low average values also affected the discretization of the emotion labels for C4.5. It resulted to having a majority class. Table 2 shows that class 1 is consistently the majority class for the data set. With a small window length, more instances are labelled with emotion value close to 0. We note, however that as window length is increased, the number of classes steadily balances out. For example, at $n = 1$, 84% of the data is labelled as class 1, but when $n = 90$, it is only 51%. This is the general trend for all the emotion models. At $n = 90$, the instances labelled as class 1 for the other emotion labels are as follows: 62.2% for Joy, 78.8% for Sadness, and 81.5% for Relaxation.

4.2 Important features used in C4.5

C4.5 builds a decision tree by finding features in the data that most effectively splits the data into subsets enriched in one class or the other. This causes a side effect of identifying music features that are most beneficial for classifying emotions.

Table 3 summarizes the features included in the trees generated by the algorithm using $n = 60$. The items are ordered according to the number of features present in the decision trees. A big portion of the features included are rhythmic features averaging 34.9% of the feature set. Features related to instrumentation also play a big part in identifying Stress unlike the other emotions. On the other hand, melody features are more important for Relaxation, Stress and Joy.

A closer inspection of the decision tree reveals that each emotion can be classified faster using a different ordering of music features. Table 4 shows the distribution of features found in the first 5 levels of the different decision

Category	Stress	Relaxation	Sadness	Joy
Rhythm	23.4%	13.5%	6.0%	14.0%
Pitch	0.0%	10.8%	9.0%	10.0%
Melody	4.3%	2.7%	1.5%	6.0%
Instrumentation	4.3%	2.7%	4.5%	4.0%
Texture	0.0%	0.0%	0.0%	0.0%
Dynamics	2.1%	2.7%	1.5%	0.0%
Others	0.0%	2.7%	0.0%	0.0%

Table 4. Distribution of features found in the first 5 levels of the decision trees of C4.5

trees. The Stress model mostly uses rhythmic features and 2 melodic features for the first 4 levels and uses Instrumentation for the 5th level. During the interview with the subject, when asked which parts of the songs are stressful, she explains that songs with electric guitar and rock songs in general are very stressful for her. Rock songs used in the dataset had a fast tempo and may be a factor as to the construction of the decision tree.

For relaxing music, the subject mentioned that there are specific parts of the songs that made her feel relaxed. These include introductory parts, transitions between chorus and verses, piano and harp instrumentals, and climactic parts of the song (i.e. last verse-chorus or bridge). Examining the decision tree for relaxation, we find that *Melodic Interval Histogram*, *Basic Pitch Histogram*, and *Music Position* are used for the first 3 levels, which are features that support the statements of the subject. Although emotion models for Joy and Sadness are available, a complete analysis of these cannot be done since the dataset was primarily focused on relaxing and stressful music.

4.3 Accuracy of Emotion labels

The manual emotion labels were also compared to the emotion values from ESAM. The average emotion value for each song was calculated and transformed into a 5-point scale. Comparing the manual annotations with the discretized continuous annotations, we find that only 25% of the emotion labels from EEG were the same with the manual annotations, 62% of the emotion labels from EEG slightly differed from the manual annotations, and 13% were completely opposite from what was originally reported. It is difficult to attribute error for the discrepancy. One possible cause could be the methodology for manual annotations. While the subject was doing the manual annotations, we observed that usually, she would only listen to the first 30 seconds of the song and in some cases skip to the middle of the song. It is possible that the manual annotation incompletely represents the emotion of the entire song.

It is also possible that the subject experienced a different kind of emotion unconsciously while listening to the music. For example some songs that were reported to be stressful turned out not stressful at all. We examined the emotion annotations and checked if there was any dependency between the values.

In Table 5 we can see that the subject treated the emotion Stress to be the bipolar opposite of Relaxation due to the high negative correlation value. Using ESAM, we find a similar situation but there is only a moderate negative correlation between the two as shown in Table 6. If we examine the other emotions, we find that Joy has a correlation with Relaxation and a negative correlation with Stress. This is consistently reported for both manual annotations and annotations using ESAM.

	Joy	Sadness	Relaxation	Stress
Sadness	-0.5638			
Relaxation	0.5870	0.0733		
Stress	-0.6221	-0.0555	-0.9791	
Familiarity	0.7190	-0.2501	0.5644	-0.6252

Table 5. Correlation of manual annotations

	Joy	Sadness	Relaxation	Stress
Sadness	-0.1187			
Relaxation	0.4598	-0.2338		
Stress	-0.4450	0.3100	-0.4223	
Familiarity	-0.0579	0.2956	-0.2343	0.5731

Table 6. Correlation of annotations using ESAM

Finally, we compared the amount of discrepancy between manual and automated annotations against the subject's familiarity with the song. We found that the discrepancy values for joyful and relaxing songs have a high correlation with familiarity : 0.6061 for Joy and 0.69551 for Relaxation. This implies that measurements of ESAM for Joy and Relaxation become more accurate when the subject is not familiar with the songs. It is possible that unfamiliar songs will help induce stronger emotions as compared to familiar music. This may be an important factor when using psychophysiological devices in measuring emotion.

5. CONCLUSION

This research focuses on building an emotion model for relaxing and stressful music. The model was built by extracting high-level music features from MIDI files using a windowing technique. The features were labelled using emotion values generated using EEG and ESAM. These values were also compared against manual emotion annotations. With the help of interviews conducted with the subject, we observe that EEG and ESAM can be used for annotating emotion in music especially when the subject experiences a strong intensity of that emotion. Familiarity of the subject with the song can affect genuine emotions.

Linear regression and C4.5 were used to build the different emotion models. Using a 10-fold cross-validation for evaluating the models, high accuracy with low relative absolute errors was obtained by using large window lengths encompassing between 38.4 seconds ($n = 60$) to 57.6 seconds ($n = 90$) of music.

6. FUTURE WORK

The current work involves one subject and it would be interesting to see if the model can be generalized using more subjects or, at the least, to verify if the current methodology will yield similar results when used with another subject.

Instead of using the average value for the emotion label, we intend to explore other metrics to summarize the emotion values for each window.

Further study on the music features is also needed. The current model uses both one-dimensional and multidimensional features. Experiments using only one set of the features will be performed. We also wish to explore the accuracy of the classification if low-level features were used instead of high-level features.

The window length greatly affects model accuracy. We have yet to investigate if there is a relationship between the average tempo of the song with window length. We hypothesize that slower songs would require longer window lengths to capture the same amount of information needed for fast songs. On the other hand, songs with fast tempo would need shorter window lengths.

Finally, this model will be integrated to a music recommendation system that can recommend songs which can induce similar emotions to the songs the user is currently listening to.

7. ACKNOWLEDGEMENTS

This research is supported in part by the Management Expenses Grants for National Universities Corporations through the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan, by the Global COE (Centers of Excellence) Program of MEXT, and by KAKENHI 23300059.

8. REFERENCES

[1] K. Ansari-Asl, G. Chanel, T. Pun, "A channel selection method for EEG classification in emotion assessment based on synchronization likelihood," *EUSIPCO 2007,15th Eur. Signal Proc. Conf.*, p. 1241–1245, 2007.

[2] G. Chanel, J. Kronegg, D. Grandjean, T. Pun: "Emotion assessment: Arousal evaluation using EEGs and peripheral physiological signals, *Lecture Notes in Computer Science*, Vol. 4105, p. 530, 2006.

[3] A. Gabrielsson, P.N. Juslin,: "Emotional expression in music." In R. J. Davidson, K. R. Scherer, and H. H. Goldsmith, editors, *Handbook of affective sciences*, New York: Oxford University Press, pp. 503-534, 2003.

[4] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten: "The WEKA Data Mining Software: An Update, SIGKDD Explorations," Vol. 11, No. 1, pp. 10–18, 2009.

[5] P.N. Juslin, J.A. Sloboda: "Handbook of music and emotion: theory, research, applications," Oxford University Press, 2010.

[6] Y.E. Kim, E.M. Schmidt, R. Migneco, B.G. Morton, P. Richardson, J. Scott, J.A. Speck, D. Turnbull: "Music Emotion Recognition: A State of the Art Review," *Proc. of the 11th ISMIR Conf.*, pp. 255–266, 2010.

[7] R. Legaspi, Y. Hashimoto, K. Moriyama, S. Kurihara, M. Numao: "Music Compositional Intelligence with an Affective Flavor," *Proc. of the 12th International Conference on Intelligent User Interfaces*, pp. 216–224, 2007.

[8] Y.-C. Lin, Y.-H. Yang, and H. H. Chen: "Exploiting online music tags for music emotion classification, *ACM Transactions on Multimedia Computing*, Communications, and Applications, Vol. 7S, No. 1, pp. 1–16, 2011.

[9] 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*, Vol. 34, No. 1, pp. 41–64, 2010.

[10] C. McKay, and I. Fujinaga: "jSymbolic: A feature extractor for MIDI files," *Proc. of the International Computer Music Conference*, pp. 302–305, 2006.

[11] E.R. Miranda, and A. Brouse: "Toward direct brain-computer musical interfaces," *New Interfaces for Musical Expression*, 2005.

[12] T. Musha, Y. Terasaki, H.A. Haque, and G.A. Ivanitsky: "Feature extraction from EEGs associated with emotions," *Journal of Artificial Life and Robotics*, Vol. 1, No. 1, pp. 15–19,1997.

[13] M. Numao, M. Kobayashi, and K. Sakaniwa: "Aquisition of human feelings in music arrangement," *Proc. of IJCAI '97*, pp. 268–273, 1997.

[14] M. Numao, S. Takagi, and K. Nakamura: Constructive adaptive user interfaces - Composing music based on human feelings," *Proc. of AAAI '02*, pp. 193–198, 2002.

[15] J.R. Quinlan: "Learning with continuous classes, *Proc. AI92, 5th Australian Joint Conference on Artificial Intelligence*, Adams & Sterling (eds.), World Scientific, Singapore, pp. 343–348, 1992.

[16] J.R. Quinlan: "C4.5: Programs for Machine Learning," Morgan Kaufmann Publishers, 1993.

[17] E. Schubert: "Affective, Evaluative, and Collative Responses to Hated and Loved Music," *Psychology of Aesthetics Creativity and the Arts*, Vol. 4, No. 1, pp. 36–46, 2010.

[18] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas: "Multilabel classification of music into emotions, in *Proc. of the 9th International Conference on Music Information Retrieval*, pp. 325–330, 2008.

MUSIC STRUCTURE ANALYSIS BY RIDGE REGRESSION OF BEAT-SYNCHRONOUS AUDIO FEATURES

Yannis Panagakis and Constantine Kotropoulos
Department of Informatics
Aristotle University of Thessaloniki
Box 451 Thessaloniki, GR-54124, Greece
{panagakis,costas}@aiia.csd.auth.gr

ABSTRACT

A novel unsupervised method for automatic music structure analysis is proposed. Three types of audio features, namely the mel-frequency cepstral coefficients, the chroma features, and the auditory temporal modulations are employed in order to form beat-synchronous feature sequences modeling the audio signal. Assume that the feature vectors from each segment lie in a subspace and the song as a whole occupies the union of several subspaces. Then any feature vector can be represented as a linear combination of the feature vectors stemming from the same subspace. The coefficients of such a linear combination are found by solving an appropriate ridge regression problem, resulting to the ridge representation (RR) of the audio features. The RR yields an affinity matrix with nonzero within-subspace affinities and zero between-subspace ones, revealing the structure of the music recording. The segmentation of the feature sequence into music segments is found by applying the normalized cuts algorithm to the RR-based affinity matrix. In the same context, the combination of multiple audio features is investigated as well. The proposed method is referred to as ridge regression-based music structure analysis (RRMSA). State-of-the-art performance is reported for the RRMSA by conducting experiments on the manually annotated Beatles benchmark dataset.

1. INTRODUCTION

The structural description of a music piece at the time scale of segments, such as intro, verse, chorus, bridge, etc. is referred to as the *musical form* of the piece [15]. Its derivation from the audio signal is a core task in music thumbnailing and summarization, chord transcription [10], learning of music semantics and music annotation [1], song segment retrieval [1], or remixing [6].

Human listeners analyze and segment music into meaningful parts by detecting the structural boundaries between the segments thanks to the perceived changes in timbre, tonality, and rhythm over the music piece. Music structure analysis extracts low-level feature sequences from the audio signal in order to model the timbral, melodic, and rhythmic content [15]. The segmentation of the feature sequences into structural parts is performed by employing methods based on either repetition, homogeneity, or novelty [1, 6, 7, 12, 14, 15, 17] to analyze a recurrence plot or a self-similarity distance matrix. For a comprehensive review on automatic music structure analysis systems the interested reader is referred to [4, 15] (and the references therein).

In this paper, a novel method for music structure analysis is proposed, which differs significantly from the previous methods. In particular, three types of audio features, namely the *mel-frequency cepstral coefficients* (MFCCs), the *Chroma* features, and the *auditory temporal modulations* (ATMs) are employed in order to form beat-synchronous feature sequences modeling the timbral, tonal, and rhythmic content of the music signal. It is reasonable to assume that due to the timbral, tonal, and rhythmic homogeneity within the music segments, the audio features extracted from a specific music segment are highly correlated and thus linearly dependent. Therefore, there is a linear subspace that spans the beat-synchronous audio features for each specific music segment implying that the sequence of feature vectors extracted from the whole music recording will lie in a union of as many linear subspaces as the music segments of this recording are. Accordingly, a feature vector extracted at the time scale of a beat can be represented as a linear combination of the feature vectors stemming from the subspace it belongs to. Formally, one solves an appropriate inverse problem in order to obtain the representation of each feature vector with respect to a dictionary, which is constructed by all the other feature vectors as atoms (i.e., column vectors). Here, it is proved that the joint *ridge representation* (RR) of the features drawn from a union of independent linear subspaces exhibits nonzero within-subspace affinities and zero between-subspace affinities. That is, a *ridge regression* [1] problem is solved, which admits a unique and closed-form solution. The segmentation of the feature sequence into music segments is revealed by applying the normalized cuts spectral clustering algorithm [16] to the RR-based affin-

[1] Ridge regression is also known as Tikhonov regularization.

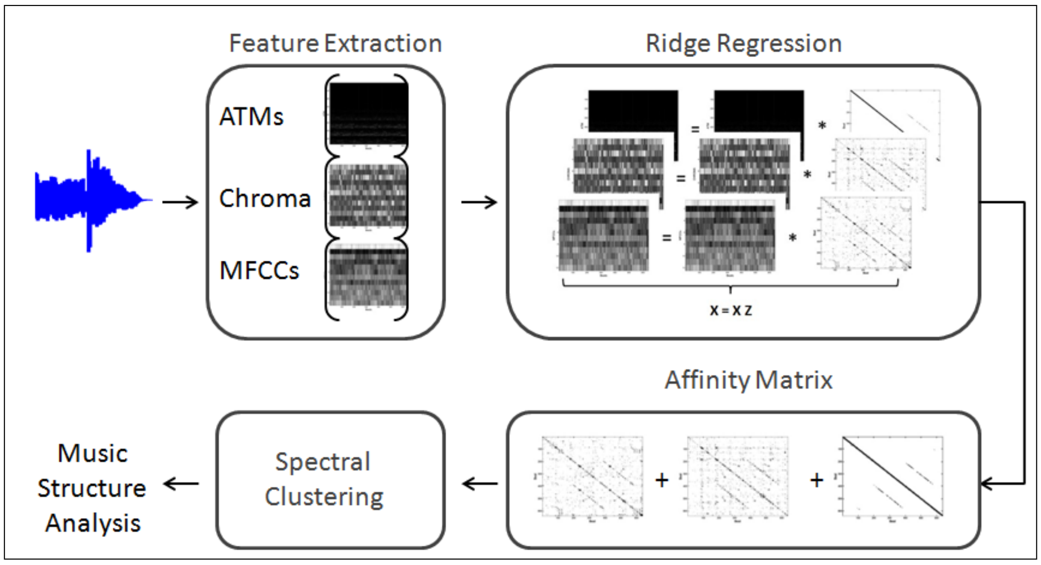


Figure 1. Each music recording is modeled by three audio features, namely the MFCCs, the Chroma features, and the ATMs resulting to three beat-synchronous feature matrices. The RR is derived for each feature matrix and three affinity matrices are obtained as described in Section 3. A cross-feature affinity matrix is obtained by linearly combining the affinity matrices obtained for the individual features. The segmentation of the music recording into music segments is obtained by applying the normalized cuts spectral clustering algorithm to the cross-feature RR-based affinity matrix.

ity matrix. Provided that music segments can seldom be revealed efficiently by resorting to a single feature, multiple features are extracted from each music recording and the cross-feature information is utilized in order to obtain a reliable music segmentation. To this end, a cross-feature RR-based affinity matrix is constructed by linearly combining the RR-based affinity matrices obtained for each individual feature. Again, the segmentation of the feature sequence into music segments is obtained by applying the normalized cuts to the cross-feature RR-based affinity matrix. The proposed method is referred to as *ridge regression-based music structure analysis* (RRMSA) and it is outlined in Fig. 1.

The performance of the RRMSA is assessed by conducting experiments on the manually annotated Beatles dataset. The RRMSA is demonstrated to yield a state-of-the-art performance.

The remainder of the paper is as follows. In Section 2, the audio features employed are briefly described. The RRMSA is detailed in Section 3. Dataset, evaluation metrics, and experimental results are presented in Section 4. Conclusions are drawn in Section 5.

2. AUDIO FEATURE REPRESENTATION

The variations between different music segments are captured by extracting three audio features from each monaural music recording sampled at 22.05-kHz. In particular, the MFCCs, the Chroma features, and the ATMs are employed.

1) The MFCCs encode the timbral properties of the music signal by parameterizing the rough shape of spectral envelope. Following [14], the MFCC 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 the frequency bands is reduced by applying the discrete cosine transform along the log-energies of the bands. The zeroth order coefficient is discarded yielding a sequence of 12-dimensional MFCCs vectors.

2) The Chroma features are able to characterize 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. They are calculated by employing 92.9 ms frames with a hope size of 23.22 ms as follows. First, the salience of 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 sequence of 12-dimensional chroma vectors.

3) The ATMs carry important time-varying information of the audio signal [11]. They are obtained by modeling the path of human auditory processing as a two-stage process. In the first stage, which models the early auditory system, the acoustic signal is converted into a time-frequency distribution along a logarithmic frequency axis, the so-called *auditory spectrogram*. The early auditory system is modeled by Lyons' passive ear model [9] employing 96 frequency channels ranging from 62 Hz to 11 kHz. The auditory spectrogram is then downsampled along the time axis in order to obtain 10 feature vectors between two successive beats. The underlying temporal modulations of the music signal are derived by applying a biorthogonal wavelet filter along each temporal row of the auditory spectrogram, where its mean has been previously subtracted, for 8 discrete rates $r \in \{2, 4, 8, 16, 32, 64, 128, 256\}$ Hz ranging from slow to fast temporal rates. Thus, the entire auditory spectrogram is modeled by a three-dimensional representation of frequency, rate, and time which is then unfolded[2] along the time-mode in order to obtain a sequence of $96 \times 8 = 728$-dimensional ATMs.

Postprocessing. Sequences of *beat-synchronous* feature vectors are obtained by averaging the feature vectors over the beat frames. The latter are found by using the beat tracking algorithm described in [5]. Each row of the beat-synchronous feature matrix is filtered by applying an average filter of length 8. Finally, each feature vector undergoes a normalization in order to have zero-mean and unit ℓ_2 norm.

3. MUSIC STRUCTURE ANALYSIS BASED ON RIDGE REGRESSION

In this section, the RRMSA is detailed. Let a given music recording of K music segments be represented by a sequence of N beat-synchronous audio feature vectors of size d, i.e., $\mathbf{X} = [\mathbf{x}_1|\mathbf{x}_2|\ldots|\mathbf{x}_N] \in \mathbb{R}^{d\times N}$. The perceived timbral, tonal, and rhythmic homogeneity within a music segment implies that the audio features extracted from this music segment are highly correlated, exhibiting linear dependence. This motivated us to assume that beat-synchronous feature vectors belonging to the same music segment live into the same subspace. Therefore, if a music recording consists of K music segments, the sequence of N beat-synchronous audio feature vectors (i.e., the columns of $\mathbf{X}$) are drawn from a union of K independent linear subspaces of unknown dimensions. Thus, each feature vector can be represented as a linear combination of feature vectors drawn from the same subspace. That is, $\mathbf{X} = \mathbf{X}\mathbf{Z}$, where $\mathbf{Z} \in \mathbb{R}^{N\times N}$ is the representation matrix, which contains the linear combination coefficients in its columns [3]. Clearly, $z_{ij} = 0$, if $\mathbf{x}_i$ and $\mathbf{x}_j$ lie on different subspaces and nonzero otherwise.

Such a representation matrix $\mathbf{Z}$ can be found by solving a least-squares problem regularized by the Frobenius norm (denoted by $\|.\|_F$), the so-called *ridge regression* problem:

$$\underset{\mathbf{Z}}{\operatorname{argmin}} \; \|\mathbf{X} - \mathbf{X}\,\mathbf{Z}\|_F^2 + \lambda\|\mathbf{Z}\|_F^2. \tag{1}$$

The unique solution of the unconstrained convex problem (1) is referred to as *ridge representation* (RR) matrix and it is given in closed-form by:

$$\mathbf{Z} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}. \tag{2}$$

Technically, the desired property of the RR matrix to admit nonzero entries for within-subspace affinities and zero entries for between-subspace affinities is enforced by the regularization term $\lambda\|\mathbf{Z}\|_F^2$ in (1) as proved in Theorem 1, which is a consequence of Lemma 1. This result indicates that if the data follow subspace structures (i.e., come from a union of independent subspaces), the correct identification of the subspaces can be obtained accurately, fast, and in closed form by solving (1) without imposing sparsity or other constraints on the data model.

Lemma 1 [2]. For any four matrices $\mathbf{B}, \mathbf{C}, \mathbf{D}$, and $\mathbf{F}$ of compatible dimensions,

$$\left\|\begin{bmatrix} \mathbf{B} & \mathbf{C} \\ \mathbf{D} & \mathbf{F} \end{bmatrix}\right\|_F^2 \geq \left\|\begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{bmatrix}\right\|_F^2 = \|\mathbf{B}\|_F^2 + \|\mathbf{F}\|_F^2. \tag{3}$$

Theorem 1. Assume the columns of $\mathbf{X}$ (i.e., the feature vectors) are drawn from a union of K linear independent subspaces of unknown dimensions. Without loss of generality, one may decompose $\mathbf{X}$ as $[\mathbf{X}_1|\mathbf{X}_2|\ldots|\mathbf{X}_K] \in \mathbb{R}^{d\times N}$, where the columns of $\mathbf{X}_k \in \mathbb{R}^{d\times N_k}$, $k = 1, 2, \ldots, K$ correspond to the N_k feature vectors originating from the kth subspace. The minimizer of (1) is block-diagonal.
The proof can be found in the Appendix.

Let $\mathbf{Z} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ be the singular value decomposition (SVD) of $\mathbf{Z}$. Set $\tilde{\mathbf{U}} = \mathbf{U}(\mathbf{\Sigma})^{\frac{1}{2}}$ and $\mathbf{M} = \tilde{\mathbf{U}}\tilde{\mathbf{U}}^T$. A RR-based nonnegative symmetric affinity matrix $\mathbf{W} \in \mathbb{R}_+^{N\times N}$ has elements [8]:

$$w_{ij} = m_{ij}^2. \tag{4}$$

The segmentation of the columns of $\mathbf{X}$ into K clusters (i.e., music segments) is performed by employing the normalized cuts [16] onto the RR-based affinity matrix $\mathbf{W}$.

Since the music segments cannot be accurately derived by resorting to one feature, cross-feature information is expected to produce a more reliable music segmentation. Let $\mathbf{W}_m$, $\mathbf{W}_c$, and $\mathbf{W}_a$ be the RR-based affinity matrix obtained, when the MFCCs, the Chroma, and the ATMs are employed, respectively. A cross-feature RR-based affinity matrix $\mathbf{W}_{cf} \in \mathbb{R}_+^{N\times N}$ can be constructed by:

$$\mathbf{W}_{cf} = 1/3(\mathbf{W}_m + \mathbf{W}_c + \mathbf{W}_a), \tag{5}$$

or any other combination of the individual affinity matrices. The segmentation of the music recording can be obtained by applying the normalized cuts [16] to the cross-feature RR-based affinity matrix $\mathbf{W}_{cf}$.

In general, the number of segments K in a music recording is unknown and thus it is reasonable to be estimated. To this end, the soft-thresholding approach is employed [8]. That is, the estimated number of segments $\bar{K}$ is found by:

$$\bar{K} = N - \text{int}\Big(\sum_{i=1}^{N} f_\tau(\sigma_i)\Big). \tag{6}$$

The function int(.) returns the nearest integer of a real number, $\{\sigma_i\}_{i=1}^N$ denotes the set of the singular values of the Laplacian matrix derived by the corresponding affinity matrix, and f_τ is the soft-thresholding operator defined as $f_\tau(\sigma) = 1$ if $\sigma \geq \tau$ and $\log_2(1 + \frac{\sigma^2}{\tau^2})$, otherwise. Clearly, the threshold $\tau \in (0, 1)$.

[2] The tensor unfolding can be implemented in Matlab by employing the `tenmat` function of the MATLAB Tensor Toolbox available at: `http://csmr.ca.sandia.gov/~tgkolda/TensorToolbox/`.

[3] Due to the assumptions stated at the beginning of Section 3, the matrix $\mathbf{X}$ does not has full column rank. Therefore, $\mathbf{X} = \mathbf{X}\mathbf{Z}$ does not admit the identity matrix as solution.

4. EXPERIMENTAL EVALUATION

4.1 Dataset, Evaluation Procedure, and Evaluation Metrics

Beatles dataset[4]: The dataset consists of 180 songs by The Beatles. The songs are annotated by the musicologist Alan W. Pollack. Segmentation time stamps were inserted at Universitat Pompeu Fabra (UPF). Each music recording contains on average 10 segments from 5 unique classes [17].

The structure segmentation is obtained by applying the RRMSA to each individual feature sequence as well as to all possible feature combinations. In Fig. 2, sample RR-based affinity matrices are depicted. Two sets of experiments were conducted on the Beatles dataset. First, following the experimental setup employed in [1,3,6,7,12,14,17], the number of clusters (i.e., segments) K was kept constant and equal to 4. Second, for each music recording, the number of segments was estimated by (6). The optimal values of the various parameters (i.e., λ, τ) were determined by a grid search over 10 randomly selected music recordings of the dataset.

In order to compare fairly the RRMSA with the state-of-the-art music structure analysis methods, the segment labels are evaluated by employing the *pairwise F-measure* as in [3, 6, 7, 12, 14, 17]. The pairwise F-measure is a standard evaluation metric for clustering algorithms. It is defined as the harmonic mean of the pairwise precision and recall. The segmentation results and the reference segmentation (i.e., the ground truth) are handled at the time scale of beats. Let $\mathbb{F}_A$ be the set of identically labeled pairs of beats in a recording according to the music structure analysis algorithm and $\mathbb{F}_H$ be the set of identically labeled pairs in the human reference segmentation. The pairwise precision, PP, the pairwise recall, PR, and the pairwise F-measure, PF, are defined as:

$$PP = \frac{|\mathbb{F}_A \cap \mathbb{F}_H|}{|\mathbb{F}_A|}, \tag{7}$$

$$PR = \frac{|\mathbb{F}_A \cap \mathbb{F}_H|}{|\mathbb{F}_H|}, \tag{8}$$

$$PF = 2 \cdot \frac{PP \cdot PR}{PP + PR}, \tag{9}$$

where $|.|$ denotes the set cardinality.

4.2 Experimental Results

The segment-type labeling performance of the RRMSA on the Beatles dataset is summarized in Table 1 for a fixed number of segments (i.e., $K = 4$) as in [3, 6, 7, 12, 14, 17]. By inspecting Table 1, one can see that the ATMs are more suitable for music structure analysis than the MFCCs and the Chroma features. Furthermore, the latter two features lead to an undesirable over-segmentation of the music recordings. Similar findings were reported in [12]. The best result reported for segment-type labeling on the Beatles dataset is obtained here, when the RR-based affinity matrices of the MFCCs and the ATMs are combined. Interestingly to note that by employing cross-feature affinity matrices the average number of segments approaches 10 (i.e., the actual average number of segments according to the ground-truth), although no constraints have been enforced during clustering. In addition to the very promising performance of the RRMSA with respect to PF, it is worth mentioning that the RRMSA is fast. The average CPU time for the calculation of the RR-based affinity matrix is 0.858 CPU seconds.

Features	**Parameters**	PF	**Segments**
MFCCs	($\lambda = 0.3$)	0.54	37.1
Chroma	($\lambda = 0.1$)	0.57	36.7
ATMs	($\lambda = 0.1$)	**0.61**	6.1
MFCCs & Chroma	($\lambda = 0.3, 0.1$)	0.55	20.6
MFCCs & ATMs	($\lambda = 0.3, 0.1$)	**0.63**	7.1
Chroma & ATMs	($\lambda = 0.1, 0.1$)	0.60	8.1
MFCCs & Chroma & ATMs	($\lambda = 0.3, 0.1, 0.1$)	0.61	8.8

Table 1. Segment-type labeling performance of the RRMSA on the Beatles dataset with fixed $K = 4$.

The best result obtained by the RRMSA on the Beatles dataset with respect to PF (i.e., 0.63) outperforms the results obtained by the majority of the state-of-the-art music segmentation methods listed in Table 2 on the same dataset previously. The results were rounded down to the nearest second decimal digit. It is seen that the RRMSA admits the highest PF when the MFCCs and the ATMs are combined. Similarly, MFCCs combined with Chroma yielded the top PF in [3] and [6]. Similar conclusions were drawn in [13], when multiple audio features were combined. It is worth mentioning that the RRMSA does not involve any postprocessing based on music knowledge, such as eliminating too short segments or restricting the segment length to improve the accuracy of music segmentation. This is not the case for the methods in [7] and [14]. Furthermore, the RRMSA involves only one parameter in contrast to methods [3, 7, 14, 17], where the tuning of multiple parameters is needed.

The segment-type labeling performance of the RRMSA on the Beatles dataset, when K is estimated by (6), is reported in Table 3. Again, the use of the ATMs for music representation makes the RRMSA to achieve better performance than that when either the MFCCs or the Chroma features are used. By combining the ATMs and the MFCCs, the PF for the RRMSA reaches 0.60. In this case, the estimated average number of segments equals the actual av-

Reference	**Features**	PF
[3]	MFCCs & Chroma	**0.63**
[6]	MFCCs & Chroma	0.62
[17]	Chroma	0.60
[14]	MFCCs	0.60
[12]	ATMs	0.59
Method in [7] as evaluated in [14]	MPEG-7	0.58

Table 2. Segment-type labeling performance on the Beatles dataset obtained by state-of-the-art methods with fixed $K = 4$.

[4] http://www.dtic.upf.edu/ perfe/annotations/sections/license.html

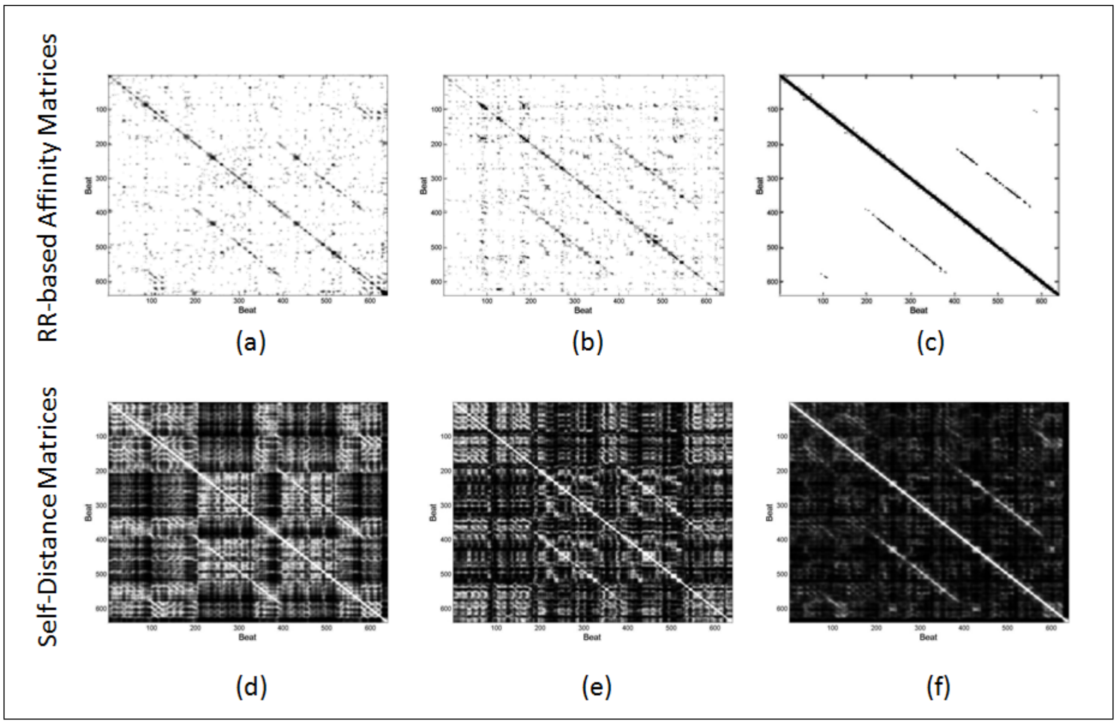


Figure 2. RR-based affinity and self-distance matrices of beat-synchronous feature vectors extracted from the Anna (Go to Him) by The Beatles when employing the MFCCs (a) and (d), the Chroma features (b) and (e), or the ATMs (c) and (f). The negative image of the affinity matrices is depicted. It is obvious that RR-based affinity matrices provide more clear and noise-free structural information than the self-distance matrices for all features.

erage number of segments according to the ground-truth (i.e., 10). This result indicates that it is possible to perform a robust unsupervised music structure analysis in a fully automatic setting.

Further details related to the estimation of K by employing various audio features and their combinations are shown in Table 4. The absolute error is defined as $|\bar{K} - K_g|$, where K_g is the actual number of segments based on the ground-truth. The prediction rate refers to the ratio of the number of music recordings where the number of segments was predicted correctly over the total number of music recordings in the dataset. If we consider the value $\bar{K} = K_g \pm 1$ as the correct number of predicted segments, then we obtain the Proximal Prediction Rate (PPR) (i.e., the last column in Table 4). The results presented in Table 4 indicate that the combination of the MFCCs and the ATMs yields the lowest absolute error, resulting to the highest prediction rate and thus the highest segmentation accuracy.

Features	Parameters	PF	Segments
MFCCs	$(\lambda = 0.3, \tau = 0.7)$	0.54	24.9
Chroma	$(\lambda = 0.1, \tau = 0.64)$	0.48	26.6
ATMs	$(\lambda = 0.1, \tau = 0.64)$	**0.59**	6.4
MFCCs + Chroma	$(\lambda = 0.3, 0.1, \tau = 0.7)$	0.59	12.2
MFCCs & ATMs	$(\lambda = 0.3, 0.1, \tau = 0.23)$	**0.60**	10.0
Chroma & ATMs	$(\lambda = 0.3, 0.1, \tau = 0.27)$	0.56	12.8
MFCCs & Chroma & ATMs	$(\lambda = 0.3, 0.1, \tau = 0.33)$	0.53	20.0

Table 3. Segment-type labeling performance of the RRMSA on the Beatles dataset for automatically estimated K.

Features	Absolute Error	Prediction Rate (%)	PPR (%)
MFCCs	**1.24**	**25.26**	**65.59**
Chroma	1.88	15.59	42.47
ATMs	1.72	18.81	52.68
MFCCs & Chroma	1.88	15.60	42.47
MFCCs & ATMs	**1.15**	**30.10**	**73.11**
Chroma& ATMs	1.43	22.58	61.82
MFCCs & Chroma & ATMs	1.22	26.34	67.20

Table 4. Accuracy of the estimation of the number of segments, K, on the Beatles dataset.

5. CONCLUSIONS

In this paper, a robust and fast method for music structure analysis (i.e., the RRMSA) has been proposed. In particular, the ridge regression representation of the MFCCs, the Chroma, and the ATMs have been used to derive affinity matrices, where the normalized cuts algorithm has been applied to obtain the music structure. Among the three features, the ATMs and the MFCCs have been proved the most powerful. By linearly combining the RR-based affinity matrices of the MFCCs and the ATMs and applying next the normalized cuts, state-of-the-art performance on the Beatles dataset has been reported for a fixed number of segments. Furthermore, an accurate method to estimate the number of segments in each music recording has been developed, enabling a fully automatic unsupervised music structure analysis.

APPENDIX: PROOF OF THEOREM 1

Let us denote by $\{\mathfrak{S}_1, \mathfrak{S}_2, \ldots, \mathfrak{S}_K\}$, a collection of K independent subspaces. The direct sum of a collection of K subspaces is denoted by $\oplus_{k=1}^{K}\mathfrak{S}_k$. Let $\mathbf{Z}$ be the unique minimizer of (1) and $\mathbf{D}$ be a block-diagonal matrix with elements $d_{ij} = z_{ij}$, if $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same subspace (i.e., music segment here), and $d_{ij} = 0$ otherwise. We can define $\mathbf{Q} = \mathbf{Z} - \mathbf{D}$. Without loss of generality let us suppose that $\mathbf{x}_j$ belongs to the ith subspace, i.e., $\mathbf{x}_j = [\mathbf{XZ}]_j \in \mathfrak{S}_i$. We can write $\mathbf{Q}$ as the sum of two matrices $\mathbf{Q}_1$ and $\mathbf{Q}_2$ whose supports are on disjoint subsets of indices, such that $[\mathbf{XQ}_1]_j \in \mathfrak{S}_i$ and $[\mathbf{XQ}_2]_j \in \oplus_{k\neq i}^{K}\mathfrak{S}_k$. We show that $\mathbf{Q}_2 = \mathbf{0}$. For the sake of contradiction, we assume that $\mathbf{Q}_2 \neq \mathbf{0}$. Since $\mathbf{Z} = \mathbf{D} + \mathbf{Q}_1 + \mathbf{Q}_2$, we have $\mathbf{x}_j = [\mathbf{XZ}]_j = [\mathbf{X}(\mathbf{D} + \mathbf{Q}_1)]_j + [\mathbf{XQ}_2]_j$. Since $\mathbf{x}_j \in \mathfrak{S}_i$ and $[\mathbf{X}(\mathbf{D} + \mathbf{Q}_1)]_j \in \mathfrak{S}_i$, by the independence of subspaces, $\mathfrak{S}_i \cap \oplus_{k\neq i}^{K}\mathfrak{S}_k = \{0\}$, we should have $[\mathbf{XQ}_2]_j = \mathbf{0}$.

But $[\mathbf{XQ}_2]_j = \mathbf{0}$ implies, $\mathbf{x}_j = [\mathbf{XZ}]_j = [\mathbf{X}(\mathbf{D}+\mathbf{Q}_1)]_j$ and hence $\mathbf{D} + \mathbf{Q}_1$ is feasible solution of (1). By the fact that the supports of Q_1 and Q_2 are disjoint subsets of indices and Lemma 1, $\|\mathbf{D} + \mathbf{Q}_1\|_F^2 \leq \|\mathbf{D} + \mathbf{Q}_1 + \mathbf{Q}_2\|_F^2 = \|\mathbf{Z}\|_F^2$. That is $\mathbf{D} + \mathbf{Q}_1$, is a feasible solution of (1) attaining a smaller Frobenius norm than $\|\mathbf{Z}\|_F^2$, which contradicts the optimality of $\mathbf{Z}$. Thus, $\mathbf{Q}_2 = \mathbf{0}$, meaning that only the blocks that correspond to vectors in the true subspaces are nonzero.

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] R. Bhatia and F. Kittaneh. Norm inequalities for partitioned operators and an application. *Math. Ann.*, 287(1):719–726, 1990.

[3] R. Chen and L. Ming. Music structural segmentation by combining harmonic and timbral information. In *Proc. 12th Int. Conf. Music Information Retrieval*, pages 477–482, Miami, USA, 2011.

[4] 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.

[5] D. Ellis. Beat tracking by dynamic programming. *J. New Music Research*, 36(1):51–60, 2007.

[6] F. Kaiser and T. Sikora. Music structure discovery in popular music using non-negative matrix factorization. In *Proc. 11th Int. Conf. Music Information Retrieval*, pages 429–434, Utrecht, The Netherlands, 2010.

[7] 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.

[8] G. Liu, Z. Lin, S. Yan, J. Sun, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2011. arXiv:1010.2955v4 (preprint).

[9] 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.

[10] M. Mauch, K. Noland, and S. Dixon. Using musical structure to enhance automatic chord transcription. In *Proc. 10th Int. Conf. Music Information Retrieval*, pages 231–236, Kobe, Japan, 2009.

[11] 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.

[12] Y. Panagakis, C. Kotropoulos, and G. R. Arce. l1-graph based music structure analysis. In *Proc. 12th Int. Conf. Music Information Retrieval*, pages 495–500, Miami, USA, 2011.

[13] J. Paulus and A. Klapuri. Acoustic features for music piece structure analysis. In *Proc. 11th Int. Conf. Digital Audio Effects*, pages 309–312, Espoo, Finland, 2008.

[14] 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.

[15] J. Paulus, M. Müller, and A. Klapuri. Audio-based music structure analysis. In *Proc. 11th Int. Conf. Music Information Retrieval*, pages 625–636, Utrecht, The Netherlands, 2010.

[16] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[17] R. Weiss and J. Bello. Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization. In *Proc. 11th Int. Conf. Music Information Retrieval*, pages 123–128, Utrecht, The Netherlands, 2010.

SCORE-INFORMED LEADING VOICE SEPARATION FROM MONAURAL AUDIO

Cyril Joder, Björn Schuller
Institute for Human-Machine Communication, Technische Universität München, Germany
cyril.joder@tum.de, schuller@tum.de

ABSTRACT

Separating the leading voice from a musical recording seems to be natural to the human ear. Yet, it remains a difficult problem for automatic systems, in particular in the blind case, where no information is known about the signal. However, in the case where a musical score is available, one can take advantage of this additional information. In this paper, we present a novel application of this idea for leading voice separation exploiting a temporally-aligned MIDI Score.

The model used is based on Nonnegative Matrix Factorization (NMF), whose solo part is represented by a source-filter model. We exploit the score information by constraining the source activations to conform to the aligned MIDI file. Experiments run on a database of real popular songs show that the use of these constraints can significantly improve the separation quality, in terms of both signal-based and perceptual evaluation metrics.

1. INTRODUCTION

Extracting the main melody from a musical signal can be of interest, for example, for the remixing of a musical piece or the creation of a 'play-along' version of a recording, in the context of karaoke or classical concertos. Whereas this task is quite natural to the human ear, the automated solving of such a separation problem is notoriously difficult.

In the past, many works have considered the separation of musical sources as a blind audio source separation problem, assuming only general knowledge about the sources, such as temporal and harmonicity priors [16] or timbre information [14]. On the other hand, audio source separation approaches which integrate specific information about the content of each recording (see [15] for example) have recently received a large interest. In the case of music, valuable information about the sources can be found in the score when it is available. Hence, the topic of score-informed source separation, exploiting a temporally aligned score, has recently emerged. The score information is used to initialize the parameters of a model, which are then re-estimated in order to precisely match the data. Several kinds of models have been proposed, such as a sinusoidal model [12], Nonnegative Matrix Factorization (NMF) [4] or Probabilistic Latent Component Analysis (PLCA) [6]. The model of [10] exploits MIFI syntheses of the score, and operates a trade-off between fidelity to the synthesized sound and to the actual data to be separated. In [1], a multipitch estimator is used to model the spectral shape of each note. The authors of [9] employ a parametric NMF model which estimates a constant harmonic structure for each source. The specific problem of extracting the main melody part has also been addressed in [7] with an NMF-like probabilistic model, where each note is represented as a harmonic template.

In the present work, we exploit the physically-motivated source-filter NMF model proposed in [2], which is specifically designed for the extraction of the leading voice. We take advantage of the aligned score through time and pitch constraints. These constraints are similar to the ones already applied in [8] to a source-filter NMF model. However, while the latter work exploits the information given by a multipich estimator, we make use of actual MIDI transcriptions of the pieces. We evaluate the benefit of the score-based information on a database of real data, composed of nine multi-track recordings of popular songs. The scores are constituted by real-life MIDI scores, which are synchronized using a state-of-the-art alignment algorithm [11]. Several signal-based and perceptual evaluation criteria are used and the results show that both the interferences and the separation artifacts are reduced thanks to the score information. Furthermore, the use of time-frequency constraints applied on the leading voice components allows for a multi-pass approach for the removing of the reverberated voice, which improves the perceived quality of the separated accompaniment.

The rest of this paper is organized as follows: in Section 2 we present the source-filter NMF model used as baseline system for blind leading voice separation. Section 3 explains how the aligned MIDI score is exploited in the proposed methods for score-informed leading voice separation. We finally report the performed experiments in Section 4, before drawing some conclusions.

2. BASELINE SYSTEM: BLIND SEPARATION

As baseline system for the blind separation of the leading voice, we use the model proposed in [2]. Let us now detail the main features of this model.

2.1 Signal Model

Let $\mathbf{S}$ be the matrix representing the short-time power spectrum of the musical recording, which is assumed to be of single-channel nature. We suppose that this matrix can be decomposed as:

$$\mathbf{S} = \mathbf{S}^V + \mathbf{S}^A, \tag{1}$$

where $\mathbf{S}^V$ and $\mathbf{S}^A$ are the short-time power spectrum of the leading voice and of the musical accompaniment, respectively. Furthermore, a source-filter model is assumed for the solo part. Thus, the matrix $\mathbf{S}^V$ can be written as the element-wise product of a 'source' matrix $\mathbf{S}^{F_0}$ by a 'filter' matrix $\mathbf{S}^\Phi$:

$$\mathbf{S}^V = \mathbf{S}^\Phi \odot \mathbf{S}^{F_0}, \tag{2}$$

where $\odot$ denotes the element-wise product.

For the contributions $\mathbf{S}^A$, $\mathbf{S}^{F_0}$ and $\mathbf{S}^\Phi$, an NMF model is assumed. Each of these terms is modeled as the product of two nonnegative matrices $\mathbf{W}$ and $\mathbf{H}$. The former is a dictionary of spectrum templates (in columns) and the latter contains the corresponding activation amplitudes over time. Finally, we have:

$$\mathbf{S} = \left(\mathbf{W}^\Phi \mathbf{H}^\Phi\right) \odot \left(\mathbf{W}^{F_0} \mathbf{H}^{F_0}\right) + \mathbf{W}^A \mathbf{H}^A \tag{3}$$

The 'source spectral shapes' of matrix $\mathbf{W}^{F_0}$ are set to fixed harmonic combs with logarithmically-spaced fundamental frequencies, with 20 F_0 values per semitone (in order to take into account tuning variations or vibratos) between 100 Hz and 800 Hz. This range is sufficient for most popular songs. In order to estimate smooth filters, the elements of the filters dictionary $\mathbf{W}^\Phi$ are modeled as the combination of overlapping Hann windows (in the frequency domain). As default parameters, the size of the filter dictionary is 10 and the rank of the accompaniment decomposition is set to 40.

2.2 Separation Strategy

The leading voice separation procedure consists of several steps. First, the matrices of eq. (3) (except for $\mathbf{W}^{F_0}$) are estimated from the processed signal using an NMF optimization algorithm based on the Itakura-Saito divergence. From this first result, only the estimated main source activation matrix $\mathbf{H}^{F_0}$ is kept. It can be interpreted as the instantaneous 'power' of the corresponding fundamental frequencies. Since the signal of interest is the *leading* voice, it is assumed to correspond to the dominant pitch. However, in order to avoid spurious 'jumps' in the case where the voice stops or if another instrument has a higher energy in a distant pitch, a tracking algorithm is used to estimate the whole sequence of F_0 values for the leading voice.

In the second step, another estimation of the model (3) is performed, in which $\mathbf{H}^{F_0}$ is constrained so that only the values around the tracked pitch are allowed to be non-zero. Finally, the leading voice is reconstructed by Wiener filtering. The estimate $\hat{\mathbf{S}}^V$ of the short-time power spectrum is then given by:

$$\hat{\mathbf{S}}^V = \frac{\left(\mathbf{W}^\Phi \mathbf{H}^\Phi\right) \odot \left(\mathbf{W}^{F_0} \mathbf{H}^{F_0}\right)}{\left(\mathbf{W}^\Phi \mathbf{H}^\Phi\right) \odot \left(\mathbf{W}^{F_0} \mathbf{H}^{F_0}\right) + \mathbf{W}^A \mathbf{H}^A} \tag{4}$$

and the time-domain signal is retrieved by inverse Fourier transform (using the phase of the original mixture) and overlap-add.

3. EXPLOITATION OF THE SCORE INFORMATION

We now explain how we exploit the additional information given by the aligned musical score.

3.1 Information Conveyed by the Score

By musical score, we designate a set of notes characterized by their pitch, onset time and duration. In this work, we additionally assume that the notes in the score corresponding to the leading voice can be discriminated from the other notes. This is the case in most score MIDI files, where the instruments correspond to different *tracks*. Hence, the score can provide valuable information for the leading voice separation task. However, the score employs a temporal scale (expressed in beats), whose correspondence with the actual time in second is in general both unknown and variable. Fortunately, systems for accurate music-to-score alignment have been proposed to overcome this problem [5, 11, 13].

The aligned score then provides the pitch, onset and offset time of the notes played in the musical piece. Nevertheless, some limitations have to be taken into account. In particular, the pitches of the score are expressed in semitones, which constitutes a coarser frequency resolution than the short-time Fourier transform representation of the audio. Furthermore, there can be various sources of imprecision or mismatch between the score and the actual recording. For example, vibratos can strongly alter the fundamental frequency of a note. There may also be transcription errors or different interpretations of the music. In particular, the synchronization between the instruments or voices in polyphonic music may not be perfect, yielding a temporal indeterminacy and thus a possible imprecision in the alignment. For these reasons, the information conveyed by the aligned score cannot be fully trusted at a precise level. Nevertheless, it can be used at a coarser level, so as to narrow the search for the voice components in the spectrogram. In this work, we use only the 'voice track' of the aligned score. Indeed, in most cases of popular music, no reference musical score exists and the available transcriptions can often resemble 'lead sheet' scores, which focus on the main melody and only describe the global harmony of the accompaniment.

3.2 Time and Pitch Constraints

We propose to exploit the score information through two types of constraints applied in the model (3). The first approach only makes use of the information regarding whether the leading voice is present or not in each frame.

This corresponds to the case where the pitch of the aligned score is not sufficiently reliable. This *temporal constraint* consists in forcing all the activations of the source element (contained in the matrix $\mathbf{H}^{F_0}$) to be equal to zero when the voice is known to be absent. A time tolerance window is allowed, in order to overcome the possible temporal imprecision of the score alignment. The voice is then considered as absent in a frame when no note of the aligned score is present inside a temporal window of length θ_t. The value of this tolerance threshold is a trade-off between two goals. If it is short, one may 'miss' the voice in the case where the score is imperfectly aligned. On the other hand, a long tolerance window may result in the extraction of another instrument as the leading voice, when the latter is absent.

The second approach takes advantage of both, time and pitch information, on the aligned score. As previously, the constraint used consists in forcing zero values of the source activation matrix where the source is known to be absent. This implies the use of an additional tolerance threshold θ_f on the fundamental frequency, in order to limit the pitch imprecisions. Hence, a component $\mathbf{H}^{F_0}_{i,j}$ or the source activation matrix (corresponding to fundamental frequency i in frame j) is allowed to be non-zero only if there is a note in the aligned score, of pitch p, onset time t_1 and offset time t_2, such that:

$$|p - i| \leq \theta_f \text{ and } t_1 - \theta_t \leq j \leq t_2 + \theta_t. \qquad (5)$$

4. EXPERIMENTS

4.1 Database and Settings

The database used in this work is composed of nine separated-track versions of well-known popular songs, for which a MIDI transcription was found on the internet. The list of the songs is displayed in Table 1. Unfortunately, these data cannot be shared due to copyright restrictions. In all these pieces, the source of interest is a human voice. Some of the songs contain vocal harmonies (several vocal parts), which introduce an ambiguity about the determination of the main source. In some others, mistakes are found in the MIDI score, where some vocal parts are not transcribed. In the corresponding pieces, only an excerpt where these problems do not occur has been used. All the files were converted to mono signals with 44.1 kHz sampling rate. For each piece, the file to be processed was created by linearly mixing the leading voice with the accompaniment. This procedure is much simpler than the mixing phase of professionally processed music, which often involves additional filtering or dynamic range compression. However, it was necessary to ensure that the final mixtures perfectly correspond to the separated tracks.

The MIDI scores were aligned by the method presented in [11]. This results in very accurate alignment, and the imprecision between the recording and the synchronized MIDI are most of the time not noticeable. As this system is reported to detect almost all the notes within a 300 ms window around their actual position, we set the threshold θ_t to this value. The frequency tolerance threshold θ_f is heuristically set to 3 semitones.

	Title	Original Artist
1	A Day in the Life	The Beatles
2	Genie in a Bottle	Christina Aguilera
3	I Heard it Through the Grapevine	Marvin Gaye
4	Is This Love	Bob Marley
5	Long Train Running	The Doobie Brothers
6	Sgt Pepper's Lonely Hearts Club Band	The Beatles
7	She's Leaving Home	The Beatles
8	Stop Me If You Think You've Heard This One Before	The Smiths
9	With a Little Help From My Friends	The Beatles

Table 1. List of the songs in the database.

In the experiments, we compare the separation obtained with both systems described in Section 3 with the baseline system of Section 2. We also introduce an additional method, which exploits the temporal information of the aligned score for a post-processing of the baseline system. In this approach, a 'temporal mask' is applied on the leading voice estimate: when the voice is considered as absent (in the sense of Subsection 3.2), the corresponding signal frames are shifted to the accompaniment estimate.

The separation quality is measured by the criteria described in [3]. They comprise three signal-based and four perceptual metrics, namely the Signal-to-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR), Signal-to-Artifacts Ratio (SAR), Overall Perceptual Score (OPS) and Target-, Interference- and Artifacts-related Perceptual Scores (TPS, IPS and APS respectively).

4.2 Results

The results of the evaluations, averaged over the nine pieces, are compiled in Table 2. One can first notice that the system exploiting the time-frequency constraint obtains the best results according to almost all the measures used. In particular, the average OPS of the leading voice estimates improves from 21.5 to 32.5 and the average SDR increases by 1.5 dB. This indicates that the proposed approach does improve the leading voice separation quality, since both the interferences and artifacts are reduced compared to the baseline system.

The use of the temporal indications of the score, which indicate when the leading voice is active, results in an improvement of the quality of both leading voice and accompaniment estimates. As expected, the interferences of the accompaniment in the leading voice estimates are greatly reduced with the 'temporal mask' post-processing of the baseline system. Hence, the average SIR increases from 8.4 dB to 11.0 dB. Moreover, the constraints detailed in Subsection 3.2, which forces the voice to be active only where the main melody is actually present, leads to a further improvement for most of the songs. The use of these constraints results in a more precise estimation of the spectral components of the NMF and, as a consequence, in a reduction of the artifacts. For instance, the average APS on the accompaniment parts increases from 59.0 to 63.8 with

	SDR (dB)		SIR (dB)		SAR (dB)		OPS		TPS		IPS		APS	
	LV	Ac	LV	Ac	LV	Ac	LV	Ac	LV	Ac	LV	Ac	LV	Ac
Baseline	5.8	9.1	8.4	12.4	15.2	19.3	21.5	37.0	39.5	62.9	50.8	55.6	31.4	50.6
Baseline + Temporal Mask	6.7	10.0	11.0	12.9	15.8	20.5	29.5	**43.6**	41.4	68.1	58.3	57.6	35.1	59.0
Time Constraint	7.0	10.3	11.5	13.3	16.1	20.8	31.6	43.3	45.4	67.9	**58.9**	57.5	37.4	62.6
Time-Frequency Constraint	**7.3**	**10.5**	**11.9**	**13.7**	**16.9**	**21.5**	**32.5**	42.9	**46.4**	**68.3**	57.9	**58.1**	**39.9**	**63.8**

Table 2. Average evaluation criteria, measured on the leading voice (LV) and accompaniment (Ac) parts. In boldface are the best value of each column.

time and frequency constraints.

A more precise representation of the SDR values for every tested song is displayed in Figure 1. This figure confirms that the use of the the information conveyed in the musical score is valuable. Indeed, in terms of SDR, the baseline system is outperformed by all the other approaches. An observation which can seem surprising is that in many of the pieces, the addition of the frequency constraint does not improve the SDR measure. This is explained by the efficiency of the tracking algorithm used for the determination of the fundamental frequency of the leading voice. Hence, when the leading voice is strongly dominant in the recording, this tracking does not need to be constrained. On the other hand, the constraint has a visible effect on recording no. 4: *Is This Love*. Indeed, this song contains background vocals which can incidentally be tracked as main voice, when the lead singer is not dominant (for example in the case of breaths). Hence, the global average SDR slightly increase from 8.7 dB to 8.9 dB.

The OPS criterion measured on the database is displayed in Figure 2. In general, this metric exhibits the same tendencies as the SDR. However, there are some noticeable differences concerning the accompaniment estimates. Indeed, for the first three songs, the best OPS is obtained with the original system. A more specific analysis reveals that these correspond to cases where the score does not perfectly match the performance.

One of the main sources of deviation is the length of the notes in the MIDI score. Indeed, whereas the note onsets can be relatively well defined, determining the offsets is a notoriously hard problem, which can even be ill-posed. The score often indicates how the notes are to be *played*, which can actually be different from how the notes are *heard* in the recording, mainly because of the reverberation phenomenon (which is often increased by artificial effects). This phenomenon is strongest in song no. 1 *A Day in the Life*. In this piece, with the proposed constraints, the voice is 'cut' at the end of some musical phrases, because it is considered as absent while it can still be heard in the recording. This phenomenon is not prominent from the 'signal' point of view: indeed, the SIR criteria measured on the accompaniment estimates of this piece are 16.3 dB with the time-frequency constraint and 15.6 dB with the baseline system. However, this results in intermittent 'bursts' of voice in the accompaniment part, which is more strongly penalized by the perceptual measures. Hence, the value of the IPS degrades from 60.4 to 50.9.

In the songs no. 2 *Genie in a Bottle* and no. 3 *I Heard it Through the Grapevine*, this note length problem is also visible. Besides, the lead singer sometimes adds 'ornaments' to the transcribed score, in particular through 'vocalises', which are common in the *soul music* style. Hence, both time and frequency priors indicated in the MIDI file can be misleading at some point. As previously, this does not have a large influence on the signal-based measures, since the SIR of the accompaniment estimate only decreases from 13.3 dB to 12.6 dB. However, the perceptual importance of these separation errors is greater: the OPS drops from 45.3 to 34.1.

4.3 Constrained Second Pass

In order to reduce the problem caused by the reverberation of the leading voice, we experimented with the use of a second pass of the separation algorithm. Indeed, the reverberation often introduces 'polyphony', in the sense that several notes of the leading voice can be present at the same time in the recording. Since the separation model is inherently monophonic, because it is motivated by the physics of voice production, a multi-pass approach is needed for the handling of several simultaneous notes.

Hence, after the first separation with the time-frequency constraint, we apply the same algorithm on the accompaniment estimate, where some reverberated leading voice is supposed to remain. In this second pass however, the time tolerance for the offset is modified, so that each note is allowed to be active for 800 ms after its annotated extinction in the synchronized MIDI score. The threshold for the onset time is unchanged. The estimated reverberated voice is then added to the voice estimate of the first separation.

Figure 3 displays the influence of this approach on the OPS criteria. While it has little effect on the OPS of the leading voice estimate, it can recover from some of the previously described problems of the accompaniment. Indeed, on six of the nine tested pieces, the two-pass separation visibly increases the OPS. Furthermore, the use of this approach leads to an improvement on every piece compared to the baseline system, except for song no. 9 *With a Little Help From My Friends*, where the score is equivalent.

More thorough analysis reveals that the second pass actually slightly degrades the leading voice estimates, according to many evaluation metrics. In particular, it adds some interference in the vocal track, since in many places, the reverberated voice is dominated by the accompaniment. Thus, the average SIR decreases from 11.9 dB to

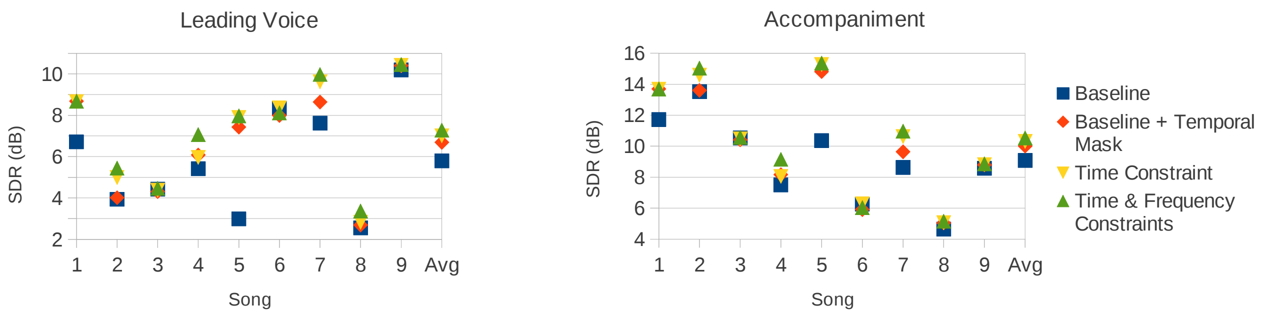


Figure 1. Signal to Distortion Ratio (SDR) measured on each of the tested songs and average.

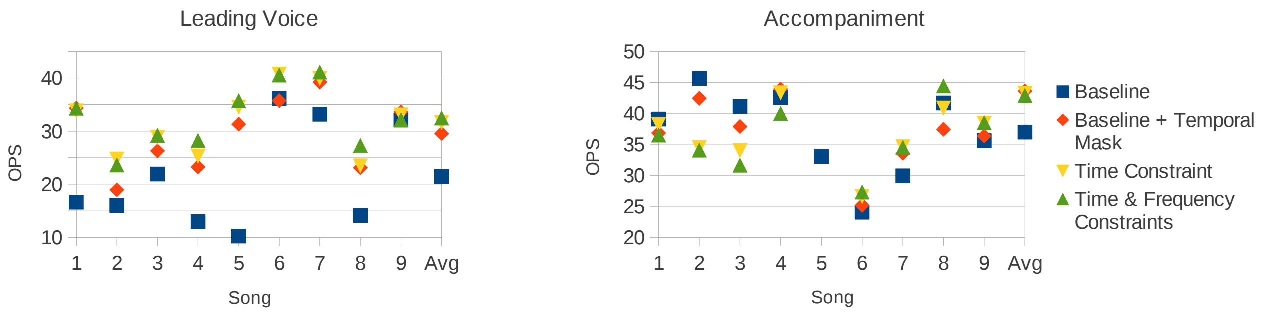


Figure 2. Overall Perceptual Score (OPS) measured on each of the tested songs and average. For the accompaniment of song no. 5, the extraction is nearly perfect, since all proposed methods obtain an OPS of 99 (not represented here).

9.6 dB. However, the artifacts are somewhat reduced and the voice seems to be better preserved. Hence, the average TPS improves from 46.4 to 53.7.

The opposite effect is observed on the accompaniment estimates, since more artifacts are measured. Hence, the average APS decreases from 63.8 to 55.3. However, these artifacts, which are very limited in terms of signal energy (the average SAR is 21.2 dB), are counterbalanced by the reduction of the interferences: the average SIR increases from 13.7 dB to 15.0 dB.

5. CONCLUSION

In this work, we exploited of a time-aligned MIDI file to perform a score-informed separation of the leading voice from a musical recording. The source-filter model assumed for the leading voice allowed for a natural use of the score information, by means of time and frequency constraints on the source components. We evaluated the usefulness of these constraints on a database of real recordings of popular songs and corresponding MIDI scores.

The results show that the score-guided constraints applied to the model not only reduce the interferences of the accompaniment in the leading voice separated track, but also allow for a more accurate estimation of the spectral shapes of all the components. Hence, this results in a reduction of the separation artifacts on both leading voice and accompaniment estimates. These improvement can be measured with perceptual metrics as well as signal energy-based criteria. Furthermore, a two-pass approach is made possible by the time-frequency constraints on the voice components. This allows for the removal of the remaining reverberated voice in the accompaniment estimate, while limiting the artifacts introduced when the voice has been correctly eliminated.

However, some problems are observed when the score does not exactly match the performance.In these cases, the score-based constraints can prevent the system from estimating the right components. Although these problems do not generally represent much in terms of signal energy, they can have some perceptual importance. Thus, future work for the improvement of the separation could involve musically-motivated modifications of the constraints, for example allowing more frequency deviation in the beginning and at the end of the notes, in order to account for *glissandi*. One could also investigate a ‘soft constraint’ approach, which would penalize source activations which are far from the score indication, without completely forbidding them. The influence of the separation parameters (number of components for the accompaniment, size of the filter dictionary) could also be more thoroughly investigated. In particular, the search for a relation between the optimal parameters and some features extracted from the musical score could be interesting. Finally, another perspective can be the exploitation of the score information for the extraction of the unvoiced components of the leading voice, which were not taken into account in this work.

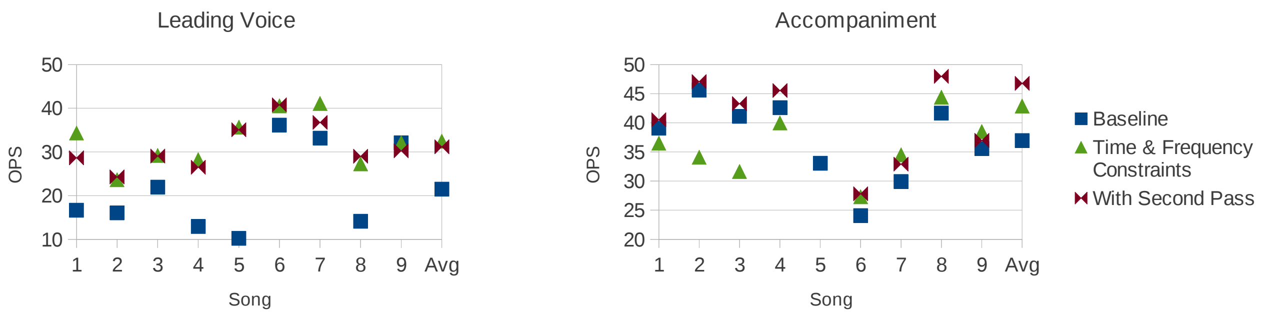


Figure 3. Influence of the second pass of leading voice separation on the OPS criterion. The same remark as in Figure 2 holds for song no. 5.

6. REFERENCES

[1] Zhiyao Duan and Bryan Pardo. Soundprism: An online system for score-informed source separation of music audio. *IEEE J. Select. Topics Signal Processing*, 5(6):1205–1215, October 2011.

[2] Jean-Louis Durrieu, Bertrand David, and Gaël Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *IEEE J. Select. Topics Signal Processing*, 5(6):1180 –1191, October 2011.

[3] Valentin Emiya, Emmanuel Vincent, Niklas Harlander, and Volker Hohmann. Subjective and objective quality assessment of audio source separation. *IEEE Trans. Audio, Speech, Language Processing*, 19(7):2046 – 2057, September 2011.

[4] Sebastian Ewert and Meinard Müller. Using score-informed constraints for NMF-based source separation. In *Proc. IEEE ICASSP*, Kyoto, Japan, 2012.

[5] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proc. IEEE ICASSP*, pages 1869–1872, Taipei, Taiwan, 2009.

[6] Joachim Ganseman, Paul Scheuners, Gautham J. Mysore, and Jonathan S. Abel. Evaluation of a score-informed source separation system. In *Proc. ISMIR*, pages 219–224, Utrecht, Netherlands, August 2010.

[7] Yushen Han and Christopher Raphael. Desoloing monaural audio using mixture models. In *Proc. ISMIR*, pages 145–148, Vienna, Austria, 2007.

[8] Toni Heittola, Anssi Klapuri, and Tuomas Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. In *Proc. ISMIR*, pages 327–332, Kobe, Japan, 2009.

[9] Romain Hennequin, Bertrand David, and Roland Badeau. Score informed audio source separation using a parametric model of non-negative spectrogram. In *Proc. IEEE ICASSP*, pages 45–48, Prag, Czech Republic, 2011.

[10] Katrsutoshi Itoyama, Masataka Goto, Kazumori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Integration and adaptation of harmonic and inharmonic models for separating polyphonic musical signals. In *Proc. IEEE ICASSP*, pages 57–60, Honolulu, Hawaii, USA, 2007.

[11] Cyril Joder, Slim Essid, and Gaël Richard. A conditional random field framework for robust and scalable audio-to-score matching. *IEEE Trans. Audio, Speech, Language Processing*, 19(8):2385–2397, November 2011.

[12] Yipeng Li, John Woodruff, and DeLiang Wang. Monaural musical sound separation based on pitch and common amplitude modulation. *IEEE Trans. Audio, Speech, Language Processing*, 17(7):1361–1371, September 2009.

[13] Bernhard Niedermayer and Gerhard Widmer. A multi-pass algorithm for accurate audio-to-score alignment. In *Proc. ISMIR*, pages 417–422, Utrecht, the Netherlands, 2010.

[14] Alexey Ozerov, Pierrick Philippe, Frédéric Bimbot, and Rémi Gribonval. Adaptation of bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *IEEE Trans. Audio, Speech, Language Processing*, 15(5):1564–1578, July 2007.

[15] Alexey Ozerov, Emmanuel Vincent, and Frédéric Bimbot. A general flexible framework for the handling of prior information in audio source separation. *IEEE Trans. Audio, Speech, Language Processing*, 20(4):1118–1133, May 2012.

[16] Emmanuel Vincent. Musical source separation using time-frequency source priors. *IEEE Trans. Audio, Speech, Language Processing*, 14(1):91–98, January 2006.

ON MEASURING SYNCOPATION TO DRIVE AN INTERACTIVE MUSIC SYSTEM

George Sioros
Faculdade de Engenharia da Universidade do Porto
gsioros@gmail.com

André Holzapfel
Music Technology Group, Universitat Pompeu Fabra
hannover@csd.uoc.gr

Carlos Guedes
Faculdade de Engenharia da Universidade do Porto / INESC Porto
cguedes@fe.up.pt

ABSTRACT

In this paper we address the problem of measuring syncopation in order to mediate a musically meaningful interaction between a live music performance and an automatically generated rhythm. To this end we present a simple, yet effective interactive music system we developed. We shed some light on the complex nature of syncopation by looking into MIDI data from drum loops and whole songs. We conclude that segregation into individual rhythmic layers is necessary in order to measure the syncopation of a music ensemble. This implies that measuring syncopation on polyphonic audio signals is not yet tractable using the current state-of-the-art in audio analysis.

1. INTRODUCTION

Rhythmic syncopation is an essential notion both in analyzing and characterizing music as well as in automatically generating musically interesting rhythmic performances. It is commonly related to rhythmic complexity and tension. Several operational and formal definitions of syncopation have been given (see [1], [2]), such as the one found in the *New Harvard Dictionary of Music* which describes syncopation as a temporary contradiction to the prevailing meter.

Various syncopation metrics have been reported (see [3], [4]), however, a reliable computational model that can measure syncopation directly on an actual music performance does not yet exist. Most metrics use binary representations as input and disregard information contained in the amplitudes of events. However, music performances are usually captured as audio signals or MIDI events, and in both cases the amplitudes of events play an important role in rhythm perception. A new syncopation measure was recently reported by Sioros and Guedes [5] (referred to as SG henceforth) that considers the amplitude of events can be applied to obtain a more detailed representation of rhythm. This kind of representation is closer to an actual music signal; it resembles a monophonic real time stream of MIDI events.

We aim to develop a system that uses syncopation to mediate the interaction between a musician performing live and an automatic rhythm generator. To this end, we explore the difficulties in measuring syncopation in a live music performance. The current study focuses on measuring syncopation in MIDI streams, from which we draw conclusions on how to measure syncopation in audio signals.

We examined the difficulties in measuring syncopation on rhythmic patterns derived from multichannel, multitimbre MIDI streams by analyzing two datasets, one comprised of short drum-loops and the second of whole songs in various genres. We used the Longuet-Higgins and Lee's metric [6] (referred to as LHL) as it is well-known and shows good agreement with human judgments ([3], [7]). We conclude that the segregation of the instruments in the performance is needed to obtain meaningful syncopation measurements. A comparison between the SG and the LHL metrics was performed, which shows agreement between the two measures, and deviations that can be attributed to processing the amplitude information in the SG metric.

Finally, we developed a software system that maps real time syncopation measurements to aspects of a rhythmic performance automatically generated by the kin.rhythmicator software [8]. The measurements are performed on either audio or MIDI inputs, as long as they are the result of a single instrument. The system serves as a tool for exploring, designing and creating interactive music performances.

In Section 2, we describe the two syncopation metrics used in the current study. In Section 3, a small study on syncopation follows, where the two MIDI datasets are examined and a comparison between the two metrics is made. In Section 4, we describe the interactive music system we have developed.

2. SYNCOPATION MEASUREMENTS

2.1 Binary representations of rhythms.

The computation of syncopation using the LHL algorithm [6] is based on a hierarchical metrical structure constructed by stratifying the given meter into metrical layers. The structure can be represented as a tree diagram with the whole bar at the top and the lower metrical levels under it. The exact form of the tree depends on the time signature of the meter. For a 4/4 meter, the bar is subdivided first into half notes, then quarter notes, eighth notes etc. until a level is reached which represents the shortest note value to be considered. In this way the meter is subdivided into pulses that belong to different metrical levels,. In the LHL measure, each pulse is assigned a metrical weight according to the metrical level it belongs to, starting with 0 for the whole bar level, −1 for the half note, −2 for the quarter note, and for each following level the weights are further lowered by 1. While in many studies, *e.g.*[7], the lowest level chosen is the 8th note, for applications involving actual music data at least the 16th note level is necessary. In the current study, all syncopation measurements are done on rhythmic patterns in 4/4 stratified to the 16th note level, however, in the examples given in this section we only show metrical levels down to the 8th note level for visual clarity.

Given a monophonic note sequence, we can compute a syncopation score based on the weights of the metrical positions of the notes and the rests. Note durations are ignored. A score is assigned to each rest that is the difference between the weight for that pulse minus the weight for the preceding note event. Summing all scores yields the syncopation value for the sequence. Some examples are given in Table 1. We placed the syncopation scores assigned to the rest-note combinations at the metrical positions of the rests. The depicted sequences are considered to wrap around as loops. In Example 2 we get a syncopation score of $(0-(-3))+(-2-(-3))+(-2-(-3))=3+1+1=5$.

A closer look at these examples reveals, in certain cases, that the results of the algorithm contradict our musical intuition. Example 1 receives a value of 0 since it contains no rests. Example 3, however, also receives a syncopation of 0, against our experience that it is more syncopated than Example 1. This arises because negative scores compensate positive scores: (-3-0) + (-1- (- 3)) + (- 2- (- 3)) = 0. We note that summing only positive scores in Example 3 would yield a positive syncopation value. The negative values computed by the LHL algorithm negatively correlate with what could be referred to as metrical strength: while the sequence of 8th notes in Example 1 has a neutral score, Example 4 supports the beat more strongly, as indicated by the larger negative values. However, since we are mainly interested in reliably detecting the syncopation of a bar, we can sum only the positive scores (last column) in Table 1.

In the present study, the results of the algorithm are normalized by the maximum possible score—one less than the total number of pulses in a bar—in order for the results to be independent of the number of pulses in the bar and the lowest stratification level chosen. The normalized syncopation will be referred to as NLHL-p.

Since the LHL algorithm was proposed for individual monophonic sequences, we need a method to compute syncopation when several layers of events take place simultaneously. This is the case of multiple instruments in a music ensemble, where a different rhythmic pattern might be performed on each instrument. The overall syncopation of the performance depends on how syncopated each one of the patterns is. We will explain the applied method by examining the four rhythmic patterns of Table 1 as if they were the four parts of a quartet. To obtain an overall syncopation measure for the polyphonic sequence we combine the maximum values for each metrical position and sum them. This results in a syncopation value of 7/7=1 for our example (last row of Table 1). Note that this polyphonic syncopation value can exceed the value of 1, which follows our musical intuition that the syncopation of a combination of instruments can be higher than their individual maximum values. This polyphonic syncopation will be referred to as POLYSYNC in the following sections.

2.2 Sequences of amplitude values.

We now provide an overview of the syncopation measure proposed by Sioros & Guedes in [5] (SG). This algorithm can be applied to a more complex representation of rhythmic patterns which, in addition to the metrical positions of the events, also includes their amplitudes. We also discuss the advantages of this kind of representation over a binary one with regard to measuring syncopation.

As in the case of the LHL algorithm described above, the SG syncopation measure is also based on a hierarchical model of musical meter. It compares a sequence of amplitude values to a metrical template similar to the one described in [9]. The algorithm is performed in two phas-

	pulses / sequences								**weights / scores**								**syncopation**	
4/4 in ♪	1	2	3	4	5	6	7	8	0	-3	-2	-3	-1	-3	-2	-3	LHL	LHL-p
Example 1	♪	♪	♪	♪	♪	♪	♪	♪									0	0
Example 2	𝄾	♪	𝄾	♪	♪	♪	𝄾	♪	**3**		**1**				**1**		5	5
Example 3	♪	𝄾	♪	♪	𝄾	♪	𝄾	♪		(-3)			**2**		**1**		0	3
Example 4	♪	𝄾	♪	𝄾	♪	♪	♪	♪		(-3)		(-1)					-4	0
								combined	3		1		2		1			7

Table 1: Computation of the LHL syncopation metric on 4 example sequences. LEFT: the four sequences in binary form. RIGHT, the corresponding weights and scores of the pulses. The negative scores are shown in parentheses as they are ignored in our LHL-p measure.

es. First, it tries to identify loud events that do not occur regularly on the beat in any metrical level. These isolated events contribute to the overall syncopation feel of the pattern. The second phase of the algorithm scales this contribution according to the potential of each metrical position to produce syncopation. The algorithm is performed in five steps (Figure 1). The first phase includes steps 1 to 3 and the second phase, steps 4 and 5. We will demonstrate the algorithm by calculating step by step the syncopation of pulse 5 in Figure 1.

In the first phase, the events that occur regularly on some metrical level are eliminated. Step 1 consists of determining the metrical levels each pulse belongs to, according to the time signature of the meter. In the example of Figure 1, pulse 5 belongs to the half note metrical level (level 1), as well as to all lower ones, i.e. the quarter (2) and eighth (3) note levels. In step 2 the amplitude differences are taken from the neighboring events and are averaged in pairs for each metrical level. The corresponding amplitude differences and averages for pulse 5 would be: i) at the half note metrical level, pulses 5 – 1 in the current bar and 5 – 1 in the next bar; ii) at the quarter note level, pulses 5 – 3 and 5 – 7; and iii) at the eighth note level, pulses 5 – 4 and 5 – 6. In step 3, the lowest value of the calculated averages is taken as the syncopation score of the pulse. If the amplitudes of the events in pulses 1, 5 and 7 of the example are 0.5, 1.0 and 0.5, then the three averages are 0.75, 0.75 and 1. Taking the minimum, the syncopation score of pulse 5 is 0.75.

The second phase of the algorithm (steps 4 and 5) is needed to account for the fact that not all the metrical positions have equal potentials to contradict the prevailing meter: the higher the metrical level the lower its syncopation potential. In step 4 the syncopation potentials are calculated for each pulse as $1\text{-}0.5^{m}$, where m is the highest metrical level the pulse belongs to. In step 5, the syncopation score for each pulse is multiplied by the corresponding syncopation potential. For pulse 5 of the example m = 1 and the final syncopation is 0.75 x $(1\text{-}0.5^{1})$ = 0.375. The final result is calculated as the sum of the syncopation of the individual events and is further normalized to the maximum possible syncopation for the same number of events in the bar. This maximum corresponds to a pattern where all events are placed at the lowest metrical level and with amplitude equal to 100%.

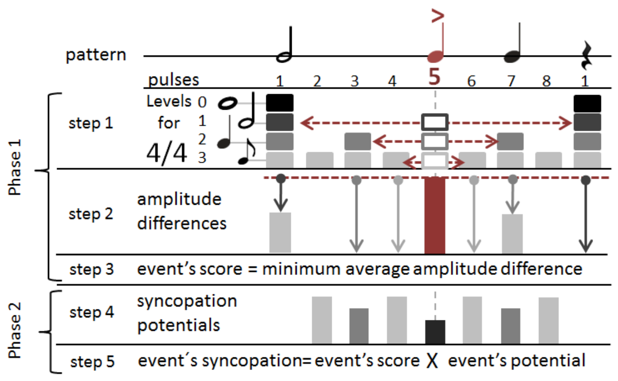


Figure 1: Example of the SG algorithm. Calculation of the syncopation of the 2nd half note in a 4/4 meter (pulse 5).

The two syncopation measures used in this article have an important difference. The SG algorithm is applied to a more detailed representation of the rhythmic patterns that includes the amplitudes of the events. This makes it possible for the SG algorithm to measure syncopation in drum rolls or arpeggios where events are present in all metrical positions and the syncopation arises from accents in offbeat positions.

3. A SMALL STUDY ON SYNCOPATION

3.1 Methodology

We applied the NLHL-p and the SG algorithm to two different kinds of MIDI datasets. The MIDI data was imported and quantized to the 16th note metrical grid. Syncopation measurements using the SG algorithm were obtained from sequences of amplitudes derived by the MIDI note velocities. When more than one note event was found at the same metrical position, the one with highest velocity was kept. The first dataset, which will be referred to as the Loops-dataset, consists of 602 drum loops from the following genres: Rock, Funk and Disco/Dance. The second dataset, which will be referred to as RWC36, consists of the first 36 songs from the RWC Music Genre[1] dataset that belong to genres of Western popular music. In contrast to the Loops-dataset, the RWC dataset contains whole songs, with each instrument part found in a different MIDI track. All loops and songs examined in here belong to the 4/4 meter, as comparing results between rhythms in different meters is a difficult and unexamined topic which is outside of the scope of this paper.

We used the algorithms to analyze the differences in the musical genres and instruments in terms of syncopation. This analysis reveals which of the examined genres make most use of rhythmic syncopation, as well as how this syncopation is distributed among the various instruments and sections of the songs. It serves as a first towards understanding how syncopation could be measured in audio signals. It must be noted that the results cannot be evaluated against a ground-truth, as there is no syncopation ground-truth available for any music dataset. Instead, we verified that the results are consistent with what is known about and expected for the syncopation in Western popular music. The same qualitative results were observed for both algorithms, so we restrict the representation of the results in sections 3.2 and 3.3 to the NLHL-p algorithm. In section 3.4 we make a more detailed comparison of the two measures.

[1] http://staff.aist.go.jp/m.goto/RWC-MDB/

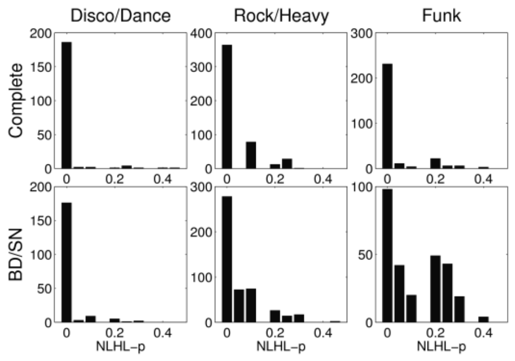


Figure 2: Histograms of the number of bars each syncopation score was calculated by the NLHL-p algorithm, for the three most frequent styles in the Loops-dataset. Upper row: for the complete drum-set, lower row: only for the bass-drum and snare-drum events.

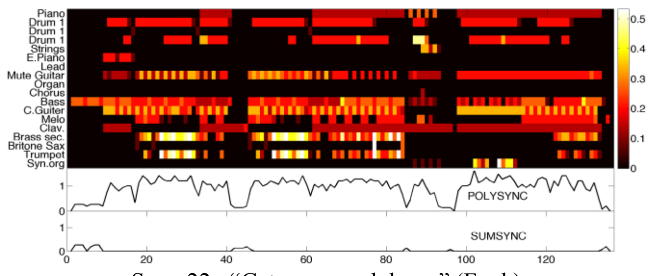
3.2 Loops-dataset

The Loops-dataset contains only short MIDI drum-loops of a few bars that use only the general MIDI drums set sounds. We measured the syncopation in every bar found in the MIDI files of the Loops-dataset, by applying NLHL-p algorithm to each bar separately, as if it constituted an independent loop. We were able to obtain a large number of syncopation measurements for three musical styles: Dance/Disco (198 bars), Funk (286 bars) and Rock/Heavy (484 bars). The histograms of the measured syncopation values are depicted in Figure 2. In the upper parts of the figure, the measurements were performed on the complete group of the general MIDI sounds, in effect ignoring MIDI note numbers. In this case, the Disco/Dance genre appears to be almost totally unsyncopated. While Rock and Funk appear slightly more syncopated, they still seem to contradict our expectations for higher syncopation. If we examine the rhythmic patterns of the bass-drum/snare-drum pair separately, ignoring all other drum sounds, we get more meaningful results as shown in the lower part of Figure 2. These histograms show an increasing percentage of syncopated bars from Disco/Dance to Rock/Heavy to Funk, as expected from these styles. This is a first indication towards the more general conclusion of this study that syncopation needs to be measured in the individual rhythmic patterns that comprise a musical performance, implying that at least a basic source/instrument separation is necessary.

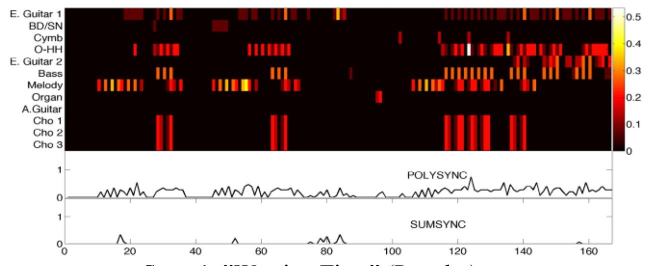
3.3 RWC-dataset

In contrast to the Loops-dataset, the RWC-dataset contains complete songs, with several instruments, each in its own MIDI track. We computed the NLHL-p syncopation measure for each track separately and combined the most syncopated events to compute the overall syncopation of the ensemble, using the POLYSYNC method described in Section 2.1. The drum tracks were separated into the following groups: Bass-Drum/Snare, Cymbals, and Open-hihat. Such a separation was found to be appropriate from the analysis of the loops dataset. We also applied the same syncopation algorithm to the complete ensemble, considering all note events regardless of the tracks or instruments (SUMSYNC method). The results for two representative songs of the collection are shown Figure 3. The two methods clearly give different syncopation results. They only coincide when a single instrument is syncopating while the rest are silent or when all instrument play in unison. Computing the syncopation on the whole ensemble fails to capture the actual syncopation in the song, and only when we combined the syncopation measurements for each individual instrument the results reflected the actual performance, as can be seen from the POLYSYNC and SUMSYNC curves. Additionally, a much stronger syncopation is encountered in the Funk song, and with a wider distribution among the instruments and among the different sections of the song, as seen in the syncopation matrices of Figure 3.

The above conclusions are not limited to the two depicted examples but are quite common for all 36 songs of the collection. In fact, in less than 2% of the total bars that have syncopated events in some MIDI track, the two methods, the POLYSYNC and SUMSYNC, agree with each other. In contrast, almost 90% of the examined bars show detectable syncopation when using the POLYSYNC method. The syncopation measured in the rhythmic patterns derived from the complete ensemble shows little to no syncopation and only when combining information of the various individual instruments can we get a realistic picture which agrees with our experience. This implies



Song 22 : “Get on up and dance” (Funk)



Song 1: ”Wasting Time” (Popular)

Figure 3: Syncopation scores for two songs of the RWC collection. **Top**: individual instruments. **Bottom**: overall syncopation for the whole ensemble (SUMSYNC) and as the combination of the scores of the individual instruments (POLYSYNC).

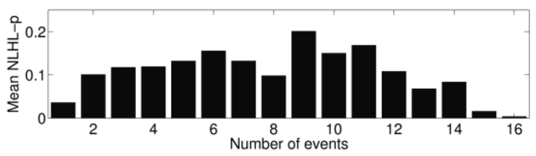


Figure 4: Mean syncopation vs. density of events per bar.

that detection of syncopation in audio signals is only possible after at least some basic instrument segregation.

Figure 4 shows how the measured syncopation is related to the density of events per metrical cycle. As expected for very high density values, the measured syncopation is close to zero, as all metrical positions are occupied by an event. Lower than average syncopation values are also obtained when only one event exists in a whole bar. Interestingly, a low mean NLHL-p value appears for bars with eight events. This is related to the fact that we only analyzed music in 4/4 where the most typical pattern with eight events would be a sequence of 8th notes that merely tend to keep the beat and therefore have no syncopation. Again, if we would consider the amplitudes of the events the average syncopation might increase.

Some conclusions about the different music genres in the RWC collection and their use of syncopation can also be made. They cannot, however, be generalized as the number of examined songs was very small. Rap and Funk songs are characterized by the highest syncopation values. In Rap, syncopation is mainly encountered in the vocals, whereas in Funk it is always spread among several instruments. Notably, the Modern Jazz pieces were not characterized by high mean values, with the lead instrument in the trios always being more syncopated than the accompaniment.

3.4 Comparing NLHL-p and SG measure

We will now compare the two measures by considering the NLHL-p as ground truth, using all separate MIDI tracks of the RWC data. Bars were marked as syncopated when the NLHL-p measure showed syncopation non-zero values. Then we examined how well the SG measure detected those syncopated bars by applying a threshold *d* to the SG measurements, above which the bars were considered to be syncopated. The comparison was made in terms of F-measure, Precision and Recall (Figure 5).

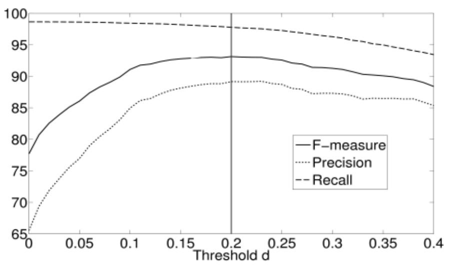


Figure 5: F-measure, Precision, and Recall for the detected syncopated bars by the SG algorithm with respect to the NLHL-p.

The comparison of the two measures shows a good agreement between them in detecting syncopation. The optimum threshold according to the F-measure is *d*=0.2 (F-measure=93.11%). The two measures exhibit a different behavior at low threshold values, where the Precision (*i.e.* the ratio between number of correct detections and number of all detections) is lower. This is caused by the fact that the SG algorithm results in an almost continuous syncopation measurement that can distinguish between rhythmic patterns based on small differences in the amplitudes of events. In contrast, the LHL measure gives a syncopation ranking of 16 steps, as it depends only on the existence or not of an event in each of the 16 pulses of a bar.

In principle, it is possible to use both algorithms, the LHL and the SG, for measuring the syncopation in a music performance in real time. As shown here, both result in similar syncopation values for most cases, yet, the SG algorithm seems to be advantageous when syncopation originates from accenting certain notes in a sequence, e.g. in drum rolls. Thus, we chose the SG algorithm to develop our system that generates rhythms based on real-time syncopation measurements of user performances.

4. A SYNCOPATION DRIVEN INTERACTIVE MUSIC SYSTEM

We developed an interactive music system based on real-time syncopation measurements. The system comprises four Max4Live devices—Max/MSP based applications that have the form of plugins for the Ableton Live sequencer[1]. Two devices measure the syncopation and density of events in the input music signal, one maps those measurements to any parameter inside the Ableton Live environment and the kin.rhythmicator [8] device generates rhythmic patterns. The input music signal can either be MIDI note events directly grabbed from music instruments and MIDI clips, or it can be simple monophonic audio that is fed to the [bonk~] [10] object for onset detection. Both MIDI and audio signals should be monophonic, i.e. the result of a performance on a single instrument. Otherwise, the syncopation measurements will not reflect the syncopation of the input, as shown in the Section 3. The MIDI input or the detected onsets are converted into a sequence of amplitudes, suitable for measuring syncopation with the SG algorithm. The measurements are performed against a metrical template automatically generated according to the time signature of the Ableton Live Set. The implementation of the SG algorithm is similar to the one used in the kin.recombinator application described in [5] with the addition of the normalization described in section 2.2. In addition to the syncopation, the

[1] http://www.ableton.com; http://www.cycling74.com;

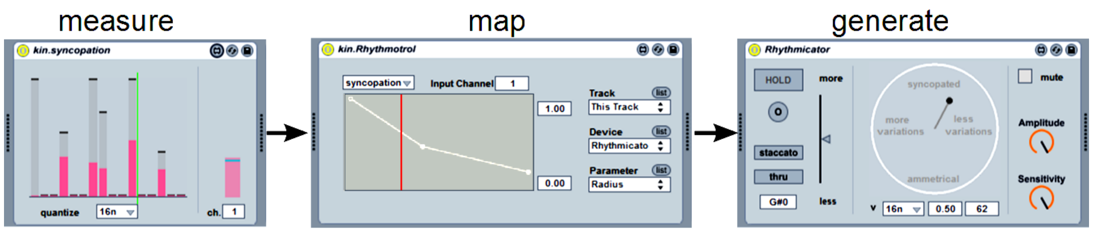


Figure 6: The Max4Live devices. The left most device receives the real time input and calculates its syncopation. The middle device receives the syncopation value and maps it to the radius parameter of the right most device, the rhythmicator. Finally, the rhythmicator generates a rhythm with the complexity being controlled by the syncopation measurements.

density of events per bar is also calculated. The measurements are then received by a second device that maps them to any parameter of any other device that the user chooses. The user also controls the exact form of the mapping.

A device like the kin.rhythmicator can be used to automatically generate rhythms. The kin.rhythmicatror features a real time control over the complexity of the generated patterns, by controlling the amount of syncopation, variation and the strength of the metrical feel. It was chosen exactly for its explicit control of the syncopation. A possible "chain" of devices is shown in Figure 6. In this way, a user can "prepare" the rhythmicator to interact in real time with a musician, e.g. as the musician performs more complex and syncopated rhythms the automatically generated patterns are more steady and simple, while when the musician tends to perform simpler and less "dense" rhythms, the generated patterns become more complex, creating a more syncopated result.

Simple to complex mappings can be realized, involving several parameters in several devices and more than one performer. The described devices are meant as a way of creating direct links between musically meaningful qualities of a performance and an automatically generated output.

The Max4Live devices are available at our website: *http://smc.inescporto.pt/kinetic/*

5. CONCLUSIONS

In this paper we presented an interactive music system driven by syncopation measurements. In order to better understand and be able to reliably measure syncopation in an actual music performance, we analyzed two MIDI datasets, one consisting of drum loops, and one of whole songs using the NLHL-p and the SG syncopation metrics. We concluded that in any musical signal, whether it is a MIDI stream or an audio signal, it is important for syncopation measurements that it is first separated into the individual rhythmic layers or the instruments that comprise it. Our findings are of particular importance for our future research that focuses in computing syncopation in more complex music signals in order to drive a meaningful interaction between a musician and a rhythm that is being automatically generated.

6. AKNOWLEDGMENTS

This work was partly supported by the Portuguese Foundation for Science and Technology, within project ref. SFRH / BPD / 51348 / 2011 and PTDC / EAT-MMU / 112255 / 2009.

7. REFERENCES

[1] D. Huron and A. Ommen, "An Empirical Study of Syncopation in American Popular Music, 1890-1939," *Music Theory Spectrum*, vol. 28, no. 2, pp. 211-231, 2006.

[2] D. Temperley, "Syncopation in rock: a perceptual perspective," *Popular Music*, vol. 18, no. 1, pp. 19-40, 1999.

[3] F. Gómez, E. Thul, and G. Toussaint, "An experimental comparison of formal measures of rhythmic syncopation," in *Proceedings of the International Computer Music Conference*, 2007, pp. 101-104.

[4] I. Shmulevich and D.-J. Povel, "Measures of temporal pattern complexity," *Journal of New Music Research*, vol. 29, no. 1, pp. 61-69, 2000.

[5] G. Sioros and C. Guedes, "Complexity Driven Recombination of MIDI Loops," in *Proceedings of the 12th International Society for Music Information Retrieval Conference*, 2011, pp. 381-386.

[6] H. C. Longuet-Higgins and C. S. Lee, "The rhythmic interpretation of monophonic music," *Music Perception*, vol. 1, no. 4, pp. 424–441, 1984.

[7] W. T. Fitch and A. J. Rosenfeld, "Perception and Production of Syncopated Rhythms," *Music Perception*, vol. 25, no. 1, pp. 43-58, 2007.

[8] G. Sioros and C. Guedes, "Automatic rhythmic performance in Max/MSP: the kin. rhythmicator," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2011, pp. 88-91.

[9] F. Lerdahl and R. Jackendoff, *A Generative Theory of Tonal Music,* Cambridge: The MIT Press, 1996.

[10] M. S. Puckette, T. Apel, and D. D. Zicarelli, "Real-time audio analysis tools for Pd and MSP analysis," in *International Computer Music Conference*, 1998, vol. 74, pp. 109-112.

CURRENT CHALLENGES IN THE EVALUATION OF PREDOMINANT MELODY EXTRACTION ALGORITHMS

Justin Salamon
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
justin.salamon@upf.edu

Julián Urbano
Department of Computer Science
University Carlos III of Madrid, Leganés, Spain
jurbano@inf.uc3m.es

ABSTRACT

In this paper we analyze the reliability of the evaluation of Audio Melody Extraction algorithms. We focus on the procedures and collections currently used as part of the annual Music Information Retrieval Evaluation eXchange (MIREX), which has become the de-facto benchmark for evaluating and comparing melody extraction algorithms. We study several factors: the duration of the audio clips, time offsets in the ground truth annotations, and the size and musical content of the collection. The results show that the clips currently used are too short to predict performance on full songs, highlighting the paramount need to use complete musical pieces. Concerning the ground truth, we show how a minor error, specifically a time offset between the annotation and the audio, can have a dramatic effect on the results, emphasizing the importance of establishing a common protocol for ground truth annotation and system output. We also show that results based on the small ADC04, MIREX05 and INDIAN08 collections are unreliable, while the MIREX09 collections are larger than necessary. This evidences the need for new and larger collections containing realistic music material, for reliable and meaningful evaluation of Audio Melody Extraction.

1. INTRODUCTION

The task of melody extraction has received growing attention from the research community in recent years [4–7, 10–12]. Also referred to as Audio Melody Extraction, Predominant Melody Extraction, Predominant Melody Estimation or Predominant Fundamental Frequency (F0) Estimation, the task involves automatically obtaining a sequence of frequency values representing the pitch of the main melodic line from the audio signal of a polyphonic piece of music. As the number of researchers working on the task grew, so did the need for proper means of evaluating and comparing the performance of different algorithms. In 2004, the first Audio Description Contest (ADC) was hosted by the Music Technology Group at Universitat Pompeu Fabra in Barcelona, Spain. This initiative later evolved into the Music Information Retrieval Evaluation eXchange (MIREX) [3], which is held annually in conjunction with the ISMIR conference.

MIREX has become the de-facto benchmark for evaluating and comparing the performance of melody extraction algorithms, with over 50 algorithms evaluated since the first run in ADC 2004. Whilst this is without doubt an indication of the formalization of the topic as an established research area, it has recently been argued that some of the evaluation procedures employed by the Music Information Retrieval (MIR) research community still lack the rigor found in other disciplines such as Text IR [13]. In this paper we examine the evaluation of melody extraction algorithms, as currently carried out in the MIREX Audio Melody Extraction (AME) task. We focus on three aspects of the evaluation: first, we examine the annotation procedure used for generating a ground truth for evaluation. Specifically, we study the influence of a systematic error in the annotations, in the form of a fixed time offset between the ground truth annotation and the output of the algorithms. This issue is particularly relevant, as such an error has actually been detected in past MIREX AME evaluations. Next, we consider the duration of the audio excerpts (clips) used for evaluation. Currently all collections used for evaluation are comprised of short excerpts taken from full songs. The underlying assumption is that performance on a short clip is a good predictor for performance on a full song. However to date this assumption has neither been confirmed nor confuted. Finally, we consider the aspect of collection size. Currently, the size of most collections used for AME evaluation is relatively small compared to collections used in other IR tasks, and so we assess whether this presents any problems or not. Through these factors, we aim to assess the reliability of the evaluation procedure, as well as the meaningfulness of the results and the conclusions that are drawn from them.

The remainder of the paper is as follows. In Section 2 we explain the current evaluation procedure for AME algorithms. Section 3 takes a closer look at the annotation procedure, assessing the potential influence of a systematic error in the annotation process. In Section 4 we study the relationship between system performance and clip duration. In Section 5 we consider the influence of the size of the music collection used for evaluation. Then, in Section 6 we provide further insight into the results obtained in the previous sections, and finally we present the conclusions in Section 7.

2. MELODY EXTRACTION EVALUATION

We start by describing the current procedure for evaluating melody extraction algorithms, as carried out in the yearly MIREX AME evaluation.

2.1 Ground Truth Annotation

The ground truth for each audio excerpt is generated using the following procedure: first, the annotator must acquire the audio track containing just the melody of the excerpt. This is done by using multitrack recordings for which the separate tracks are available. Given the melody track, the pitch of the melody is estimated using a monophonic pitch tracker with a graphical user interface such as SMSTools [1] or WaveSurfer [2], producing an estimate of the fundamental frequency (F0) of the melody in every frame. This annotation is then manually inspected and corrected in cases of octave errors (double or half frequency) or when pitch is detected in frames where the melody is not present (unvoiced frames). Finally, the estimated frequency sequence is saved into a file with two columns - the first containing the time-stamp of every frame, starting from time 0, and the second the value of the fundamental frequency in Hertz. In ADC 2004 a hop size of 5.8 ms was used for the annotation, and since 2005 a hop size of 10 ms between frames is used. Frames in which there is no melody present are labelled with 0 Hz.

2.2 Evaluation Measures

An algorithm's output for a single excerpt is evaluated by comparing it to the ground truth annotation on a frame-by-frame basis, and computing five measures which summarize its performance for the complete excerpt. For a full music collection, these five measures are computed per excerpt and then averaged over the entire collection. To facilitate the evaluation, algorithms are required to provide the output in the same format as the ground truth. The only difference between the algorithm's output and the ground truth annotation is that for frames estimated as unvoiced (i.e. no melody present) by the algorithm, the algorithm may return either 0 Hz (as in the ground truth) or a negative frequency value. The negative value represents the algorithm's pitch estimation in case its voicing estimation is wrong and the melody is actually present in that frame. This allows us to separate two different aspects in the evaluation - the algorithm's voicing estimation (determining when the melody is present and when it is not) and the algorithm's pitch estimation (determining the F0 of the melody). The five evaluation measures currently employed in MIREX, as defined in [11], are summarized in Table 1.

2.3 Music Collections

Over the years, efforts by different researchers/groups have been made to generate annotated music collections for AME evaluation. The combination of the limited amount of multi-track recordings freely available, and the time-consuming annotation process, means most of these collections are quite small compared to those used in other MIR disciplines. In Table 2 we provide a summary of the music collections used in MIREX for AME evaluation since 2009.

Voicing Recall Rate: the proportion of frames labeled as voiced in the ground truth that are estimated as voiced by the algorithm.
Voicing False Alarm Rate: the proportion of unvoiced frames in the ground truth that are estimated as voiced by the algorithm.
Raw Pitch Accuracy: the proportion of voiced frames in the ground truth for which the F0 estimated by the algorithm is within $\pm\frac{1}{4}$ tone (50 cents) of the ground truth annotation.
Raw Chroma Accuracy: same as the raw pitch accuracy, except that both the estimated and ground truth F0 sequences are mapped into a single octave, in this way ignoring octave errors in the estimation.
Overall Accuracy: combines the performance of the pitch estimation and voicing detection to give an overall performance score. Defined as the proportion of frames (out of the entire excerpt) correctly estimated by the algorithm, i.e. unvoiced frames that are labeled as unvoiced and voiced frames with a correct pitch estimate.

Table 1. AME evaluation measures used in MIREX.

Collection	Description
ADC2004	20 excerpts of roughly 20s in the genres of pop, jazz and opera. Includes real recordings, synthesized singing and audio generated from MIDI files. Total play time: 369s.
MIREX05	25 excerpts of 10-40s duration in the genres of rock, R&B, pop, jazz and solo classical piano. Includes real recordings and audio generated from MIDI files. Total play time: 686s.
INDIAN08	Four 1 minute long excerpts from north Indian classical vocal performances. There are two mixes per excerpt with differing amounts of accompaniment resulting in a total of 8 audio clips. Total play time: 501s.
MIREX09	374 Karaoke recordings of Chinese songs (i.e. recorded singing with karaoke accompaniment). Each recording is mixed at three different levels of signal-to-accompaniment ratio {-5dB, 0dB, +5dB} resulting in a total of 1,122 audio clips. Total play time: 10,022s.

Table 2. Test collections for AME evaluation in MIREX.

3. GROUND TRUTH ANNOTATION OFFSET

In this section we study the influence of a specific type of systematic error in the annotation on the results. Whilst there are other aspects of the annotation process that are also worth consideration, we find this issue to be of particular interest, since it was actually identified recently in one of the music collections used for Audio Melody Extraction evaluation in MIREX.

As explained in the previous section, all AME evaluation measures are based on a frame-by-frame comparison of the algorithm's output to the ground truth annotation. Hence, if there is a time offset between the algorithm's output and the ground truth annotation, this will cause a mismatch in all frames. Since melody pitch tends to be continuous, a very small time offset may not be noticed. However, as we increase the offset between the two sequences, we expect it to have an increasingly detrimental effect on the results.

To evaluate the effect of such an offset, we compiled a collection of 30 music clips from publicly available MIREX training sets: 10 from ADC 2004, 9 similar to MIREX05 and 11 similar to MIREX09. We used the ground truth annotations generated by the original authors of each collection, and ensured that the first frame of each annota-

[1] http://mtg.upf.edu/technologies/sms
[2] http://www.speech.kth.se/wavesurfer/

tion was centered on time 0. For evaluation, we use the output of six different melody extraction algorithms that were kindly provided by their authors: KD [4], DR[3] [5], FL [6], HJ [7], RP [9] and SG [12]. For each algorithm, we computed the mean raw pitch and overall accuracy for the entire collection, as a function of a fixed time offset introduced in the ground truth annotation, from -50 ms to 50 ms using 1 ms steps. To emulate offsets smaller than the hop size of the annotation (10 ms), the ground truth was upsampled using linear interpolation.

3.1 Results

In Figure 1 we display the results of the evaluation, where we have subtracted from all values the score at offset 0. In this way, the graph reflects the absolute difference between the score at a given offset and the optimal score of the algorithm (assuming it is centered on time 0). Plot (a) contains the results for the raw pitch measure, and plot (b) for the overall accuracy.

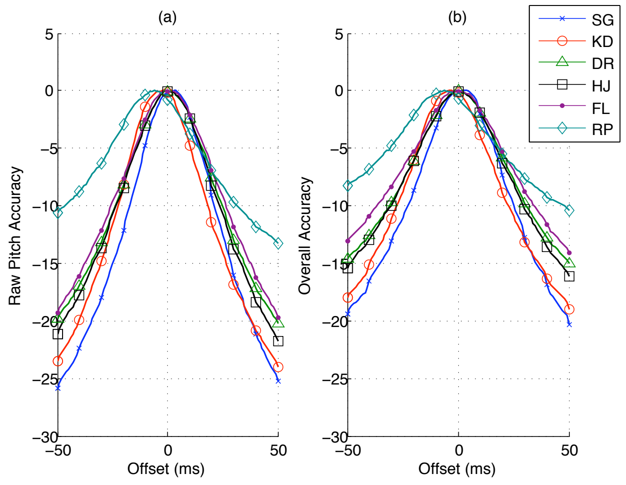


Figure 1. Absolute performance drop versus annotation offset: (a) raw pitch accuracy, (b) overall accuracy.

As can be seen, the effect of the offset is quite dramatic, causing an absolute drop of up to 25% in the raw pitch accuracy and 20% in the overall accuracy for the most extreme offset evaluated (50 ms). Though a 50 ms offset is perhaps an exaggerated case, in 2011 it was discovered that one of the MIREX collections had a 20ms offset. In our evaluation, a 20 ms offset would cause the most affected algorithms to loose 17% in raw pitch accuracy, and 13% in overall accuracy. Another interesting observation is that some algorithms do not perform best at offset 0 (most visibly RP, whose peak performance is at -6 ms). This emphasizes the fact that it does not suffice for the annotation to be centered on time 0, but rather, that there must be a strict convention to which both the annotations and algorithms adhere. Finally, we found there is a correlation between absolute performance and the effect of annotation offset: the higher the absolute performance of the algorithm, the more sensitive it is to an offset in the annotation. This is particularly important, since it suggests that the best algorithms are those who will be most affected by this type of systematic error.

[3] The output was computed using a different implementation than that of the paper, available at: https://github.com/wslihgt/separateLeadStereo

4. CLIP DURATION

A common criticism of evaluation in MIR, and particularly in MIREX, is the use of clips instead of full songs. One might argue that the use of clips is unrealistic and that observed performance on those clips may be very different from performance on full songs [13]. The collections used in the AME evaluation contain some very short excerpts, some only 10 seconds long. The use of such small clips is especially striking in AME: these clips contain primarily voiced frames, and so the generalization of the results to full songs should be questioned. We designed an experiment to assess the effect of clip duration on the reliability of the AME evaluations.

For each of the 30 clips used in the previous experiment (referred to as the *x1* clips), we created a series of subclips: 2 subclips of half the duration, 3 subclips of one third of the duration, and 4 subclips of one forth of the duration (referred to as the *x1/2*, *x1/3* and *x1/4* subclips). Note that the *x1/4* subclips can also be considered as *x1/2* versions of the *x1/2* subclips. This gives us 180 *x1/2* subclips, 90 *x1/3* subclips and 120 *x1/4* subclips, all of which were used to evaluate the six algorithms. We computed the performance difference between all subclips and their corresponding *x1* versions, leading to a grand total of 2340 data-points.

4.1 Results

In Figure 2 we show the log-scaled distribution of relative performance differences. Mean differences vary between 13% and 21% for overall accuracy and raw pitch, while for voicing false-alarm the means are around 50%. We note that there is a large amount of outliers in the distributions. However, these outliers were not found to correspond to particular songs or algorithms (they are rather randomly distributed). There seems to be a clear correlation: the shorter the subclips, the larger the performance differences (all significant by a 1-tailed Wilcoxon test, α=0.01). In principle, therefore, one would want the clips used for evaluation to be as long as possible; ideally, the full songs.

In Figure 3 we plot the log-scaled relative performance differences in overall accuracy, this time as a function of the log-scaled actual subclip duration (other measures produce very similar plots). We see that the negative correlation between subclip duration and performance difference appears to be independent of the duration of the *x1* clip. We fitted a non-linear model of the form $diff = a \cdot duration^b$, where a and b are the parameters to fit, to the results of each of the relative durations (*x1/2*, *x1/3*, *x1/4*), and as the plot shows, they are very similar. In fact, an ANCOVA analysis revealed no significant difference between them. This suggests that the error decreases as the clip duration increases, regardless of the duration of the full song.

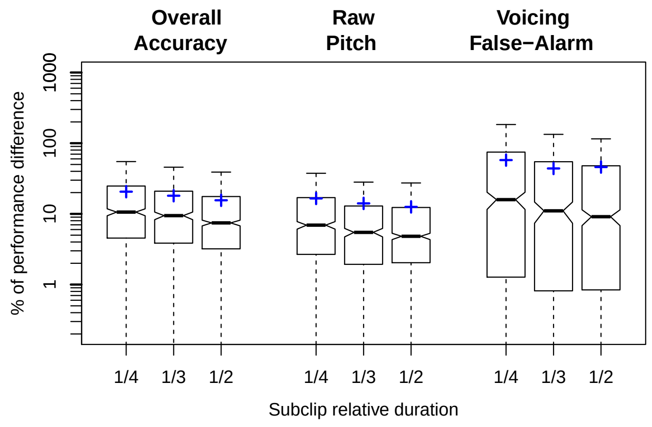


Figure 2. Releative performance differences between subclips and their corresponding *x1* clips. Blue crosses mark the means of the distributions.

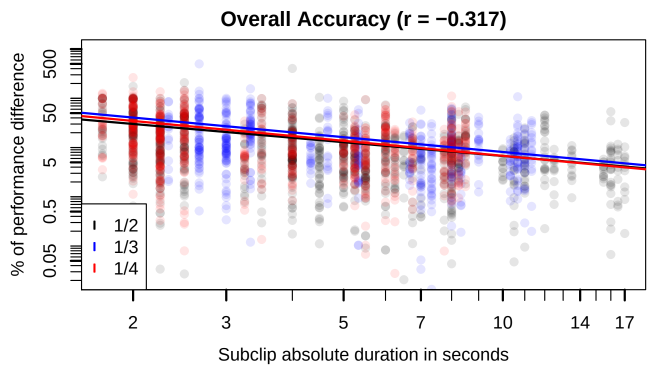


Figure 3. Relative performance differences with subclips as a function of subclip actual duration.

5. COLLECTION SIZE

Regardless of the effectiveness measure used, an AME experiment consists of evaluating a set of algorithms $\mathcal{A}$ using a set of songs $\mathcal{S}$. Such an evaluation experiment can be viewed as fitting the following model:

$$y_{as} = \overline{y} + \overline{y}_a + \overline{y}_s + \varepsilon_{as} \tag{1}$$

where y_{as} is the score of algorithm a for song s, $\overline{y}$ is the grand average score of all possible algorithms over all possible songs, $\overline{y}_a$ is the algorithm effect (the average deviation of algorithm a from the grand average $\overline{y}$), $\overline{y}_s$ is the song effect and ε_{as} is a residual modeling the particular deviation of algorithm a for song s. In our case, where we do not consider other effects such as annotators, this ε_{as} residual actually models the algorithm-song interaction effect: some algorithms are particularly better (or worse) for particular songs.

When a researcher carries out an AME evaluation experiment, they evaluate how well an algorithm performs for the set $\mathcal{S}$ of songs, but ideally they want to generalize from the performance of that specific experiment to the average score the algorithm would obtain for the population of all songs represented by the sample $\mathcal{S}$, not just the sample itself. The reliability when drawing such general conclusions based on the observations on samples (test collections) can be measured with Generalizability Theory (GT) [1, 2].

From the model in Eq. 1 we can identify two sources of variability in the observed scores: actual performance differences among algorithms and difficulty differences among songs. Ideally, we want most of the variability in y_{as} to be due to the algorithm effect, that is, the observed effectiveness differences to be due to actual differences between algorithms and not due to other sources of variability such as songs, annotators, or specific algorithm-song interactions. Note that this does not mean a collection should not contain varied musical content. Ideally, we want an algorithm to work well for all types of musical material, and hence a varied collection in terms of content does not necessarily imply large performance variability due to the song effect. However, a small collection that contains songs with a great degree of variability (in terms of difficulty) is likely to result in performance variability that is dominated by the song effect and possibly by algorithm-song interactions (e.g. algorithm X is especially good for jazz but poor for rock), thus reducing our ability to claim that the observed differences between the algorithms can be generalized to the universe of all songs. Using GT [1, 2], we can measure the proportion of observed variability that is due to actual differences between the algorithms. This proportion reflects the stability of the evaluation, and as such it is also a measure of efficiency: the higher the stability, the fewer the songs necessary to reliably evaluate algorithms [1, 8]. GT does not only help evaluate the stability of past collections, but also estimate the reliability of yet-to-be created collections as a function of their size. However, the results of GT only hold if the original data used for the analysis is *representative* of the wider population of songs to which we want to generalize in the future.

5.1 Variance Analysis and Collection Stability

In the model in Eq. 1, the grand mean $\overline{y}$ is a constant, and the other effects can be modeled as random variables with their own expectation and variance. As such, the variance of the observed scores is modeled as the sum of these variance components:

$$\sigma^2 = \sigma_a^2 + \sigma_s^2 + \sigma_{as}^2 \tag{2}$$

where σ_a^2 is the variance due to the algorithm effect, σ_s^2 is the variance due to the song effect, and σ_{as}^2 is the variance due to the algorithm-song interaction effect (the residual). This variance decomposition can be estimated by fitting a fully-crossed ANOVA model for Eq. 1:

$$\begin{gathered}\widehat{\sigma}_{as}^2 = \mathrm{E}MS_{as} = \mathrm{E}MS_{residual} \\ \widehat{\sigma}_a^2 = \frac{\mathrm{E}MS_a - \widehat{\sigma}_{as}^2}{|\mathcal{S}|} \quad , \quad \widehat{\sigma}_s^2 = \frac{\mathrm{E}MS_s - \widehat{\sigma}_{as}^2}{|\mathcal{A}|}\end{gathered} \tag{3}$$

where $\mathrm{E}MS_x$ is the expected Mean Square of component x. In practice, $\mathrm{E}MS_x$ is approximated by the Mean Square of component x as computed with the ANOVA model [1, 2]. Using the estimates in Eq. 3 we can estimate the proportion of variability due to the algorithm effect as per Eq. 2. The stability of the evaluation can then be quantified with the dependability index Φ:

	Overall Accuracy				Raw Pitch				Voicing False-Alarm			
	$\widehat{\sigma}_a^2$	$\widehat{\sigma}_s^2$	$\widehat{\sigma}_{as}^2$	$\widehat{\Phi}$	$\widehat{\sigma}_a^2$	$\widehat{\sigma}_s^2$	$\widehat{\sigma}_{as}^2$	$\widehat{\Phi}$	$\widehat{\sigma}_a^2$	$\widehat{\sigma}_s^2$	$\widehat{\sigma}_{as}^2$	$\widehat{\Phi}$
ADC04	27%	27%	46%	.879	23%	28%	49%	.859	55%	21%	23%	.961
MIREX05	11%	47%	42%	.758	15%	54%	31%	.817	57%	20%	23%	.971
INDIAN08	16%	50%	34%	.600	24%	57%	19%	.721	70%	13%	16%	.950
04 + 05 + 08	16%	39%	45%	.909	16%	43%	41%	.912	56%	21%	23%	.986
MIREX09 0dB	52%	20%	28%	.998	50%	20%	31%	.997	81%	5%	14%	.999
MIREX09 -5dB	40%	23%	37%	.996	40%	24%	35%	.996	82%	5%	13%	.999
MIREX09 +5dB	58%	17%	26%	.998	48%	18%	34%	.997	83%	4%	14%	.999

Table 3. Variance components and $\widehat{\Phi}$ score for all three measures and all six collections plus the joint 04+05+08 collection.

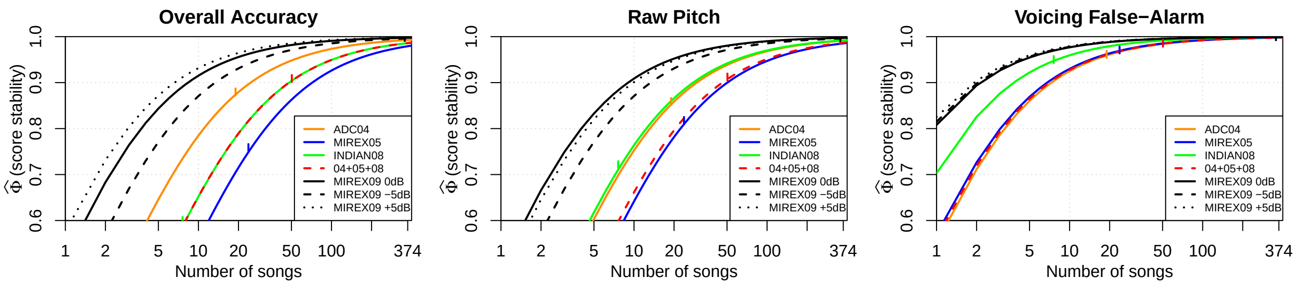


Figure 4. Dependability index as a function of the number of songs for Overall Accuracy (left), Raw Pitch (middle) and Voicing False-Alarm (right). The points mark the actual number of songs per collection.

$$\Phi = \frac{\sigma_a^2}{\sigma_a^2 + \frac{\sigma_s^2 + \sigma_{as}^2}{|S|}} \tag{4}$$

which measures the ratio between algorithm variance and the variance in absolute effectiveness scores (total variance) [1, 2]. This measure increases with the song set size (i.e. with an infinite number of songs all the observed variability would be due to algorithm differences) [8].

5.2 Results

In Table 3 we show the estimated proportion of variability due to the algorithm, song and algorithm-song interaction effects. For these calculations we used the results of the MIREX campaign directly, combining the results of the five algorithms from MIREX 2010 and ten algorithms from MIREX 2011. In both years the same six test-collections were used for evaluation, so we can consider the grouping of algorithms from both years as a single larger evaluation round leading to a fully crossed experimental design. We also joined the three smaller collections into a single larger one referred to as “04+05+08”, discussed in Section 6.

In general, it can be seen that the estimated variance due to the algorithm effect is much larger in the MIREX09 collections. For overall accuracy, the average is 50%, while for the earlier collections it is just 18%, and as low as 11% for MIREX05. These differences show that generalizations of results based on the earlier collections are not very reliable, especially in the case of the MIREX05 and INDIAN08 collections, because a large part of the variability in the scores is due to the song characteristics rather than differences between the algorithms.

Figure 4 shows the estimated dependability index as a function of the number of songs used (log scaled). The points mark the value of $\widehat{\Phi}$ for the actual number of songs in each collection (cf. Table 3). Again we observe that the MIREX09 collections are considerably more stable than the earlier collections, especially MIREX05 and INDIAN08, where $\widehat{\Phi}$ is as low as 0.6. More interesting is the fact that the dependability index in the MIREX09 collections rapidly converges to 1, and there is virtually no appreciable difference between using all 374 songs in the collection or just 100: $\widehat{\Phi}$ would only drop from an average of 0.997 to 0.990, showing that most of the variability in performance scores would still be attributable to the algorithm effect. However, we must also consider the content validity of this collection (i.e. whether it is representative or not) [13]. We discuss this in the next section.

6. DISCUSSION

Starting with the annotation offset issue, we note that there are two crucial parameters that must be fixed in order to prevent this problem: the precise time of the first frame, and the hop size. Since 2005, all the annotations use a hop size of 10 ms, and all algorithms are required to use this hop size for their output. However, the exact time of the first frame has not been explicitly agreed upon by the community. When the short-time Fourier transform (or any other transform which segments the audio signal into short frames) is used, it is common practice to consider the time-stamp of each frame to be the time exactly at the middle of the frame. Thus, for the first frame to start exactly at time zero, it must be centered on the first sample of the audio (filling the first half of the frame with zeros). Nonetheless, while this is common practice, it is not strictly imposed, meaning algorithms and annotators might, rather than center the first frame on the first sample, start the frame at this sample. In this case, the frame will not be centered on time zero, but rather on an arbitrary time which depends on the length of the frame. Since different algorithms and annotations use different frame sizes, this scenario could lead to a different fixed offset between every algorithm and every annotation, leading to a systematic error in the evaluation.

In terms of clip duration, we saw that there is a clear correlation between the relative duration of the clip (compared to the full song) and evaluation error, suggesting that performance based on clips might not really predict performance on full songs. However, Figure 3 suggests that this correlation is independent of the actual duration of the full song. That is, there might be a duration threshold of x seconds for which observed performance on clips does predict performance on full songs (within some error rate), no matter how long they are. While counter-intuitive at first, this result does somehow agree with general statistical theory. How large a sample needs to be in order to reliably estimate unknown parameters of the underlying population, is independent of how large the population actually is, as long as the sample is *representative* of the population. This usually requires to sample randomly or follow other techniques such as systematic or stratified sampling. For AME evaluation it does not make sense to randomly sample frames of a song, but the results suggest that there might be a sampling technique such that audio clips, if selected appropriately, can be representative of the full songs.

Regarding the collection size, we observed that the earlier ADC04, MIREX05 and INDIAN08 collections are unstable because a larger proportion of the variability in the observed performance scores is due to song difficulty differences rather than algorithm differences. As such, results from these collections alone are expected to be unstable, and therefore evaluations that rely solely on *one* of these collections are not very reliable. In Table 3 (and Figure 4) we see that by joining these collections into a single larger one ("04+05+08") the evaluation results are considerably more stable ($\widehat{\Phi} > 0.9$ for all three measures), and so we recommend fusing them into a single collection for future evaluations. On the other hand, we saw that the MIREX09 collections are in fact much larger than necessary: about 25% of the current songs would suffice for results to be highly stable and therefore generalize to a wider population of songs. However, all MIREX09 music material consists of Chinese karaoke songs with non-professional singers, and therefore we should expect the results to generalize to *this* population of songs, but not to the general universe of *all* songs (essentially everything that is not karaoke). Therefore, the AME community is found in the situation where the collections with sufficiently varied music material are too small to be reliable, while the ones that are reliable contain very biased music material.

7. CONCLUSION

In this paper we analyzed the reliability of the evaluation of Audio Melody Extraction algorithms, as performed in MIREX. Three main factors were studied: ground truth annotations, clip duration and collection size. We demonstrated how an offset between the ground truth and an algorithm's output can significantly degrade the results, the solution to which is the definition and adherence to a strict protocol for annotation. Next, it was shown that the clips currently used are too short to predict performance on full songs, stressing the need to use complete musical pieces. It was also shown that results based on one of the ADC04, MIREX05 or INDIAN08 collections alone are not reliable due to their small size, while the MIREX09 collection, though more reliable, does not reflect real-world musical content. The above demonstrates that whilst the MIREX AME evaluation task is an important initiative, it currently suffers from problems which require urgent attention. As a solution, we propose the creation of a new and open test collection through a joint effort of the research community. If the collection is carefully compiled and annotated, keeping in mind the issues mentioned here, it should, in theory, solve all of the aforementioned problems that current AME evaluation suffers from. Furthermore, we could consider the application of low-cost evaluation methodologies that dramatically reduce the annotation effort required [14]. Finally, in the future it would also be worth studying the appropriateness of the evaluation measures themselves, the accuracy of the manual ground truth annotations and further investigate the effect of clip duration.

8. ACKNOWLEDGMENTS

We would like to thank the authors of the melody extraction algorithms for their contribution to our experiments. This work was supported by the Programa de Formación del Profesorado Universitario (FPU) and grant TIN2010-22145-C02-02 of the Spanish Government.

9. REFERENCES

[1] D. Bodoff. Test theory for evaluating reliability of IR test collections. *Inf. Process. Manage.*, 44(3), 2008.

[2] R. L. Brennan. *Generalizability Theory*. Springer, 2001.

[3] J. Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 2008.

[4] K. Dressler. Audio melody extraction for mirex 2009. In *Music Inform. Retrieval Evaluation eXchange (MIREX)*, 2009.

[5] J.-L. Durrieu, G. Richard, B. David, and C. Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE TASLP*, 18(3), 2010.

[6] B. Fuentes, A. Liutkus, R. Badeau, and G. Richard. Probabilistic model for main melody extraction using constant-Q transform. In *IEEE ICASSP*, 2012.

[7] C. Hsu, D. Wang, and J. Jang. A trend estimation algorithm for singing pitch detection in musical recordings. In *IEEE ICASSP*, 2011.

[8] E. Kanoulas and J. Aslam. Empirical justification of the gain and discount function for nDCG. In *ACM CIKM*, 2009.

[9] R. P. Paiva. *Melody Detection in Polyphonic Audio*. PhD thesis, University of Coimbra, Portugal, 2007.

[10] R. P. Paiva, T. Mendes, and A. Cardoso. Melody detection in polyphonic musical signals: Exploiting perceptual rules, note salience, and melodic smoothness. *Comput. Music J.*, 2006.

[11] G. E. Poliner, D. P. W. Ellis, F. Ehmann, E. Gómez, S. Steich, and B. Ong. Melody transcription from music audio: Approaches and evaluation. *IEEE TASLP*, 15(4), 2007.

[12] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE TASLP*, 20(6), 2012.

[13] J. Urbano. Information retrieval meta-evaluation: Challenges and opportunities in the music domain. In *ISMIR*, 2011.

[14] J. Urbano and M. Schedl. Towards Minimal Test Collections for Evaluation of Audio Music Similarity and Retrieval. In *WWW Workshop on Advances in MIR*, 2012.

IMPROVING AUDIO CHORD TRANSCRIPTION BY EXPLOITING HARMONIC AND METRIC KNOWLEDGE

W. Bas de Haas
Utrecht University
W.B.deHaas@uu.nl

José Pedro Magalhães
University of Oxford
jpm@cs.ox.ac.uk

Frans Wiering
Utrecht University
F.Wiering@uu.nl

ABSTRACT

We present a new system for chord transcription from polyphonic musical audio that uses domain-specific knowledge about tonal harmony and metrical position to improve chord transcription performance. Low-level pulse and spectral features are extracted from an audio source using the Vamp plugin architecture. Subsequently, for each beat-synchronised chromagram we compute a list of chord candidates matching that chromagram, together with the confidence in each candidate. When one particular chord candidate matches the chromagram significantly better than all others, this chord is selected to represent the segment. However, when multiple chords match the chromagram similarly well, we use a formal music theoretical model of tonal harmony to select the chord candidate that best matches the sequence based on the surrounding chords. In an experiment we show that exploiting metrical and harmonic knowledge yields statistically significant chord transcription improvements on a corpus of 217 Beatles, Queen, and Zweieck songs.

1. INTRODUCTION

Chord labels are an indispensable and ubiquitous aid for modern musicians. Although classically trained performers still rely mainly on printed scores, describing in high detail how a piece of music should be performed, the emergence of jazz, improvised, and popular music gave rise to the need for more flexible and abstract representations of musical harmony. This led to a notational vehicle often referred to as a *lead sheet*. A lead sheet typically contains only the melody of a composition accompanied with the essential harmonic changes denoted with chord labels. It can be considered a rather informal map that guides the performers and specifies the boundaries of the musical playground. Also, in music theory, music education, composition, and harmony analysis, chord labels have proven to be a convenient way of abstracting from individual notes in a score. Hence, these days chord labels are omnipresent: there are publishers that specialise in publishing lead sheets, and many lead sheets circulate on the internet.

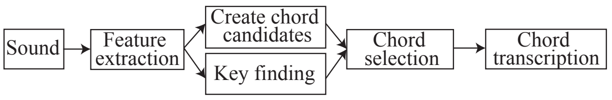


Figure 1. A schematic outline of the MPTREE system.

The many possible applications of chord labels have sparked research focusing specifically on chords. Many Music Information Retrieval (MIR) tasks, like similarity estimation, genre detection, or query by humming, can benefit from some reduction of the raw audio signal into a manageable symbolic representation. Although much progress has been made, multiple fundamental frequency (F0) estimation, the holy grail in polyphonic music transcription, might still be considered too unreliable and imprecise for many MIR tasks. Chord transcription, which deals with transforming polyphonic audio into musically feasible symbolic annotations, has offered a welcome alternative. For example, in automatic harmony analysis [8], and similarity estimation [5], chord labels are used as primary data representation.

In this paper we present a novel system, named MPTREE, [1] that automatically transcribes chord labels from a polyphonic musical audio source. This system is different from most other chord transcription systems, e.g. [12], in that it does not rely on statistical learning. Although machine learning has brought chord transcription (and MIR in general) many merits, we believe that there is a limit to what can be learned from musical data alone [6]. Certain musical segments can only be annotated correctly when musical knowledge not exhibited in the data is taken into account as well. Moreover, Hidden Markov Models (HMMs), frequently used to model the transitions between chords, model only the transition between a small number of subsequent chords, and have a bias towards sequences they have been trained on. Our system, on the other hand, relies on a knowledge-based model of tonal harmony. The HARMTRACE [2] harmony model [4] is explicitly designed for modelling the relations between chords, also over a long time span. In this paper we show how this harmony model can be employed to improve chord transcription.

A global outline of the system is presented in Figure 1. We start by briefly reviewing some important literature in Section 2. Next, we give a complete outline of the MPTREE

[1] (Musical) Model Propelled TRanscription of Euphonic Entities
[2] Harmony Analysis and Retrieval of Music with Type-level Representations of Abstract Chords Entities

system in Section 3. In Sections 4 and 5 we discuss the experiments and results. Finally, we conclude the paper by recapitulating the main advantages and disadvantages of the MPTREE system, and highlight some directions for future research.

Contribution. In this paper we bridge the gap between top-down symbolic music analysis and bottom-up audio feature extraction. We show that exploiting metrical position and a model of tonal harmony yields significant chord transcription improvements on 217 songs by the Beatles, Queen, and Zweieck.

2. RELATED WORK

The first computational approaches to automatic chord transcription from musical audio emerged at the end of the 1990s. The first audio chord transcription system was developed by Fujishima [3]. In general, the outline of Fujishima's system is not so different from the chord transcription systems nowadays developed and also no so different from the system presented here. First, chroma features [15] are extracted at every frame, representing the intensities of the twelve different pitch classes as found in the spectrum. Next, the chroma vectors are matched with chord profiles; in Fujishima's case this is done with an Euclidean distance. Although the used digital signal processing parameters may vary, most approaches towards automatic chord transcription use a chroma feature based representation and differ in other aspects, like chroma tuning, noise reduction, chord transition smoothing, and harmonics removal. For an elaborate review of related work on automatic chord transcription we refer to Mauch [9].

From 2008 on, chord transcription has received a considerable amount of attention in the yearly benchmarking challenge MIREX. [3] Each year, between 7 and 19 different chord transcription algorithms were evaluated. In 2008, the system of Bello and Pickens [1], which was the first to synchronise chroma vectors at every beat, performed the best. The following year, Mauch et al. [10] presented a system that gave good results by structurally segmenting a piece and combining chroma information from multiple occurrences of the same segment type. In 2010, Mauch et al. [11] improved their previous results by using an approximate note transcription technique. In 2011, the system of Ni et al. [12], using only machine learning techniques, gave comparable results.

3. SYSTEM OUTLINE

An outline of the MPTREE system is shown in the flowchart of Figure 2. First, we extract chroma features and beat locations from the audio signal and synchronise the chroma features at the beat positions (Section 3.1). The chroma features are used to estimate the global key and possible modulations in the musical audio (Section 3.4), and for creating a sequence of chord candidate lists (Section 3.2). These candidate lists contain the chords that match the chroma well a particular beat position. If there is a lot

[3] `http://www.music-ir.org/mirex/wiki/MIREX_HOME`

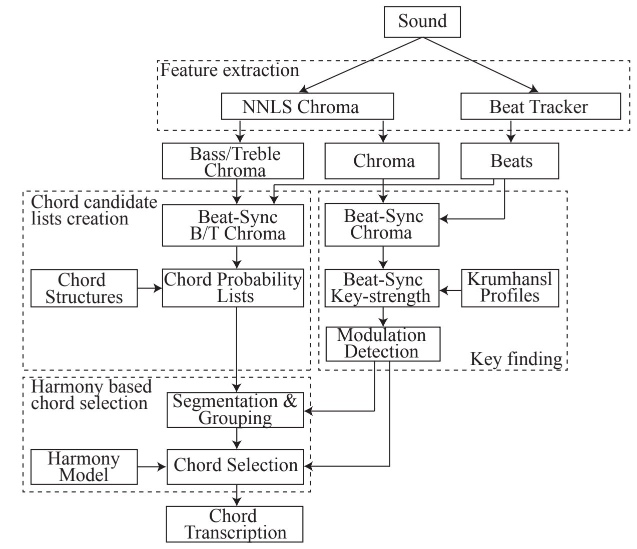


Figure 2. A schematic outline of the MPTREE system. The boxes with dotted lines denote the high level modules as outlined in Figure 1.

of uncertainty in the data, these lists might contain multiple chords; however, if there is a strong match between the spectrum and one particular chord candidate, the lists will contain a single candidate. Subsequently, the sequence of chord candidate lists is segmented (Section 3.5). Finally, the best matching sequence per segment is selected by expanding all possible sequences, and preferring the sequence with the fewest harmony errors (Section 3.6).

3.1 Feature extraction front-end

Our work depends heavily on the Vamp plugin architecture. [4] As feature extraction front-end we rely on the NNLS Chroma Vamp plugin [5] developed by Mauch [9]. The NNLS Chroma plugin transforms an audio signal into two 12-dimensional chroma vectors representing the harmonic content at each frame. The first chroma vector (*bass*) represents the low notes and emphasises the lower frequencies, while the second (*treble*) represents the higher notes, emphasising higher frequencies. The idea behind this separation is to model the prominent role of the bass note in chord transcription. We present a brief overview of the most important properties and parameters of the NNLS plugin. For specific signal processing details we refer to Mauch [9].

We use the sonic-annotator [6] (version 0.6) as Vamp host, and sample the audio tracks at 44,100 Hz. If the audio file contains two stereo channels, the mean of both channels is used for analysis. We set the sonic-annotator to use a Hann window of 16,384 samples and a hop size, i.e. the amount of samples that overlap between two subsequent frames, of 2,048 samples. Next, the spectrogram is calculated at each frame using a discrete-time Fourier transform and mapped to a spectrogram with bins that are linearly spaced in log-frequency (similar to a constant-Q transform). The NNLS

[4] `http://www.vamp-plugins.org`
[5] `http://isophonics.net/nnls-chroma`
[6] `http://omras2.org/SonicAnnotator`

C:Maj	1	0	0	0	1	0	0	1	0	0	0	0
D:Min	0	0	1	0	0	1	0	0	0	1	0	0
	C	C♯	D	E♭	E	F	F♯	G	G♯	A	B♭	B

Table 1. A binary chord structure of a C major and a D minor chord, which are matched against the chroma features.

				0.93 C^7			
				0.96 Am			
			0.94 G	0.97 C		0.94 Bm	
…	1.00 C	1.00 F	1.00 Gm	1.00 Em	1.00 F	1.00 B	…
…	1	2	3	4	1	2	…

Table 2. An excerpt of a sequence of chord candidate lists. The number to the left of the chord label represents its normalised Euclidean distance to the current beat aligned chroma vector. Below the candidate lists the beat position within the bar is printed.

Chroma Vamp plugin also accounts for tuning differences in the audio signal. Also, the NNLS plugin accounts for harmonics other then the F0 of chord notes by estimating which pitch activation generates an interference pattern that best matches the partials found in a spectrum.

3.2 Beat-synchronous chord probability estimation

After obtaining bass and treble chroma features, we beat-synchronise them by averaging the feature vectors between two beats. For this, we obtain a list of beat positions by using the Queen Mary, University of London, Bar and Beat Tracker plugin [2]. [7] Besides beat timestamps, this beat tracker also outputs the position of the beat inside the bar.

To estimate the probability of a particular chord sounding at a beat position, we assemble a dictionary of chords that we expect to occur in the music. A chord is represented as a binary 12-dimensional vector in which the simultaneously sounding pitch classes are denoted with a 1 (see the examples in Table 1). This allows us to model any possible chord within one octave. Currently, we use a limited chord dictionary with three chord structures: major, minor, and dominant seventh. We chose these three chords because they map nicely to the chord classes used by the HARMTRACE harmony model. In HARMTRACE, chords are categorised in four classes: major chords, minor chords, dominant seventh chords, and diminished seventh chords (see [4, Chapter 4] for details). However, because diminished seventh chords are not very common in pop music, we ignored this class in this study. The bass note of the chord is modelled with an additional 12-dimensional vector containing only one pitch for the bass note, to match the bass chroma vector as outputted by the NNLS chroma plugin. Next, we generate the chord dictionary by cyclically rotating all chord structures for all twelve semitones, yielding 48 different chord candidates, and a "no chord" structure containing only 0's.

Having a matrix of beat-synchronised bass and treble chromagrams and a chord dictionary, we estimate the probability of a chord sounding at a particular beat by calculating the Euclidean distance between the chord structures and the chroma feature. These distances are calculated for every chord candidate at every beat. Next, we sort the chord candidates by descending Euclidean distance. To obtain a relative measure of the fit between a chord candidate and the chroma vector in the range $[0, 1]$, the distances are normalised by dividing them by distance of the best matching chord candidate. In case the information in the spectrum clearly favours a certain chord candidate, the initial differences in normalised distances will be relatively large and will decrease quickly after the head position. Hence, we can use these differences as a measure of relative chord candidate preference. If this preference is very strong, the top chord candidate will be selected to represent that beat. If this preference is less pronounced, we use the HARMTRACE harmony model to decide which of the chord candidates would make most sense, harmonically. Typically, this results in a sequence of chord candidates similar to the one shown in Table 2. The selection is performed by cutting off the chord candidate list at a fixed distance. The cut-off value is an important parameter to the model, influencing both the speed and the transcription quality of the system. After some experimentation we found that a cut-off value of 0.9 gives good results.

[7] http://vamp-plugins.org/plugin-doc/qm-vamp-plugins.html\#qm-barbeattracker

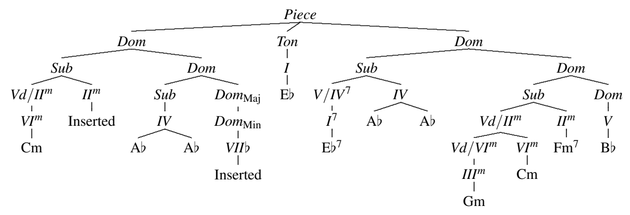


Figure 3. An excerpt of the HARMTRACE analysis of *The long and winding road* by the Beatles (of which the ground-truth chord annotations were used for parsing). The *Vd/X* represents a diatonic fifth succession, and a V/X^7 denotes a secondary dominant.

3.3 A model of tonal harmony

Given a list of chord candidates, we select a harmonically sensible sequence by exploiting a formal model of tonal harmony. This model, which is elaborately explained by De Haas [4], takes a sequence of symbolic chord labels as input and automatically derives a tree structure explaining the function of the chords in their tonal context. Figure 3 depicts an excerpt of the harmony analysis of *The long and winding road* by the Beatles.

Extending the ideas of Rohrmeier [13], a piece is modelled as a sequence of tonic and dominant nodes (*Ton* and *Dom*, respectively) that represent the global patterns of harmonic tension and release. Every *Dom* node can be preceded by a subordinate sub-dominant (*Sub*) building up the tension towards the dominant. Finally, a branch will always end in a scale degree node, representing the relation between the actual chord and the key of the piece, and the leaves of the tree show the actual input chord labels. On the path from functional annotation (*Ton*, *Dom*, and *Sub*) to chord label, various harmonic annotations, like secondary dominants, tritone substitutions, diatonic fifth chains, diminished seventh chord transpositions, etc., can occur, explaining the role of a chord label in its tonal context. If a sequence does not match the harmonic specification, like in the first phrase of Figure 3, an input chord label is automatically deleted or inserted to match the specification. Hence, for a sequence of chords we can always derive an automatic harmonic analysis.

3.4 Key-finding

To be able to use the HARMTRACE harmony model for the selection of chord sequences that are music theoretically realistic, we require information about the key of the piece. To fulfil this requirement, we present a key-finding algorithm inspired by the ideas of Temperley [14, Chapter 7] and Krumhansl [7, Chapter 4]. Again, for feature extraction we depend on the NNLS chroma Vamp plugin, which allows for exporting different kind of audio features. For key-finding we export a single tuned chroma feature without the NNLS pitch activation estimation.

To estimate the key of a piece and the possible modulations, we use a *key-profiles* based algorithm. A key-profile is a 12 value vector representing the stability of the twelve pitch classes relative to a given key. The values of these profiles are based on empirical measurements of Krumhansl and Kessler [7], in which subjects were asked to rate how well a pitchclass fits a previously established tonal context on a 1 to 7 scale. Given the major and minor key profiles, a key-strength table K is created. This table stores the estimated strength of all 24 keys at every beat position. The key strength is estimated by calculating the Pearson correlation coefficient, r. A value of r close to 0 indicates that there is little to no relation between the key-profile and chroma vector, whereas a value close to 1 or -1 indicates a positive or negative linear dependence between the key-profile and the chroma vector, respectively.

Matching the key-profiles at every beat does not yet result in the desired key assignment; because beat size segments are rather small, key changes occur too often. To overcome this problem, we use a simple dynamic programming algorithm based on the algorithm in [14] to smooth the key changes. We create a table M storing the cumulative key-strength of every key at every beat, and minimise the number of modulations. Switching to another key, i.e. changing the column j in M, is penalised. This behaviour is captured in the following recursive formula:

$$\begin{aligned} M[0,j] &= K[0,j] \\ M[i,j] &= \max \begin{cases} M[i-1,j]+K[i,j], \\ M[i-1,j]+K[i,k]+p, \end{cases} \\ &\quad \text{where } \{k \mid \forall x : K[i,x] \leqslant K[i,k]\} \end{aligned}$$

Here, M stores the cumulative key-strength for every i^{th} beat and every j^{th} key. Similarly, K stores the correlation between every i^{th} chroma vector and j^{th} key profile. k denotes the index of the best matching key at beat i. The parameter p specifies the modulation penalty. We found a value of 1 for p to give good results. Finally, we obtain the definite key assignment by keeping track of the maximum cumulative key-strength at every beat, and constraining the key segments to be at least 16 beats long.

3.5 Segmentation and grouping

Given a sequence of chord candidate lists, we analyse all possible chords sequence combinations with HARMTRACE and select the simplest analysis with the least amount of errors. However, the number of possible combinations grows exponentially with the number of candidate lists. Hence, it is vital to split our sequence of chord candidate lists into smaller, musically meaningful segments.

Also, from a musical point of view, it is unrealistic to expect chords to change at every beat. Therefore, we reduce the space of analysed sequences by merging subsequent chord candidate lists that contain the same chords. The candidate lists are merged by taking the intersection between two adjacent lists, if the intersection contains at least one chord. In this procedure we take into account that chords are more likely to change on strong metrical positions by adding two additional constraints: when two candidate lists are merged, the first and leftmost list must be positioned either at the first or third beat of the bar. The merging procedure is executed sequentially, and merged candidate lists can be merged again with the subsequent chord candidate list. For example, if the candidate lists at beat position 1 and 2 are merged, the merged list can again merge with beat position 3 if the intersection contains at least one chord. Finally, the probabilities of the merged candidate lists are summed, and the lists are sorted by descending probability.

Subsequently, the sequence of chord candidate lists is segmented on the basis of the estimated key, resulting in segments that contain only a single key assignment. Nevertheless, these sequences are still rather long for analysing all possible combinations. Within the HARMTRACE harmony model, a piece is viewed as a collection of tonics (*Ton*) and dominants (*Dom*) nodes. Hence, from the parsing point of view, splitting a chord sequence into segments that match the subtrees rooted by a *Ton* or *Dom* seems natural. Because local information about the key is available, we can calculate the key-relative scale degrees of the chords and split a sequence at every beat where a *I* or *V* scale degree occurs in a chord candidate list. This gives us sequences that are short, but still musically meaningful. In case our key-finding method is off the mark, and we still end up with a rather long sequence, we enforce the sequences to be no longer than 12 chords, and expand into no more than 30 different candidate sequences.

3.6 Chord selection by parsing

Now that we have access to both a segmented sequence of chord candidate lists and local key information, we are ready to apply the HARMTRACE harmony model. For every segment we parse all possible combinations of chord sequences and select the sequence that has the lowest error-ratio. The error-ratio is the number of insertions and deletions of the error-correcting parser divided by the number of chords. When two sequences have the same error-ratio, we select the most simple solution by picking the sequence that returns the smallest parse tree. In case the parse tree size is also identical, we select the sequence returning the parse tree of least depth.

The harmony model used in MPTREE is not the exact same model as the one described by De Haas [4, Chapter 4]. The original HARMTRACE harmony model exhibits a bias towards jazz harmony. Therefore, we made several adaptations, but the majority of the specifications remained unchanged. The original harmony model was designed to do an automatic harmonic analysis of a chord sequences and could explain a vast amount of exotic harmonic phenomena. Within the pop dataset on which the MPTREE system is evaluated, some of these specifications are unnecessary. Hence, we remove some of the specifications accounting for jazz-specific chord changes. [8] Furthermore, we add two rules that account for some blues phenomena. [9] The Haskell code of both the HARMTRACE models and the MPTREE system is freely available online. [10]

4. EXPERIMENTS

To measure the effect of the various modules on chord transcription performance we evaluate four different versions of the system described before. The simplest system, named SIMPLE, always selects the chord that best matches the bass and treble chroma vectors. The second system, GROUP, also picks the best matching chord candidate, but does incorporate the grouping as described in Section 3.5. The third system is the full MPTREE system, including key-finding. Finally, we include a fourth system, MPTREE$^{\text{key}}$, to measure the effect of the key-finding. MPTREE$^{\text{key}}$ does not use key-finding, but instead uses ground-truth key annotations [10]. All systems are implemented in the functional programming language Haskell and compiled using the Glasgow Haskell Compiler, version 7.4.1.

We evaluate the quality of an automatic chord transcription by comparing it to a transcription of the same piece made by a human expert. We evaluate our system on 179 songs from 12 Beatles albums, 20 Queen songs, and 18 Zweieck songs [10]. [11] The chord vocabulary for the MIREX evaluation is limited to 24 major and minor chords augmented with a "no chord" label, to be used for silence or non-harmonic passages, for instance. In accordance with MIREX, we also use these 25 classes. The translation from the three chord classes of HARMTRACE to major and minor chords is trivial: chords of the major and dominant class are classified as major, and chords of the minor class are classified as minor.

Typically in MIREX, the *relative correct overlap* (RCO) is used as a measure of transcription accuracy. The RCO is defined as the total duration of correctly overlapping chords divided by the total duration of the song. Both the ground-truth and the automatic chord transcription consist of a chord label and an accompanying onset and offset time-stamp. We approximate the RCO by sampling both the ground-truth and the automatic annotations every 10ms and dividing the number of correctly annotated samples by the total number of samples.

[8] The specifications with numbers 20, 21, and 22 were removed.

[9] Allowing dominant seventh chords at the *IV* and *I* scale degree to function respectively as sub-dominant and tonic, to be precise.

[10] `http://hackage.haskell.org/package/HarmTrace-2.0`

[11] `http://isophonics.net/content/reference-annotations`

	SIMPLE	GROUP	MPTREE	MPTREE$^{\text{key}}$
RCO	0.688	0.736	0.739	0.741
Running time	$5m1s$	$5m9s$	$10m23s$	$7m37s$

Table 3. The relative correct overlap and the running times for the four evaluated chord transcription systems.

5. RESULTS

We have compared the MPTREE, MPTREE$^{\text{key}}$, GROUP, and the baseline SIMPLE system on 217 songs of the Beatles, Queen, and Zweieck. All runs were performed on the same Intel Core i7-2600 Processor running at 3.40GHz. The measured differences in RCO and running times are displayed in Table 3.

We tested whether the differences in RCO are statistically significant by performing a non-parametric Friedman test [12] with a significance level of $\alpha = 0.05$. The Friedman ANOVA is chosen because the underlying distribution of the RCO data is unknown, and, in contrast to a regular ANOVA, the Friedman does not assume a specific distribution of variance. To determine which pairs of measurements differ significantly, a post-hoc Tukey HSD test is conducted. Within the MIREX challenge the same statistical procedure is followed. There are significant differences between the four systems, $\chi^2(3, N = 217) = 339$, $p < 0.0001$. Not all pairwise differences between systems are statistically significant; the difference between GROUP and MPTREE, and between MPTREE and MPTREE$^{\text{key}}$ are not significant. All other pairwise differences (including the difference between MPTREE$^{\text{key}}$ and GROUP) are statistically significant.

Considering the differences between the MPTREE$^{\text{key}}$ system and the SIMPLE and GROUP systems, we can conclude that using the HARMTRACE harmony model for chord candidate selection improves chord transcription performance, if correct key information is available. This difference in performance cannot be attributed to the merging function described in Section 3.5 alone. However, clearly a lot of the performance gain must be attributed to this merging function. Hence, we can conclude that forcing chords not to change often and mainly at strong metrical positions improves transcription performance. Although the difference between MPTREE an MPTREE$^{\text{key}}$ is not statistically significant, the errors in the key-finding do have an effect on the transcription performance, since the difference between GROUP and MPTREE is not, but the difference between GROUP and MPTREE$^{\text{key}}$ is statistically significant.

The running times as shown in Table 3 exclude the time

[12] All statistical tests were performed in Matlab 2011a.

taken by the Vamp feature extraction plugins. The results show a trade-off between transcription performance and computation time. However, the running times are acceptable, less than 3 seconds per song on average.

6. DISCUSSION

In this paper we aim at bridging the gap between bottom-up audio feature extraction and top-down symbolic music analysis. We demonstrate in a proof-of-concept how automatic chord transcription can be improved by using domain-specific knowledge about the metrical structure and tonal harmony. For feature extraction we rely on the NNLS chroma and the Bar and Beat Tracker Vamp plugin. We show that preferring harmonically valid combinations of chords yields better chord transcriptions than just picking the best matching chord at each beat, even after smoothing the chord changes with a merging function. This result is good, especially if we consider that we have only connected the different technologies without extensively tuning their parameters.

It is difficult to compare the results of this paper with current the state-of-the-art in an absolute manner; this must be done in a next iteration of the MIREX challenge. The dataset used in this paper closely resembles the one used in MIREX in 2010 and 2011. However, although the same ground-truth is used, many different remastered editions of the Beatles and Queen songs exist, and some editions are known to deviate from these ground-truth annotations. We used the LabROSA script to improve the alignment between our Beatles corpus and the ground-truth, [13] but it is hard to tell whether the results in Section 4 have been influenced by remastering artifacts. However, if this is the case, the results of all compared systems are affected equally.

In the 2011 edition of MIREX, all systems were evaluated as described in Section 4, yielding RCO values between 0.126 and 0.829, and a deliberately over-fitted result yielding an RCO of 0.976. Clearly, a system with a model trained in this manner will very likely perform poorly on unseen data. All algorithms that returned an RCO above 0.740 were HMM-based machine learning approaches, and it is unclear how much they have over-fitted on the used dataset. The chances that the HARMTRACE harmony model is over-fitting the used dataset are very low. After all, candidate chord sequences are not selected based on how often they occur in a training sample, but only based on whether they follow the general rules of tonal harmony. Another benefit of the knowledge-based approach presented in this article, is that we can analyse why certain chord sequences are preferred over others and reason about whether these choices are justified. An HMM remains a black box, which does not provide insights into the choices made.

Nevertheless, there is still room for improvement. Perhaps that using different signal processing parameters, or different plugins improve the results. Moreover, we expect that carefully adjusting the parameters and tailoring the modules to maximise their interoperability will result in an increase of performance. Also, Mauch et al. [9] successfully improved chord transcription performance by averaging the chroma vectors of segments that were classified as having very similar harmonies. Such a technique could possibly improve the results in the MPTREE system as well. We have shown that connecting state-of-the-art low-level feature extraction methods to high-level symbolic knowledge systems offers new capabilities to boost the analysis and retrieval of musical audio. We also expect similar combinations to be able to improve other common MIR related tasks, such as cover-song finding, music transcription, and structural analysis.

Acknowledgements We thank Anja Volk and Remco C. Veltkamp for providing comments on earlier drafts of this article. W. Bas de Haas is supported by the Netherlands Organization for Scientific Research, NWO-VIDI grant 276-35-001, and José Pedro Magalhães by EPSRC grant number EP/J010995/1.

[13] `http://labrosa.ee.columbia.edu/matlab/beatles_fprint/`

7. REFERENCES

[1] J.P. Bello and J. Pickens. A robust mid-level representation for harmonic content in music signals. In *ISMIR Proceedings*, pages 304–311, 2005.

[2] M.E.P. Davies and M.D. Plumbley. Context-dependent beat tracking of musical audio. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(3):1009–1020, 2007.

[3] T. Fujishima. Realtime chord recognition of musical sound: A system using common Lisp music. In *ISMIR Proceedings*, pages 464–467, 1999.

[4] W.B. de Haas. *Music information retrieval based on tonal harmony*. PhD thesis, Utrecht University, 2012.

[5] W.B. de Haas, J.P. Magalhães, F. Wiering, and R.C. Veltkamp. HarmTrace: Improving harmonic similarity estimation using functional harmony analysis. In *ISMIR Proceedings*, 2011.

[6] W.B. de Haas and F. Wiering. Hooked on music information retrieval. *Empirical Musicology Review*, 5(4):176–185, 2010.

[7] C.L. Krumhansl. *Cognitive Foundations of Musical Pitch*. Oxford University Press, USA, 2001.

[8] J. P. Magalhães and W. B. de Haas. Functional Modelling of Musical Harmony—an Experience Report. In *Proceedings of the International Conference on Functional Programming*, pages 156–162, 2011.

[9] M. Mauch. *Automatic chord transcription from audio using computational models of musical context*. PhD thesis, Queen Mary University of London, 2010.

[10] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. Omras2 metadata project 2009. In *ISMIR Proceedings*, 2009.

[11] M. Mauch and S. Dixon. Approximate note transcription for the improved identification of difficult chords. In *ISMIR Proceedings*, pages 135–140, 2010.

[12] Y. Ni, M. McVicar, R. Santos-Rodriguez, and T. De Bie. An end-to-end machine learning system for harmonic analysis of music. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(6):1771–1783, 2012.

[13] M. Rohrmeier. Towards a generative syntax of tonal harmony. *Journal of Mathematics and Music*, 5(1):35–53, 2011.

[14] D. Temperley. *The Cognition of Basic Musical Structures*. Cambridge, MA, MIT Press, 2001.

[15] G.H. Wakefield. Mathematical representation of joint time-chroma distributions. In *Conference on Advanced Signal Processing Algorithms, Architectures, and Implementations*, pages 637–645, 1999.

REDUCING TEMPO OCTAVE ERRORS BY PERIODICITY VECTOR CODING AND SVM LEARNING

Aggelos Gkiokas[1,2]
[1]Institute for Language and Speech Processing / R.C. Athena
[2]National Technical University of Athens
agkiokas@ilsp.gr

Vassilis Katsouros
Institute for Language and Speech Processing / R.C. Athena
vsk@ilsp.gr

George Carayannis
National Technical University of Athens
carayan@softlab.ece.ntua.gr

ABSTRACT

In this paper we present a method for learning tempo classes in order to reduce tempo octave errors. There are two main contributions of this paper in the rhythm analysis field. Firstly, a novel technique is proposed to code the rhythm periodicity functions of a music signal. Target tempi range is divided into overlapping “tempo bands” and the periodicity function is filtered by triangular masks aligned to those tempo bands, in order to calculate the respective saliencies, followed by the application of the DCT transform on band strengths.

The second contribution is the adoption of Support Vector Machines to learn broad tempo classes from the coded periodicity vectors. Training instances are assigned a tempo class according to annotated tempo. The classes are assumed to correspond to “music speed”. At classification phase, each target excerpt is assigned a tempo class label by the SVM. Target periodicity vector is masked by the predicted tempo class range, and tempo is estimated by peak picking in the reduced periodicity vector.

The proposed method was evaluated on the benchmark ISMIR 2004 Tempo Induction Evaluation Exchange Dataset for both tempo class and tempo value estimation tasks. Results indicate that the proposed approach provides an efficient framework to tackle the tempo estimation task.

1. INTRODUCTION

Most tempo estimation systems suffer from detecting the correct metrical level, i.e. tend to result in tempi that are fractions or multiples of the groundtruth tempo. Such errors are usually found in the literature as “octave errors”. Although many methods are reported to achieve accuracy over 90% [1-3] when ignoring octave errors, i.e. accuracy for finding the exact, double, treble, half or 1/3 of groundtruth tempo (known as *accuracy2* measure), the accuracy of these methods decreases to 50~60% for finding the exact tempo (*accuracy1*). More details on rhythm analysis systems and evaluation measures can be found in [4,5].

Two certain contemporary aspects arise when considering the octave error problem. First, when allowing an algorithm to make errors that correspond to the different metrical levels, one can say that such an approach is more close to the notion of perceptual tempo. Different users would tap at different metrical levels for the same song. Even a single user might tap at different metrical levels for the same song at different psychosocial states. Thus, it can be claimed that during the evaluation process of a tempo estimation system the usage of a single groundtruth value is not always feasible. On the other hand, not all fractions and multiples can be considered as musically correct.

One solution was the P-score evaluation measure introduced in MIREX 2005 Audio Tempo Extraction Task[1] where each excerpt was annotated with two dominant tempi, and their relative strength. Algorithms should suggest two tempi and the P-score is defined as the mean relative strength of the correct estimated tempi within an 8% tolerance. In this context, deciding the correct metrical level is less crucial.

However, consider the following example. The 4th training instance on McKinley’s dataset excerpt, which exhibits a 6/8 measure and 126 bpm tempo, was annotated by 40 experts. 10 of them tapped at eight note level, while 30 tapped at dotted quarter notes. Thus tempo value 42 bpm can be considered more salient than musical tempo 126 bpm. If these people were asked to characterize this excerpt as “slow” or “fast”, probably they would judge it as slow. Although the reliability of annotations can always be questioned, we can conclude that there is a strong relation of the notion of musical “speed” to the perceptual tempo (or metrical level).

Choosing the correct metrical level usually relies on incorporating some prior knowledge, mostly in terms of calculating prior tempi distribution [2]. Other methods adopt metrical models [1,3], inference from inter-onset intervals [6] or by considering the most predominant peak in periodicity vector as the correct tempo [7]. Seyerlehner et al. [8] incorporate instance based learning techniques,

[1] http://www.music-ir.org/mirex/wiki/2005:Audio_Tempo_Extraction

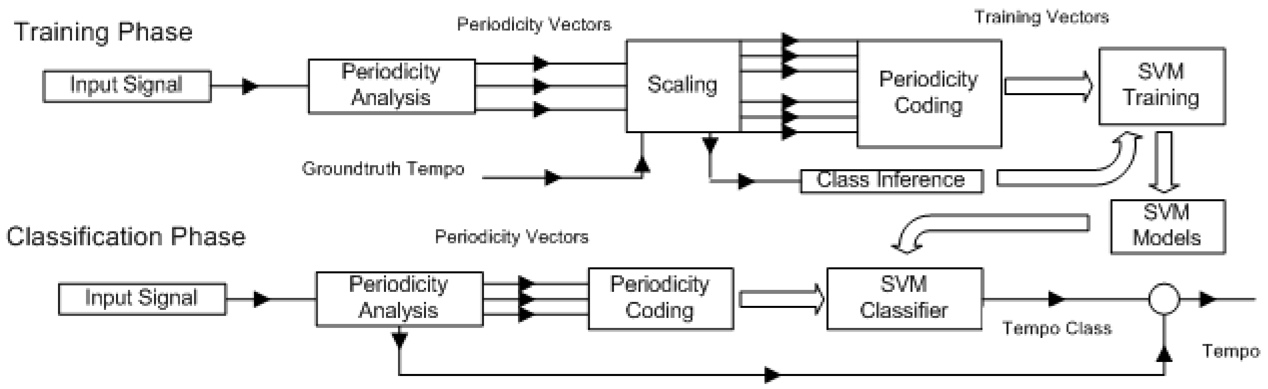


Figure 1. Overview of the proposed method.

where the periodicity vector of the target music piece is compared to periodicity functions of other annotated excerpts. The assigned tempo is equal to tempo of the excerpt with the most similar periodicity vector. In a similar manner, Peeters adopts spectral templates and a learning schema for estimating tempo [9].

Two recent approaches on characterizing the music speed are remarkable. Eronen and Klapuri [10] presented a tempo estimation system, where the predicted tempo is chosen by comparing scaled versions of the periodicity vector of the target excerpt with periodicity vectors of tempo annotated pieces. In the same paper, results were reported for a classification subsystem that classified music excerpts to three categories: slow, medium and fast. In [11] Hockman and Fujinaga proposed a system that classifies music pieces to fast/slow. Annotations were not extracted with the knowledge of any groundtruth tempo but directly from user tags on YouTube videos. Without any rhythmic analysis, but based solely on baseline frame-level features, their method achieved a classification accuracy of 96% by adopting the AdaBoost classifier. In [12] Smith proposed a system for identifying octave errors made by a baseline beat tracker.

In this paper, we present a method of learning tempo octaves, i.e. classifying a music excerpt to one of the three categories: slow, moderate and fast. The proposed method exhibits two key features. Firstly, a coded representation of periodicity vector similar to the popular MFCC features is proposed. Secondly, we adopt an SVM learner to learn tempo octaves. SVM's has been greatly used in classification tasks in the MIR domain such as [13, 14]. We applied the proposed octave learning method to a baseline tempo estimation method [3] in order to limit the target tempi space and enhance tempo extraction accuracy. Evaluation results indicate that the proposed technique enhances greatly the tempo estimation accuracy.

The rest of the paper is organized as follows. In Section 2 an overview of the proposed method is described. Section 3 is dedicated to present the periodicity function extraction procedure. SVM learning formulation is described in Section 4, while in Section 5 the tempo estimation method is presented. Evaluation results and discussion on the proposed method conclude this paper in Sections 6 and 7 respectively.

2. SYSTEM OVERVIEW

Figure 1 shows an overview of the proposed system. In training phase, periodicity analysis of the input signal is performed. A set of vectors that is supposed to contain all rhythmic information of the signal is extracted. Next, the extracted periodicity vectors are rescaled in order to produce more training instances. The periodicity vectors are then coded to a more compact representation, and along with the respective tempo class (slow, moderate, fast) which is inferred from the groundtruth tempo, are used to train the SVM model.

In classification phase, the unknown input signal is processed by the periodicity analysis module. Periodicity vectors are coded as above and feed the SVM classifier. The output class is then combined with the periodicity vectors of the input signal to find the tempo value that is consistent to the metrical level of the SVM classifier.

3. REPRESENTING RHYTHMIC CONTENT

3.1 Periodicity Analysis

Periodicity analysis is performed by the adoption of the method presented in [3]. The constant Q transform is applied to the signal, and followed by the harmonic/percussive separation algorithm reported in [15]. Two feature multidimensional sequences are extracted by the harmonic/percussive parts of the signal respectively. Eight band energies from the percussive part, denoted as $x^i, i = 1..8$ and chroma vectors from the harmonic part denoted as $ch^j, j = 1..12$. Feature sequences are convolved with a bank of resonators with oscillation frequencies set to the tempo analysis range. Resonators' outputs are segmented by square windows and the maximum values of resonators' outputs are considered as the salient values of each feature sequence to each tempo value. We denote as

$p_n^{feature}[t]$ the periodicity vectors for the input signal where $feature \in \{ch^j \cup x^i, j = 1..12, i = 1..8\}$ denotes the feature type, n denotes the time index and $t = \{30..500\}$ denotes the tempo analysis range. Reader should note that this range is larger than target tempi search space. This is due to the fact that periodicity functions contain rhythmic information in frequency regions beyond the groundtruth tempo.

3.2 Scaling Training Vectors

Since there is lack of large amount of annotated tempo data, we could produce artificial data by rescaling a music signal to faster and slower tempi. However, this approach would be computational intensive. To overcome this problem we exploit the following property of the periodicity vector, i.e., tempo-scaled versions of a signal, say by a value of *α,* produce inversely scaled versions of the periodicity vector by the value of 1/*α*. Thus, for a music signal $y[i]$, with periodicity function $p_n^{feature}[t]$

$$y[\alpha i] \xrightarrow{periodicity\ analysis} p_n^{feature}[t/\alpha]. \quad (1)$$

This property allows us to rescale directly the periodicity vectors, instead of the whole signal, reducing thus the complexity of the calculations.

In the same manner as in [10], all periodicity vectors extracted from each music excerpt are rescaled within a range of values for *α* around unity. If a music signal $y[i]$ is assigned a ground-truth tempo T_{ground}, then all α scaled versions of the periodicity vectors $p_n^{feature}[t]$ that result from $y[i]$ are assigned a tempo value $\alpha^{-1}T_{ground}$.

Under the assumption of almost constant tempo, periodicity functions are averaged for each feature across all segments *n* in order to capture better the overall rhythmic content of the signal as follows

$$\tilde{p}^{feature}[t] = \frac{1}{N}\sum_{n=1}^{N} p_n^{feature}[t] \quad (2)$$

3.3 Periodicity Coding

To satisfy the necessity to capture broad classes of tempo, it seems that it would be more efficient to use a more compact representation for the periodicity vectors. We shall exploit the fact that periodicity vectors of similar tempo music pieces, will not exhibit the same peaks, but may have a similar shape, or they will exhibit peaks in nearby tempi.

Some recent works deal with the spectral modeling of rhythmic information. Holzapfel and Stylianou [16] applied the scale transform to the autocorrelation function of music signals to form a rhythmic representation and exploit aspects of rhythmic similarity. Peeters [17] combines rhythm descriptors in a rhythm classification system

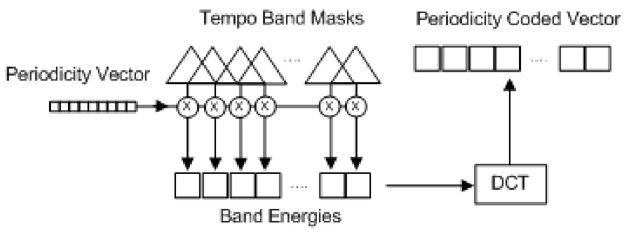


Figure 2. Periodicity vector coding process. $\otimes$ stands for inner product.

while in [9] the DFT of the accent function is subsampled at frequency bins that correspond to tempo harmonic series of certain meters.

In this paper, we introduce a filterbank-like analysis on the periodicity vectors, which is illustrated in Figure 2. The range of target tempi is divided into *K* equally tempo intervals with a 50% overlap between successive intervals. From each tempo interval, we utilize a symmetric triangular weighting mask.

The strength $s^{feature}[k]$ of the periodicity vector $\tilde{p}^{feature}[\cdot]$ for each of the K tempo intervals is calculated as the inner product with the respective mask:

$$s^{feature}[k] = \sum_{t=T_{\min}}^{T_{\max}} \tilde{p}^{feature}[t] \cdot mask^k[t] \quad (3)$$

where $mask^k[\cdot]$ denotes the mask of *k* tempo band.

Henceforth two problems arise from this modeling. Firstly, there is a strong correlation between features, caused mainly by the overlap of adjacent tempo bands. Secondly, different feature type sequences for the same piece will result to different periodicity vectors. For example energy evolution of lower spectral bands exhibit higher values in lower tempi whereas higher spectral bands exhibit faster changes, and thus higher tempi. Therefore periodicity vectors calculated from different features cannot be compared directly and cannot be treated in the same manner. To suppress the effects of band correlation we apply the Discrete Cosine Transform to each tempo-band strength vector $s^{feature}[\cdot]$, in order to obtain the uncorrelated coefficients $m^{feature}[\cdot]$:

$$m^{feature}[l] = \mathrm{DCT}(s^{feature}[\cdot]) \quad (4)$$

To cope with the different feature behaviour, the periodicity representation is finally formed by appending all coefficients $m^{feature}[\cdot]$ for each segment *n* to a single 20*K*-dimensional vector $\mathbf{m}$:

$$\mathbf{m} = [\mathbf{m}^{x^1} \mid \mathbf{m}^{x^2} \mid ... \mid \mathbf{m}^{x^8} \mid \mathbf{m}^{ch^1} \mid \mathbf{m}^{ch^1} \mid .. \mid \mathbf{m}^{ch^{12}}] \quad (5)$$

4. LEARNING TEMPO CLASSES

Let $\{(\mathbf{m}_l, t_l), l \in L\}$ denote the vectors extracted from the music signals using the method described in Section 3, where t_l are the annotated tempi. Depending on the value of t_l we assign excerpts to one of the following classes:

$$c_l = c(t_l) = \begin{cases} 1, & T_{slow} \geq t_l \\ 2, & T_{slow} < t_l < T_{fast} \\ 3, & t_l \geq T_{fast} \end{cases} \quad (6)$$

The thresholds T_{slow} and T_{fast} can either be user specified or inferred by data and they divide the target tempi range into the music speed classes of slow, moderate and fast.

We formulate two SVM problems for inferring the tempo classes; a **classification SVM** where we learn each class from the training data $\{(\mathbf{m}_l, c_l), l \in L\}$ and a **regression SVM** where we estimate a target tempo function from the training data $\{(\mathbf{m}_l, t_l), l \in L\}$ Then excerpts are classified one of the three classes by applying Eq. 6 on the estimated tempo value.

The conceptual difference between the two formulations is that while in classification we learn a function from feature space $\mathbb{R}^{20K}$ to {slow, moderate, fast}, i.e. discretization takes place directly on the training data (Eq. 6), in the case of regression discretization is applied to the regression estimate of the target tempo $\hat{t}_i$.

For the classification SVM the multiclass problem is split up to binary classification problems by applying the "one-vs-one" strategy. There is evidence [18] that the one-vs-one strategy is more suitable than the more common "one-vs-all" strategy, especially when there are imbalances between train classes, which is the case of the evaluation datasets (see Sec. 6).

In the case of the regression SVM the continuous tempo estimate $\hat{t}_i$ cannot be directly interpreted as an accurate tempo value, since in the signal representation $\mathbf{m}_i$ much of the rhythmic information such as the peaks in the periodicity vectors are suppressed by the periodicity coding process. However, the value $\hat{t}_i$ would give a rough estimate of the tempo that will be used in Eq. 6 to infer the tempo class of the excerpt.

5. ESTIMATING TEMPO

To extract the final tempo estimate from the periodicity function and the tempo class assigned by the SVM, we calculate an overall periodicity function by the superposition of the individual periodicity functions. In particular, the periodicity vectors are summed across the two feature types and the resulting vectors are multiplied to give the decision periodicity vector:

$$p[t] = \left(\sum_{i=1}^{8} p^{x_i}[t] \right) \left(\sum_{j=1}^{12} p^{ch_j}[t] \right) \quad (7)$$

Accordingly to the estimated class, $p[t]$ is reduced to the corresponding tempi range prescribed by Eq. 6. Final tempo estimate is decided as the most predominant peak in the reduced periodicity vector.

6. EVALUATION

The proposed method was evaluated on the ISMIR 2004 Tempo Induction Evaluation Exchange Dataset: *ballroom* and *songs* datasets [4]. Periodicity vectors were rescaled in the range [0.8, 1.2] with a 0.02 step. We divided the target tempi to classes by setting $T_{slow} = 80$ bpm and $T_{fast} = 130$ bpm in Eq. (6). Tempo bands number was set to K=20, tempo analysis region was set to [30..500] and target tempi space to [30..300]. Feature vector values where normalized to [-1, 1]. We adopted the LIBSVM implementation of SVM [19]. We used an RBF kernel for the SVM and parameter γ of the kernel was set to $1/20K$. For regression, we adopted the ε-support vector regression method. Experiments were run for various values of the parameter C (Eqs. 1 and 9 in [19]). Variations of ε (Eq. 9 in [19]) did not affect significantly the overall performance, and was set to 0.1.

To measure the generalization ability of the proposed method we adopted a three fold cross validation approach. Each evaluation set was split randomly to three equal subsets. Each subset was used as a test set and the remaining two as the training set. Evaluation measures were averaged on every train-test sets combination.

6.1 Assignment to Tempo Classes

The first series of experiments involves the classification accuracy to tempo classes. Figure 3 (top) illustrates the accuracy for various values of the parameter C on ballroom/songs datasets respectively, for both methods (classification / regression). The accuracy is almost constant for a wide range of C values, say for $10<C<500$. It must be noted that SVM classification approach outperforms the regression formulation. It is clear that although the tempo discretization process from the assignment of the music excerpt to one of the three classes introduces ambiguities for ground-truth tempi that are closer to either T_{slow} or T_{fast}, the classification approach is more efficient than the continuous regression approach. This can be explained by the fact that learning a continuous function on a high dimensionality space is much more demanding than separating instances into classes. In addition, the small number of training instances is probably not sufficient to learn such a function. However, there is no evidence that for larger scale experiments classification strategy will be more effective than the regression formulation.

To get a better insight to classification errors Table 1 presents the confusion matrix between classes for the classification approach (C=100), for both datasets. In the *ballroom* dataset classification fails for slow excerpts,

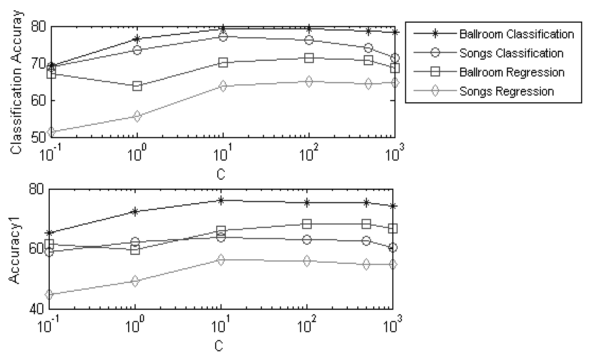


Figure 3. Top: Tempo class classification accuracy for both datasets/classifiers. Bottom: *Accuracy1* for both datasets/classifiers.

	Ballroom			Songs		
	Slow	Mod	Fast	Slow	Mod	Fast
Slow	**22**	33	44	**79**	16	5
Mod	2	**86**	12	17	**82**	1
Fast	1	32	**67**	46	28	**26**

Table 1. Confusion matrix in tempo category classification percentages for both datasets. Rows correspond to ground-truth and columns to estimates**.**

since most of them are classified as fast. This is due to the fact that there are very few excerpts with slow tempi. Thus, during training phase SVM fails to find reliable boundaries for this class. The same effect takes place in the case of fast excerpts in the *songs* dataset.

Figure 4 illustrates the tempo class error with respect to ground-truth tempo, along with dataset tempo distribution for both datasets. As expected, there are more classification errors near the tempo boundaries T_{slow} and T_{fast} with respect to the total test instances with similar tempo. Finding the optimal values for T_{slow} and T_{fast} is dataset depended and is out of the scope of this paper. T_{slow}, T_{fast} were chosen arbitrarily based on authors intuition and not on tempi distribution across data. For example, choosing $T_{slow} = 110$ and $T_{fast} = 150$ for *ballroom* dataset would give more separable classes (see Fig. 4). However, the errors ought to the quantization of tempi values demonstrate an inherent limitation of the proposed method.

Figure 3 (bottom) illustrates the *accuracy1* measure of tempo estimation for both classifiers (classification, regression) for various values of *C*. As expected by the results of previous section, the classification approach performs significantly better than regression. Comparing figures in Fig. 3, we can see that tempo class and tempo values estimations are very similar for the *ballroom* dataset: *accuracy1* is about 4 percent below tempo class accuracy. However, this is not the case for *songs* dataset, where *accuracy1* is significantly lower (>10%) than classification accuracy. To verify this, we estimated tempo for both

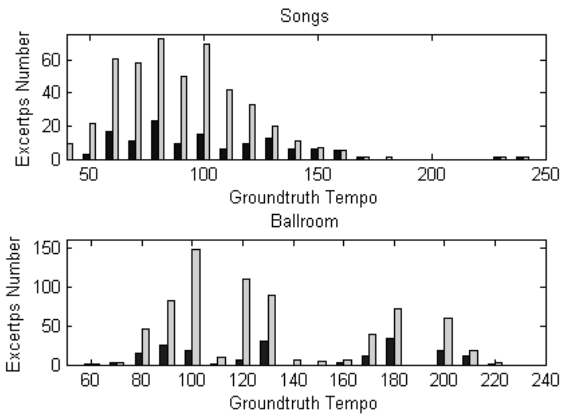


Figure 4. Distribution of classification errors (dark bars) with respect to ground-truth tempo compared to the overall dataset tempo distribution (light bars).

	Ballroom	**Songs**
Our Method	**75.93**	**63.87**
Baseline	59.89	58.49
SE1 [6]	78.51	40.86
SE2 [6]	73.78	60.43
Peeters [9]	75.2	-
Peeters [1]	65.2	49.5
Klapuri [2,4]	63.18	58.49
Uhle [4]	56.45	41.94
Scheirer [4]	51.86	37.85

Table 2. *Accuracy1* of the proposed method compared to best performing methods reported on ballroom/songs datasets.

datasets by providing the correct tempo class. Accuracies reported are 88% and 76% for *ballroom*/*songs* datasets respectively. Thus, for the *songs* dataset, even with prior knowledge of the tempo class, periodicity analysis and peak-picking are not always adequate.

Table 2 shows the performance of the proposed method compared to the baseline method adopted and the best performing algorithms reported in the literature for both datasets. It is evident that the proposed method outperforms all other methods. It should be mentioned that although Seyerlehner's SE1 [8] performs better in *ballroom* dataset, results are not directly comparable because they adopt a leave-one-out cross validation. Moreover SE1 reports very low accuracy for *songs* dataset. The significant performance increase of our method is somewhat expected, since it incorporates prior knowledge of the datasets. Although the cross-fold validation strategy splits data to independent subsets, there is still some prior information propagated to test sets caused by the uniformity of the datasets, i.e. most artists/styles are always present in both train/test sets. However the proposed method offers a promising approach to handle large datasets.

7. DISCUSSION AND FUTURE WORK

We presented a method for learning tempo classes with Support Vector Machines. Tempo class classification accuracies of 75% were achieved for both datasets, while most errors were made for excerpts close to class boundaries. The limitation of the target tempi decision space accordingly to the tempo class found for a given excerpt, reduced octave errors made by a baseline tempo estimation system significantly. Estimation accuracies where increased by a margin of 16% and 5% for *ballroom*/*songs* datasets respectively.

It must be noted that classification errors are propagated to the final tempo decision, especially for excerpts that have tempo close to the tempo class decision boundaries. A softer classification decision may be more sensible, as for example providing a confidence measure instead of a hard decision. Moreover, a different treatment of the periodicity function such as analyzing metrical levels considering knowledge of music speed may be proved more efficient. These two main aspects of the proposed method would be investigated in future research.

8. REFERENCES

[1] Peeters G., "Template-based estimation of time-varying tempo," in *EURASIP Journal on Applied Signal Processing,* Volume 2007 Issue 1, 2007 .

[2] Klapuri A., Eronen A. and Astola J., "Analysis of the Meter of Music Acoustic Signals," *IEEE Trans. on Audio, Speech and Language Processing*, Vol. 14(1), 2006.

[3] Gkiokas A., Katsouros V., Carayannis G. and Stafylakis T., "Music Tempo Estimation and Beat Tracking by Applying Source Separation and Metrical Relations," in *Proc. of the 37th IEEE ICASSP*, Kyoto, Japan, March 25-30, 2012.

[4] Gouyon F., Klapuri A., Dixon S., Alonso M., Tzanetakis G., Uhle C., and Cano P., "An Experimental Comparison of Audio Tempo Induction Algorithms," in *IEEE Transactions on Audio, Speech, and Language Processing,* Vol. 14(5) , September 2006.

[5] Gouyon F. and Dixon S., "A Review of Automatic Rhythm Description Systems," *Computer Music Journal, 29:1, pp 34-54,* Spring 2005.

[6] Dixon S., "Automatic Extraction of Tempo and Beat from Expressive Performances," *J. New Music Research*, 30(1):39–58, 2001.

[7] Alonso M., Richard G. and David B., "Accurate Tempo Estimation Based on Harmonic + Noise Decomposition," *EURASIP Journal on Applied Signal Processing,* Volume 2007, Issue 1, January 2007

[8] Seyerlehner K., Widmer G., and Schnitzer D., "From Rhythm Patterns to Perceived Tempo," in *Proc. of ISMIR,* Vienna, Austria, 2007.

[9] Peeters G., "Template-Based Estimation of Tempo: Using Unsupervised or Supervised Learning to Create Better Spectral Templates," in *Proc. of DAFx-10*, Graz, Austria, 2010.

[10] Eronen A. and Klapuri A., "Music Tempo Estimation with k-NN Regression," in *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 18, No. 1, January 2010.

[11] Hockman, J.A. and I. Fujinaga. "Fast vs slow: Learning tempo octaves from user data," in *Proc. of ISMIR*, Utrecht, Netherlands, 2010.

[12] Smith L.M., "Beat-Critic: Beat-Tracking Octave Error Identification by Metrical Profile Analysis," in *Proc. of ISMIR*, Utrecht, Netherlands, 2010.

[13] Mandel M.I., Poliner G.E. and Ellis D.P.W., "Support vector machine active learning for music retrieval," *Multimedia systems*, 12(1):1-11, August 2006.

[14] Wack N., Laurier C., Meyers O., Marxer R., Bogdanov D., Serrà J., Gómez E. & Herrera P., "Music Classification Using High-Level Models," in *Proc. of ISMIR,* Kobe, Japan, 2009.

[15] FitzGerald D., "Harmonic/Percussive Separation Using Median Filtering," in *Proc. of DAFx-10*, Graz, Austria, 2010.

[16] Holzapfel A. and Stylianou Y., "Scale transform in rhythmic similarity of music," in *IEEE Trans. on Audio, Speech and Language Processing*, Vol. 19(1), 2011.

[17] Peeters G., "Spectral and Temporal Periodicity Representations of Rhythm for the Automatic Classification of Music Audio Signal," in *IEEE Trans. on Audio, Speech and Language Processing*, Vol. 19 (5), 2011

[18] Hsu C.W. and Lin C.J., "A comparison of methods for multiclass support vector machines," in *IEEE Transactions on Neural Networks,* Vol. 13(2), March 2002.

[19] Chang C.-C. and Lin C.-J., "LIBSVM : a Library for Support Vector Machines," *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27, 2011.

CONTEXT-FREE 2D TREE STRUCTURE MODEL OF MUSICAL NOTES FOR BAYESIAN MODELING OF POLYPHONIC SPECTROGRAMS

Hirokazu Kameoka[1,2], **Kazuki Ochiai**[1], **Masahiro Nakano**[2], **Masato Tsuchiya**[1], **Shigeki Sagayama**[1]
[1]Graduate School of Information Science and Technology, The University of Tokyo
Hongo 7-3-1, Bunkyo, Tokyo, 113-8656, Japan
[2]NTT Communication Science Laboratories, NTT Corporation
Morinosato Wakamiya 3-1, Atsugi, Kanagawa, 243-0198, Japan

ABSTRACT

This paper proposes a Bayesian model for automatic music transcription. Automatic music transcription involves several subproblems that are interdependent of each other: multiple fundamental frequency estimation, onset detection, and rhythm/tempo recognition. In general, simultaneous estimation is preferable when several estimation problems have chicken-and-egg relationships. This paper proposes modeling the generative process of an entire music spectrogram by combining the sub-process by which a musically natural tempo curve is generated, the sub-process by which a set of note onset positions is generated based on a 2-dimensional tree structure representation of music, and the sub-process by which a music spectrogram is generated according to the tempo curve and the note onset positions. Most conventional approaches to music transcription perform note extraction prior to structure analysis, but accurate note extraction has been a difficult task. By contrast, thanks to the combined generative model, the present method performs note extraction and structure estimation simultaneously and thus the optimal solution is obtained within a unified framework. We show some of the transcription results obtained with the present method.

1. INTRODUCTION

Music transcription is the process of automatically converting a given audio signal into a musical score. Although there are a number of viable ways of transcribing monophonic music, polyphonic music still poses a formidable challenge.

Several subproblems must be solved if we are to transcribe polyphonic music automatically, namely source separation, multiple fundamental frequency estimation (the estimation of the fundamental frequencies of concurrent musical sounds), onset detection (the detection of the position in the signal where each note begins), and rhythm recognition (the estimation of the tempo, beat locations, and the note value of each note). The difficulty is that these subproblems involve many ambiguities.

An audio signal of a musical note typically consists of many overtones, some of which usually overlap when multiple notes are played simultaneously. To detect which notes are present at a certain time instant, we need to know which musical note each frequency component belongs to. Since this information is missing for the spectrum of a mixture signal, there can be multiple interpretations of how the spectrum of each sound should appear as well as which pitches are present in the mixture. On the other hand, a music performance often involves temporal fluctuation in terms of both rhythm and tempo, which means performers do not always play notes with a perfectly timed rhythm and constant tempo. Since we cannot define a note value without having a notion for tempo and vice versa, there can be infinite interpretations regarding what the intended rhythm was and how the tempo varied if both types of information are missing.

Many methods have already been developed for polyphonic music transcription, most of which try to tackle the problem by dealing with the abovementioned subproblems separately [1]. However, the inherent difficulty of the music transcription problem lies in the chicken-and-egg interdependency between these subproblems [2]. Firstly, if the given signal is already decomposed into individual notes, it is a simple matter to detect their fundamental frequencies. On the other hand, the decomposition of a given spectrogram into individual notes can be accomplished more accurately when the fundamental frequencies of the concurrent sounds are given. Also, if we know the fundamental frequencies of all the underlying notes in the signal, they can constitute very useful information for accurately estimating their onset times and vice versa. Furthermore, as the onset times of notes are usually governed by the rhythmic structure of a piece of music, the "chicken and egg" situation also applies to the detection of note onsets and the determination of beat locations and tempo. If we know the beat locations of a piece of music, then it is much easier to detect the onset times of notes and vice versa, since the inter-onset times are likely to be multiples or fractions of the beat period.

Simultaneous estimation is generally preferable when several estimation problems are interdependent. Thus, we consider it necessary to introduce a unified model, which could be used to jointly solve the problems of determining the pitch and onset time of each musical note, the rhythm and the overall tempo variation of a piece of music. In this paper, we take a Bayesian approach (a generative model approach) as in [3–6] to formulate and solve this simultaneous estimation problem.

2. GENERATIVE MODEL OF SPECTROGRAM

2.1 Overview

Motivated by the above, this paper proposes modeling the generative process of an entire spectrogram of a piece of music by formulating the following three sub-processes and combining them into one process: (1) the sub-process by which the tempo curve of a piece of music is generated, (2) the sub-process by which a set of note onset positions (in terms of the relative time) is generated based on a 2-dimensional tree structure representation of music, and (3) the sub-process by which a music spectrogram is generated according to the tempo curve generated by sub-process 1 and the set of note onset positions generated by sub-process 2. In the following, we model sub-process 1 in 2.2, sub-process 2 in 2.3 and sub-process 3 in 2.4, respectively. Our aim is to use this complete generative model to explain how a given spectrogram is generated. The most likely model parameters given the observation would then give a musically likely interpretation of what is actually happening in the spectrogram (*i.e.*, a musical score). To this end, we employ a Bayesian approach to infer the posterior distributions of all the latent parameters. An approximate posterior inference algorithm is derived, which is described in Section 3.

2.2 Sub-process for generating tempo curve

The tempo of a piece of music is not always constant and in most cases it varies gradually over time. If we use a [1] "tick" as a relative time notion, an instantaneous (or local) tempo may be defined as the length of 1 tick in seconds. Now let us use $\ _d$ to denote the real duration (in units of seconds) corresponding to the interval between d and $d+1$ ticks. Thus, $\ _d$ corresponds to the local tempo and so the sequence $\ _1,\dots,\ _D$ can be regarded as the overall tempo curve of a piece of music. One reasonable way to ensure a smooth overall change in tempo is to place a Markov-chain prior distribution over the sequence $\ _1,\dots,\ _D$ that is likely to generate a sequence $\ _1,\dots,\ _D$ such that $\ _1 \simeq\ _2,\ _2 \simeq\ _3,\dots,\ _{D\ 1} \simeq\ _D$. Here, we assume a Gaussian-chain prior for convenience:

$$\ _d|\ _{d\ 1} \sim \mathcal{N}(\ _d;\ _{d\ 1},(\sigma^\mu)^2)\ \ (d=2,\dots,D), \quad (1)$$

where $\mathcal{N}(x;\ ,\sigma) \propto e^{\ \frac{(x\ \ \mu)^2}{2\sigma^2}}$. If we use ψ_d to denote the absolute time (in units of seconds) corresponding to d ticks, ψ_d can thus be written as $\psi_d = \psi_{d\ 1} +\ _{d\ 1}$, which plays the role of mapping a relative time in units of ticks (integer) to an absolute time in units of seconds (continuous value).

2.3 Sub-process for generating note onset positions

Here we describe the generative model of the set of some number R of note onset positions $S_1,\dots,S_R$ (in units of ticks). Most people would probably agree that music has a 2-dimensional hierarchical structure. Frequent motifs, phrases or melodic themes consist of a hierarchy that can be described as time-span trees. In addition, polyphony often consists of multiple independent voices. That is, we can assume that music consists of a time-spanning tree structure and a synchronizing structure of multiple events at several levels of a hierarchy. We would like to describe this 2-dimensional tree structure representation of music in the form of a generative model. This can be accomplished by introducing a generative model that is conceptually similar to the one proposed in [6].

[1] Tick is a relative measure of time represented by the number of discrete divisions a quarter note has been split into. So, if we consider 16 divisions per quarter note, for instance, the duration of 40 ticks corresponds to two-and-a-half beats.

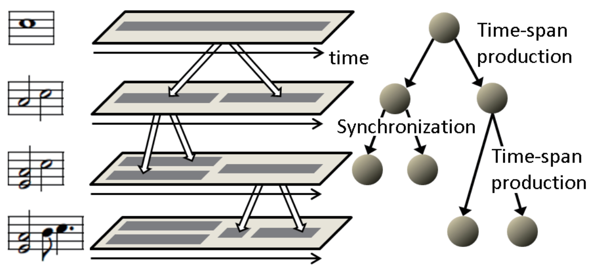


Figure 1. Generative model of a 2-dimensional tree structure representation of musical notes.

Fig. 1 shows an example of the generative process of four musical notes in one bar of 4/4. In this example, a whole note is first split into two consecutive half notes. We call this process "time-spanning." Next, the former half note is copied in the same location, thus resulting in a chord of two half notes. We call this process "synchronization." A chord with an arbitrary number of notes can thus be generated by successively employing this type of binary production. Finally, the latter half note is split into a quaver and a dotted quarter note via the time-spanning process. This kind of generative process can be modeled by extending the idea of the probabilistic context-free grammar (PCFG) [7]. For simplicity, this paper focuses only on Chomsky normal form grammars, which consist of only two types of rules: emissions and binary productions. A PCFG is a pair consisting of a context-free grammar (a set of symbols and productions of the form $A \rightarrow BC$ or $A \rightarrow w$, where A, B, and C are called "nonterminal symbols" and w is called a "terminal symbol") and production probabilities, and defines a probability distribution over the trees of symbols. The parameters of each symbol consist of (1) a distribution over rule types, (2) an emission distribution over terminal symbols, and (3) a binary production over pairs of symbols.

To describe the generative process shown in Fig. 1, we must introduce an extension of PCFG. As we explain later, we explicitly incorporate a process of stochastically choosing either "time-spanning" or "synchronization" in the binary production process. Fig. 2 defines the proposed generative process of the set of the onset positions of some number R of musical notes. In our model, each node n of the parse tree corresponds to one musical note (with no pitch information) and a pair consisting of the onset position S_n and duration L_n of that note is considered to be a nonterminal symbol. We first draw a "switching" distribution (namely, a Bernoulli distribution) ϕ^{T} over the two rule types {EMISSION, BINARY-PRODUCTION} from a Beta distribution. Next, we draw another "switching" dis-

```
Draw rule probabilities:
  $\phi^T \sim \mathrm{Beta}(\phi^{\mathrm{T}}; 1, \quad ^{\mathrm{T}})$
  [Probability of choosing either of two rule types]
  $\phi^N \sim \mathrm{Beta}(\phi^N; 1, \quad ^N)$
  [Probability of choosing either of two binary-production types]
  For each duration $l$:
    $\boldsymbol{\phi}_l^B \sim \mathrm{Dirichlet}(\boldsymbol{\phi}_l^B; \boldsymbol{\beta}_l^B)$
    [Probability of position at which segment of length $l$ is split]
For each node $n$ in the parse tree:
  $b_n \sim \mathrm{Bernoulli}(b_n; \phi^{\mathrm{T}})$
  [Choose either EMISSION or BINARY-PRODUCTION]
  If $b_n =$ EMISSION
    $S_r \sim \quad _{S_r,S_n}, \; L_r \sim \quad _{L_r,L_n}$
    [Emit terminal symbol]
  If $b_n =$ BINARY-PRODUCTION
    $\quad _n \sim \mathrm{Bernoulli}( \quad _n; \phi^{\mathrm{N}})$
    [Choose either SYNCHRONIZATION or TIME-SPANNING]
    If $\quad _n =$ SYNCHRONIZATION
      $S_{n_1} \sim \quad _{S_{n_1},S_n}, \; S_{n_2} \sim \quad _{S_{n_2},S_n}$
      $L_{n_1} \sim \quad _{L_{n_1},L_n}, \; L_{n_2} \sim \quad _{L_{n_2},L_n}$
      [Produce two copies of note $n$]
    If $\quad _n =$ TIME-SPANNING
      $S_{n_1} \sim \quad _{S_{n_1},S_n}, \; S_{n_2} \sim \quad _{S_{n_2},S_n+L_{n_1}}$
      $L_{n_1} \sim \quad _{L_{n_1},L_n \quad L_{n_2}}$
      $L_{n_2} \sim \mathrm{Discrete}(L_{n_2}; \boldsymbol{\phi}^{\mathrm{B}}_{L_n})$
      [Split note $n$ into two consecutive notes $n_1$ and $n_2$]
```

Figure 2. The probabilistic specification of the present generative model of a 2-dimensional tree structure representation of musical notes. denotes Kronecker's delta. Thus, $x \sim \quad _{x,y}$ means $x = y$ (with probability 1). $\mathrm{Bernoulli}(x;y)$ and $\mathrm{Beta}(y;\boldsymbol{z})$ are defined as $\mathrm{Bernoulli}(x;y) = y^x(1 \quad y)^{1 \quad x}$ and $\mathrm{Beta}(y;\boldsymbol{z}) \propto y^{z_1 \quad 1}(1 \quad y)^{z_2 \quad 1}$, where $x \in \{0,1\}$, $0 \quad y \quad 1$ and $\boldsymbol{z} = (z_1, z_2)$, respectively. $\mathrm{Discrete}(x;\boldsymbol{y})$ and $\mathrm{Dirichlet}(\boldsymbol{y};\boldsymbol{z})$ are defined as $\mathrm{Discrete}(x;\boldsymbol{y}) = y_x$ and $\mathrm{Dirichlet}(\boldsymbol{y};\boldsymbol{z}) \propto \prod_i y_i^{z_i \quad 1}$ where $\boldsymbol{y} = (y_1, \ldots, y_I)$ with $y_1 + \quad + y_I = 1$ and $\boldsymbol{z} = (z_1, \ldots, z_I)$, respectively.

tribution ϕ^{N} over the two binary-production types {TIME-SPANNING, SYNCHRONIZATION} similarly from a Beta distribution. Finally, we generate a discrete distribution $\boldsymbol{\phi}_l^{\mathrm{B}} = (\phi_{l,1}^{\mathrm{B}}, \ldots, \phi_{l,l}^{\mathrm{B}})$ over the position l' at which the segment of duration l is split when BINARY-PRODUCTION is chosen. The shapes of all the Beta distributions and the Dirichlet distribution in our model are governed by concentration hyperparameters: $\quad ^{\mathrm{T}}, \quad ^{\mathrm{N}}$ and $\boldsymbol{\beta}_1^{\mathrm{B}}, \ldots, \boldsymbol{\beta}_D^{\mathrm{B}}$.

Given a grammar, we generate a parse tree in the following manner: start with a root node that has the designated root symbol, $S_{\mathrm{Root}} = 0$ and $L_{\mathrm{Root}} = D$ where D denotes the overall length of a piece of music in ticks. For each nonterminal node n, we first choose a rule type b_n using ϕ^{T}. If $b_n =$ EMISSION, we produce a terminal symbol S_r with the value of S_n, namely the onset position of note r. If $b_n =$ BINARY-PRODUCTION, we then choose a binary-production type $\quad _n$ using ϕ^{N}. If $\quad _n =$ SYNCHRONIZATION, we produce two nonterminal children n_1 and n_2 such that $S_{n_1} = S_{n_2} = S_n$, $L_{n_1} = L_{n_2} = L_n$. This means that the notes of the child nodes have exactly the same onset and duration. If $\quad _n =$ TIME-SPANNING, we produce two nonterminal children n_1 and n_2 with $S_{n_1} = S_n$, $L_{n_1} = L_n \quad L_{n_2}$, $S_{n_2} = S_n + L_{n_1}$ where L_{n_2} is drawn from a discrete distribution $\boldsymbol{\phi}^{\mathrm{B}}_{L_n}$. L_{n_2} corresponds to the position at which the segment of duration L_n is divided. We apply the procedure recursively to any nonterminal children and finally obtain a sequence $S_1, \ldots, S_R$ corresponding to the onset positions of R musical notes.

None of the notes r yet contains pitch information. We assign a pitch index κ_r to each note r in the same way as an ordinary cluster assignment process:

$$\boldsymbol{\phi}_r^{\mathrm{K}} \sim \mathrm{Dirichlet}(\boldsymbol{\phi}_r^{\mathrm{K}}; \boldsymbol{\alpha}^{\mathrm{K}}), \tag{2}$$

$$\kappa_r \sim \mathrm{Discrete}(\kappa_r; \boldsymbol{\phi}_r^{\mathrm{K}}), \tag{3}$$

where $\mathrm{Discrete}(x;\boldsymbol{y}) = y_x$ (where $\boldsymbol{y} = (y_1, \ldots, y_I)$ with $y_1 + \quad + y_I = 1$) and $\mathrm{Dirichlet}(\boldsymbol{y};\boldsymbol{z}) \propto \prod_i y_i^{z_i \quad 1}$ (where $\boldsymbol{z} = (z_1, \ldots, z_I)$). The k-th element of $\boldsymbol{\phi}_r^{\mathrm{K}}$ defines how likely each pitch index is to be chosen. It should be noted here that the generative processes of S_r and κ_r should not be considered independently, since harmony and rhythm are in general interdependent of each other. An interesting direction for future work is the joint modeling of these two generative processes.

2.4 Sub-process for generating spectrogram

We now turn to describing the sub-process by which a music spectrogram is generated. Here, we consider that a music spectrogram is generated according to the tempo curve and the set of note onset positions, that have been generated by the sub-processes described in 2.2 and 2.3. To model a spectrogram of a musical audio signal, we make the following assumptions about musical notes:

(A1) Each musical note has a static spectral profile characterized by a particular pitch.

(A2) The magnitude spectrum of music at a certain time instant is represented by a superposition of the spectra of multiple musical notes.

(A3) The power of each musical note varies smoothly in time in the interval between the onset and offset.

From assumption (A1), a magnitude spectrogram of a musical note r can be described as

$$X_{\omega,t} = \sum_{r=1}^{R} H_{\omega,\kappa_r} W_{r,t}, \tag{4}$$

where ω and t are frequency and time indices, respectively. A set consisting of $H_{1,k}, \ldots, H_{\Omega,k} \quad 0$ represents the static spectrum of the k-th pitch and so a set consisting of $H_{1,\kappa_r}, \ldots, H_{\Omega,\kappa_r}$ signifies the spectrum of note r. $W_{r,t} \quad 0$ denotes the power of note r at time t. As the assumptions (A1) and (A2) do not always hold exactly in reality, an actual music spectrogram $Y_{\omega,t}$ may diverge from the "ideal model" $X_{\omega,t}$ to some extent. One way to simplify this kind of deviation process is to assume a probability distribution

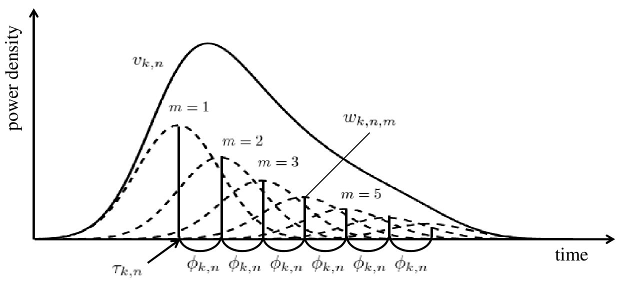

power density
$v_{k,n}$
$m=1$
$m=2$
$m=3$
$w_{k,n,m}$
$m=5$
time
$\tau_{k,n}$
$\phi_{k,n}$ $\phi_{k,n}$ $\phi_{k,n}$ $\phi_{k,n}$ $\phi_{k,n}$ $\phi_{k,n}$

Figure 3. Power envelope $W_{r,t}$ of musical note r.

on $Y_{\omega,t}$ with the expected value of $X_{\omega,t}$. Here, we assume that $Y_{\omega,t}$ follows a Poisson distribution with mean $X_{\omega,t}$:

$$Y_{\omega,t} \sim \text{Poisson}(Y_{\omega,t}; X_{\omega,t}), \tag{5}$$

where $\text{Poisson}(y; x) = x^y e^{\ \ x}/y!$. It should be noted that the maximization of the Poisson likelihood with respect to $X_{\omega,t}$ amounts to optimally fitting $X_{\omega,t}$ to $Y_{\omega,t}$ by using the I-divergence as the fitting criterion [3, 8]. To avoid any indeterminacy in the scaling of H_{ω,κ_r} and $W_{r,t}$, we assume

$$\sum_{\omega} H_{\omega,k} = 1 \quad (k = 1, \ldots, K). \tag{6}$$

Each spectral profile $H_{\omega,k}$ must have the harmonic structure of a particular pitch. One way of ensuring this is to assume a prior distribution over $H_{\omega,k}$ so that it is likely to generate a spectrum with a certain harmonic structure of the k-th pitch. Here, we choose to place a Gamma prior over $H_{\omega,k}$, namely

$$H_{\omega,k} \sim \text{Gamma}(H_{\omega,k};\ \ \bar{H}_{\omega,k} + 1,\ \ ), \tag{7}$$

where $\text{Gamma}(x; a, b) \propto x^{a\ \ 1} e^{\ \ bx}$. The mode of this prior distribution is given by $\bar{H}_{\omega,k}$, which should be defined such that it corresponds to the most likely spectral profile for the k-th pitch.   determines the peakiness of the density around the mode.

To incorporate assumption (A3) into $W_{r,t}$, we propose describing $W_{r,t}$ using a parametric model expressed as a sum of Gaussians [8] (Fig. 3):

$$W_{r,t} = \sum_{m=1}^{M} G_{r,m,t}, \tag{8}$$
$$G_{r,m,t} = \frac{w_r u_{r,m}}{\sqrt{2\ \ }\ \varphi} e^{\ \ (t\ \ (m\ \ 1)\varphi\ \ \ \ r)^2/2\varphi^2},$$

where w_r is the total energy of note r, and ${}_r$ is the center of the first Gaussian, which can be considered the onset time of note r (in seconds). The centers of the Gaussians are constrained so that they are equally spaced with the distance φ, which is equal to the "standard deviation" of all the Gaussians. $u_{r,1}, \ldots, u_{r,M}$ are weights associated with the M Gaussians, which determine the overall shape of the power envelope. To avoid any indeterminacy in the scaling of w_r and $u_{r,m}$, we assume

$$\forall_r : \sum_{m=1}^{M} u_{r,m} = 1. \tag{9}$$

The number of consecutive Gaussians with non-zero weights corresponds to the duration of the note, which we hope to infer automatically from an observed spectrogram. To this end, we choose to use a stick-breaking representation [9] to describe the generative process of $u_{r,1}, \ldots, u_{r,M}$:

$$V_{r,m} \sim \text{Beta}(V_{r,m}; 1,\ \ {}^{\mathrm{V}}_{r}) \tag{10}$$
$$u_{r,m} = V_{r,m} \prod_{m'=1}^{m\ \ 1} (1\ \ \ V_{r,m'}), \tag{11}$$

which contributes to sparsifying the Gaussian weights.

Now, recall that the onset position S_r (in ticks) of note r is assumed to have been generated via the generative process described in 2.3. The onset position ${}_r$ of note r should thus be placed near the absolute time into which S_r is converted. Recall also that ψ_d, which can be considered a function that takes a relative time d as an input and returns the corresponding absolute time as an output, is also assumed to have been generated (via the generative process described in 2.2). Given S_r and ψ_d, we find it convenient to write the generative process of ${}_r$ as

$${}_r \sim \mathcal{N}(\ \ {}_r; \psi_{S_r}, (\sigma\ \ )^2). \tag{12}$$

2.5 Expansion of generative process

We can describe an expanded version of the generative process of $Y_{\omega,t}$ as

$$C_{r,m,\omega,t} \sim \text{Poisson}(C_{r,m,\omega,t}; H_{\omega,\kappa_r} G_{r,m,\omega,t})$$
$$Y_{\omega,t} \sim\ \ \big(Y_{\omega,t}\ \ \sum_{r,m} C_{r,m,\omega,t}\big), \tag{13}$$

by introducing an auxiliary variable $C_{r,m,\omega,t}$. For convenience of analysis, we use this generative process instead of (5) in the following. Note that it can be readily verified that marginalizing out $C_{r,m,\omega,t}$ reduces (13) to (5).

3. APPROXIMATE POSTERIOR INFERENCE

3.1 Variational Bayesian approach

In this section, we describe an approximate posterior inference algorithm for our generative model based on variational inference. The random variables of interest in our model are

$H = \{H_{\omega,k}\}_{\omega,k}$	:	spectrum of pitch k,
$w = \{w_r\}_r$	:	total energy of note r,
$V = \{V_{r,m}\}_{r,m}$	:	shape of power envelope of note r,
$= \{\ \ {}_r\}_r$	:	onset time (sec) of note r,
$\kappa = \{\kappa_r\}_r$	:	pitch index assigned to note r,
$\psi = \{\psi_d\}_d$	:	absolute time corresponding to d ticks,
$= \{\ \ {}_d\}_d$	:	local tempo between d and $d+1$ ticks,
$S = \{S_r\}_r$	:	onset position of note r (in ticks),
$L = \{L_r\}_r$	:	duration of note r (in ticks), and
$\phi^{\mathrm{B}}, \phi^{\mathrm{T}}, \phi^{\mathrm{N}}, \phi^{\mathrm{K}}$	:	rule probabilities,

which we denote as Θ. Our goal is to compute the posterior $p(\Theta, C|Y)$ where $Y = \{Y_{\omega,t}\}$ and $C = \{C_{r,m,\omega,t}\}$ are sets consisting of observed magnitude spectra and auxiliary variables, respectively. By using the conditional distributions defined in 2.2, 2.3, 2.4, and 2.5, we can write the joint distribution $p(Y, \Theta, C)$ as

$$\begin{aligned}
&p(Y, H, w, V,\ \ , \kappa, \psi,\ \ , S, L, \boldsymbol{\phi}^B, \phi^T, \phi^N, \boldsymbol{\phi}^K, C) \\
&= p(Y|C)p(C|H, w, V,\ \ , \kappa)p(H)p(V)p(w) \\
&\quad p(\ \ |\psi, S)p(\psi|\ \ )p(\ \ )p(\kappa|\boldsymbol{\phi}^K)p(\boldsymbol{\phi}^K) \\
&\quad p(S, L|\boldsymbol{\phi}^B, \phi^T, \phi^N)p(\boldsymbol{\phi}^B)p(\phi^T)p(\phi^N), \qquad (14)
\end{aligned}$$

but to obtain the exact posterior $p(\Theta, C|Y)$, we need to compute $p(Y)$, which involves many intractable integrals.

We can express this posterior variationally as the solution to an optimization problem:

$$\underset{q \in \mathcal{Q}}{\operatorname{argmin}}\ \mathrm{KL}(q(\Theta, C) \| p(\Theta, C|Y)), \qquad (15)$$

where KL(‖) denotes the Kullback-Leibler (KL) divergence between its two arguments. Indeed, if we let $\mathcal{Q}$ be the family of all distributions over Θ and C, the solution to the optimization problem is the exact posterior $p(\Theta, C|Y)$, since KL divergence is minimized exactly when its two arguments are equal. Of course, solving this optimization problem is just as intractable as directly computing the posterior. Although it may appear that no progress has been made, having a variational formulation allows us to consider tractable choices of $\mathcal{Q}$ in order to obtain principled approximate solutions.

For our model, we define the set of approximate distributions $\mathcal{Q}$ as those that factor as follows:

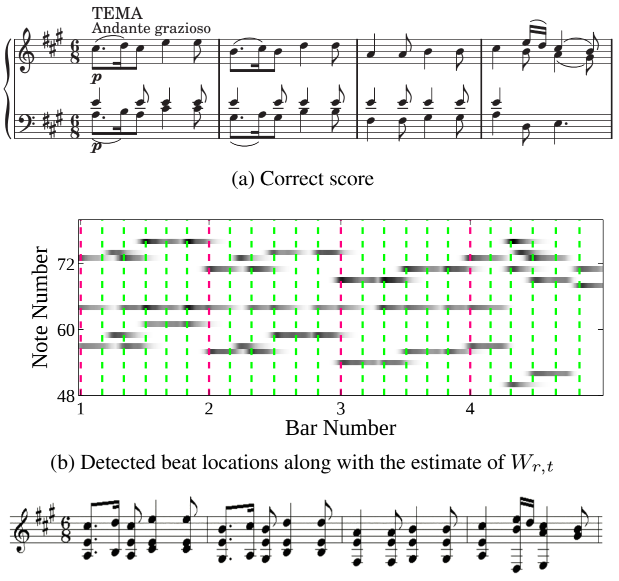
$$\begin{aligned}
\mathcal{Q} = \big\{q : q(C)q(H)q(w)q(V)q(\ \ , \psi,\ \ )q(\kappa) \\
q(S, L)q(\phi^{\mathrm{K}})q(\phi^{\mathrm{B}})q(\phi^{\mathrm{T}})q(\phi^{\mathrm{N}})\big\}. \qquad (16)
\end{aligned}$$

We admit that this is a strong assumption. Its validity and how it affects the parameter inference result must be investigated in the future.

3.2 Coordinate ascent

We now present an algorithm for solving the optimization problem described in (15) and (16). Unfortunately, the optimization problem is non-convex, and it is intractable to find the global optimum. However, we can use a simple coordinate ascent algorithm to find a local optimum. The algorithm optimizes one factor in the mean-field approximation of the posterior at a time while fixing all the other factors. The mean-field update equations for the variational distributions are given in the following form:

$$\begin{aligned}
q(\boldsymbol{C}_{\omega,t}) &= \mathrm{Multinomial}(\boldsymbol{C}_{\omega,t}; Y_{\omega,t}, \boldsymbol{f}^C_{\omega,t}), &(17)\\
q(H_{\omega,k}) &= \mathrm{Gamma}(H_{\omega,k};\ {}^H_{\omega,k}, \zeta^H_{\omega,k}), &(18)\\
q(w_r) &= \mathrm{Gamma}(w_r;\ {}^w_r, \zeta^w_r), &(19)\\
q(V_{r,m}) &= \mathrm{Beta}(V_{r,m};\ {}^V_{r,m}, \zeta^V_{r,m}), &(20)\\
q(\boldsymbol{\tau}, \boldsymbol{\psi}, \boldsymbol{\mu}) &= \mathcal{N}(\boldsymbol{\chi};\ {}^{\chi}, \boldsymbol{\zeta}^{\chi}), &(21)\\
q(\kappa_r) &= \mathrm{Discrete}(\kappa_r; \boldsymbol{f}^{\kappa}_r), &(22)\\
q(\boldsymbol{\phi}^K_r) &= \mathrm{Dirichlet}(\boldsymbol{\phi}^K_r;\ {}^K_r), &(23)\\
q(S_r, L_r) &= \mathrm{Discrete}(S_r, L_r; \boldsymbol{f}^{SL}_r), &(24)\\
q(\boldsymbol{\phi}^B_l) &= \mathrm{Dirichlet}(\boldsymbol{\phi}^B_l;\ {}^B_l), &(25)\\
q(\phi^T) &= \mathrm{Beta}(\phi^T;\ {}^T, \zeta^T), &(26)
\end{aligned}$$



(a) Correct score

(b) Detected beat locations along with the estimate of $W_{r,t}$

(c) Score transcribed with the proposed method

Figure 4. Transcription result obtained with the proposed method applied to Morzart: Piano Sonata No. 11 in A major, K. 331/300i under the situation where $_1, \ldots, _R$ are given. In (b), the red and green lines indicate the estimates of bar lines and the positions of beat locations obtained with the present method, respectively.

$$q(\phi^N) = \mathrm{Beta}(\phi^N;\ {}^N, \zeta^N), \qquad (27)$$

where

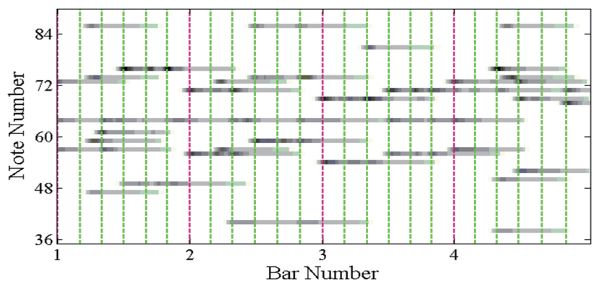
$$\boldsymbol{\chi} = \begin{bmatrix}\boldsymbol{\tau}\\ \boldsymbol{\psi}\\ \boldsymbol{\mu}\end{bmatrix},\quad {}^{\chi} = \begin{bmatrix}\boldsymbol{\eta}\\ \boldsymbol{\eta}^{\psi}\\ \boldsymbol{\eta}^{\mu}\end{bmatrix},\quad \boldsymbol{\zeta}^{\chi} = \begin{bmatrix}\boldsymbol{\nu} & \boldsymbol{\nu}\ ^{\psi} & \boldsymbol{\nu}\ ^{\mu}\\ \boldsymbol{\nu}\ ^{\psi} & \boldsymbol{\nu}^{\psi} & \boldsymbol{\nu}^{\psi\mu}\\ \boldsymbol{\nu}\ ^{\mu} & \boldsymbol{\nu}^{\psi\mu} & \boldsymbol{\nu}^{\mu}\end{bmatrix}.$$

(25)–(27) are performed only when we want to learn the rule probabilities. (24)–(27) can be updated using the inside-outside algorithm. The update formulas of the variational parameters are all given in analytical form, but they are omitted here owing to space limitations.

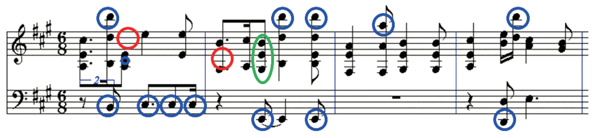
4. EXPERIMENTAL RESULTS

We now present experimental results obtained with our proposed model. We first conducted a preliminary experiment to confirm that our model can transcribe a score (appropriately estimate the note values of musical notes, beat locations, and the tempo of a music piece) when the onset times of all the musical notes (namely, $_r$'s) are given. We then show an example of transcription results obtained using the complete model directly from an audio spectrogram.

For the first experiment, we used a few piano recordings (RWC-MDB-C-2001 No. 26, 27, 30) excerpted from the RWC music database [12]. The data were the first 10 s, mixed down to a monaural signal and resampled to 16 kHz. The constant-Q transform was used to compute spectrograms where the time resolution, the lower bound of the frequency range, and the frequency resolution were set at 16 ms, 30 Hz and 12 cents, respectively. In this experiment, all the values $_1, \ldots, _R$ were given manually. The hyperparameters and initial parameters were set at $K = 74, M = 40, \varphi = 3,\ {}^H_{\omega,k} = {}^H_{\omega,k}\bar{H}_{\omega,k}+1,\ {}^H_{\omega,k} = 500,\ {}^w_r =$



(a) Detected beat locations along with the estimate of $W_{r,t}$



(b) Score transcribed with the proposed method

Figure 5. Transcription result obtained with the proposed method applied to Morzart: Piano Sonata No. 11 in A major, K. 331/300i. In (a), the red and green lines indicate the estimates of bar lines and the positions of beat locations obtained with the present method, respectively. In (b), the red, green and blue circles indicate the deletion errors, pitch errors and octave errors, respectively.

${}^{w}_{r} = 0$, ${}^{V}_{r,m} = 10e^{\ m/8}/\sum_{m'} e^{\ m'/8}, \sigma\ \ = 2, \sigma^{\psi} = 1, \sigma^{\mu} = 0.5$, ${}_{r,k} = 2$, ${}^{T} = 1$, ${}^{N} = 2$. The initial values of $H_{\omega,k}$ and $\bar{H}_{\omega,k}$ were set at the value obtained with the non-netaive matrix factorization [13] applied to the magnitude spectrogram of the piano excerpts from the RWC musical instrument sound database [11]. We set the resolution of the relative time at 4 ticks per quarter note. D and the initial values of ψ_d were set at the values obtained with [10]. The algorithm was run for 10 iterations. After convergence, we took the expected values of the posteriors and regarded them as the parameter estimates.

Fig. 4 shows an example of the score we obtained when we applied the present method to Mozart's Sonata (RWC-MDB-C-2001 No. 26). As can be seen from this example, the note values and the beat locations were appropriately estimated. We also confirmed that reasonably good results were obtained for other recordings such as Chopin's Nocturne No. 2 in E♭-maj, Op. 9 (RWC-MDB-C-2001 No. 30).

For the second experiment, we applied our method without providing any information about . The experimental conditions were the same as above except that we assumed that   was unknown. Fig. 5 shows an example of the estimates of $W_{r,t}$ (namely, the power envelope of note r) and the score obtained with the present method applied to the same data in Fig. 5. The result showed that many octave errors had occurred. This kind of error often occurs when there is a mismatch between a spectrum model and an actual spectrum. The validity of the assumptions we have made about the spectra of musical sounds in 2.4 must be carefully examined in the future.

5. CONCLUSION

This paper proposed a Bayesian model for automatic music transcription. Automatic music transcription involves several interdependent subproblems: multiple fundamental frequency estimation, onset detection, and rhythm/tempo recognition. To circumvent the chicken-and-egg problem, we modeled the generative process of an entire music spectrogram by combining the sub-process by which a musically natural tempo curve is generated, the sub-process by which a set of note onset positions is generated based on a 2-dimensional tree structure representation of music, and the sub-process by which a music spectrogram is generated according to the tempo curve and the note onset positions. Thanks to this combined generative model, the present method performs note extraction and structure estimation simultaneously and thus an optimal solution is obtained within a unified framework. We described some of the transcription results obtained with the present method.

6. REFERENCES

[1] N. Bertin, R. Badeau, and G. Richard, "Blind signal decompositions for automatic transcription of polyphonic music: NMF and K-SVD on the benchmark," In *Proc. ICASSP2007*, Vol. 1, pp. 65–68, 2007.

[2] K. Ochiai, H. Kameoka, and S. Sagayama, "Explicit beat structure modeling for non-negative matrix factorization-based multipitch analysis," in *Proc. ICASSP2012*, pp. 133–136, 2012.

[3] A. T. Cemgil, "Bayesian inference in non-negative matrix factorisation models," Technical Report CUED/F-INFENG/TR.609, University of Cambridge, 2008.

[4] M. D. Hoffman, D. M. Blei, and P. R. Cook, "Bayesian nonparametric matrix factorization for recorded music," in *Proc. ICML2010*, pp. 439–446.

[5] K. Yoshii, and M. Goto, "A nonparametric Bayesian multipitch analyzer based on infinite latent harmonic allocation," *IEEE Trans. Audio, Speech, Language Process.*, Vol. 20, No. 3, pp. 717–730, 2012.

[6] M. Nakano, Y. Ohishi, H. Kameoka, R. Mukai, and K. Kashino, "Bayesian nonparametric music parser," in *Proc. ICASSP2012*, pp. 461–464, 2012.

[7] P. Liang, S. Petrov, M. I. Jordan, and D. Klein, "The infinite PCFG using hierarchical Dirichlet processes," in *EMNLP2007*, pp. 688–697.

[8] H. Kameoka, T. Nishimoto, and S. Sagayama, "A multipitch analyzer based on harmonic temporal structured clustering," *IEEE Trans. on Audio, Speech, Language Process.*, Vol. 15, No. 3, pp. 982–994, 2007.

[9] J. Sethuraman, "A constructive definition of Dirichlet priors," *Statistica Sinica*, Vol. 4, pp. 639–650, 1994.

[10] D. P. W. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, 36(1):51–60, 2007.

[11] M. Goto, "Development of the RWC music database," In *Proc. the 18th International Congress on Acoustics (ICA 2004)*, pp. I-553–I-556, 2004.

[12] M Goto, H Hashiguchi, T Nishimura, and R Oka, "RWC music database: Popular, classical, and jazz music database," In *Proc. ISMIR*, pp. 287–288, 2002.

[13] P. Smaragdis and J. C. Brown, "Non-negative matrix factorization for music transcription," in *Proc. WASPAA2003*, pp. 177–180, 2003.

COMPRESSING MUSIC RECORDINGS INTO AUDIO SUMMARIES

Oriol Nieto
New York University
oriol@nyu.edu

Eric J. Humphrey
New York University
ejhumphrey@nyu.edu

Juan Pablo Bello
New York University
jpbello@nyu.edu

ABSTRACT

We present a criterion to generate audible summaries of music recordings that optimally explain a given track with mutually disjoint segments of itself. We represent audio as sequences of beat-synchronous harmonic features and use an exhaustive search to identify the best summary. To demonstrate the merit of this approach, we evaluate the criterion and show consistency across a collection of multiple recordings of different works. Finally, we present a fast algorithm that approximates the exhaustive search and allows us to automatically learn the hyperparameters of the algorithm for a given track.

1. INTRODUCTION

One of the classic motivations in the field of music informatics is facilitating the navigation of massive digital music collections by human users. Research in this area aims to develop computational methods of organizing and retrieving music recordings —tracks— in the spirit of reducing the amount of effort necessary to find desired content. Ultimately, the user must listen to any unfamiliar track to validate the search results, making the process considerably time consuming.

In digital music storefronts and other kinds of large collections, the traditional solution is to represent a full track with a single, identifiable excerpt. Known as audio thumbnailing, much effort has been invested into the development of automatic systems to these ends; for a partial review, we refer to [1, 2, 6, 7]. For some popular music that is highly repetitive in nature, these methods perform well in identifying useful thumbnails. Regardless, representing a full track with a single excerpt presents one unavoidable deficiency: the defining characteristics of a track are rarely concentrated in one specific section.

Recognizing this shortcoming, we motivate an alternative approach to classical thumbnailing that instead creates a short, listenable audio summary, capturing both the most unique and representative parts of a track. Specifically, this paper presents a novel audio summary criterion and an efficient method of automatically generating these summaries from real music recordings. The criterion is maximal for the set of segments that best explain the overall track while simultaneously exhibiting minimal overlap between them. Via examples and an experimental study we show how this measure yields good audio summaries. Furthermore, we show that it is possible to automatically select the optimal number and length of the selected subsequences specific to a given recording.

The remainder of this paper is organized as follows: Section 2 addresses the topic of feature representation. Section 3 defines the music summary criterion and showcases the measure in practice. Section 4 details a heuristic approximation to the exhaustive evaluation over the free parameters. Section 5 presents a systematic evaluation of the feature representation, heuristic solution and effect of automatically learning hyperparameters. Finally, we discuss our conclusions and observations for future work in Section 6.

2. FEATURE REPRESENTATION

The goal of developing an appropriate representation is to capture the information relevant to a given task while discarding unnecessary attributes. With this in mind, we describe the method of transforming time-domain audio signals into beat-synchronous sequences of harmonic features from which audio summaries can be identified.

2.1 Beat-Synchronicity

As a preprocessing stage, a recording is first analyzed by a beat tracking algorithm adapted from [3] for subsequent beat-synchronous feature extraction. In the interest of mitigating octave errors and producing consistent feature sequences across a variety of content, we impose constraints on the range of possible tempi the system can track. This is achieved by the following modification: periodicity analysis of the novelty function Δ_n is computed at $N\ log_2$ spaced frequencies per octave over the range $[1:8]$ Hz, producing the tempogram $\mathcal{T}$ as defined in [3]. This time-frequency representation is then wrapped to a single tempo octave of N bins and the most likely tempo path is extracted via the Viterbi decoder. In lieu of static transition probabilities, the transition probability matrix p_{trans} is defined as an identity matrix I of rank N convolved with a 1-D, 0-mean Gaussian window $\mathcal{N}$, where the standard deviation σ_n is parameterized by the relative amplitude of the maximum tempogram as a function of time n, as follows:

$$p_{trans}[n] = I_N * \mathcal{N}\left(\mu = 0, \sigma_n = \frac{max(|\mathcal{T}[n]|)}{\mu_{|\mathcal{T}[n]|}}\right) \quad (1)$$

This has the desirable effect of allowing the tempo estimator to adapt when the pulse strength is high, but resist change when the tempo becomes ambiguous. To find the best tempo octave to unwrap the path into, we analyzed a histogram of the chord durations contained in publicly available chord annotations [1]. Having found that approximately 95% of the chord durations are greater than 0.5 seconds in duration, we select 2Hz as a natural upper bound and map the optimal path through the single octave tempogram into the range of 60-120 BPM. At this stage, the remainder of the implementation follows the reference algorithm.

2.2 Harmonic Representations

Conventional approaches to harmonic analysis tasks in music informatics are predominantly built upon the use of chroma features, and we continue that tradition here. We also explore the use of tonal centroids, or Tonnetz features, as a mid-level harmonic representation. Introduced for the purpose of detecting harmonic change by Harte et al [4], the intuition for this decision is motivated as follows. First, typical distance metrics fail to capture musical significance between chroma vectors. In a pitch class representation, for example, the L_2 distance between a C major triad and a C♯ triad is equal to the distance between either triad and the notes B, B♭, and A. Additionally, chroma behaves like a mass function and it is not immediately apparent how to best measure the distance between these vectors. A Tonnetz representation, however, provides a geometric interpretation of pitch collections where distance is better defined as a musical and an Euclidean sense.

To compute both harmonic feature variants, we apply the constant-Q transform to a frame of audio over the range of 110–1760 Hz with 12 bins per octave, producing a pitch vector X. The length of the analysis window is determined by the longest filter, and is set to 0.45 seconds. Inspired by [5], a modified pitch vector Y is produced by standardizing the log-coefficients $log(\lambda X)$ and half-wave rectifying the result. The λ scale factor is heuristically set to 1000, but values within an order of magnitude in either direction produce similar results. Chroma features are derived from Y by wrapping onto a single octave and scaling by the L_2 norm, and Tonnetz features are computed identically to the method presented in [4].

2.3 Feature Quantization

It is computationally advantageous to quantize the feature space into a finite number of discrete values. We perform vector quantization by clustering the feature space via K-means and replacing each feature vector by its cluster's centroid. The pairwise distances between centroids are precomputed to accelerate distance calculations between symbolic feature sequences (see Section 3). Though larger values of K more faithfully reproduce the original features, this might result in an intractable process due to computation limitations as we see in subsection 3.3.

[1] https://github.com/tmc323/Chord-Annotations

3. DEFINING AN AUDIO SUMMARY CRITERION

Structure and repetition are fundamental characteristics of a musical work, and an audio summary should retain the minimum number of distinct parts that are necessary to describe it. Therefore, a good summary criterion actually synthesizes two opposing notions: we seek to lose as little information as possible, while avoiding overlap between chosen segments. A summary is defined as the set $\Gamma = [\gamma_1^N, \ldots, \gamma_P^N]$ of P, N-length subsequences that maximizes a function Θ over a feature sequence $\mathbf{S}$ of length M, where $\exists m$ s.t. $s_m^N = \gamma_i^N, m \in [1 : M]$, $s_m^N \in \mathbf{S}$, and $i \in [1 : P]$.

3.1 Compression Measure

The goal of describing a sequence in terms of itself with a minimal loss of information is fundamentally a data compression problem. Building upon this idea, we define a compression measure $\mathcal{C}(\Gamma|\mathbf{S})$ that quantifies the extent to which Γ explains a given $\mathbf{S}$, defined as follows:

$$\mathcal{C}(\Gamma|\mathbf{S}) = 1 - \frac{1}{PJ}\sum_{i=1}^{P}\sum_{m=1}^{J} ||\gamma_i^N, s_m^N||_2 \quad (2)$$

This measure can be interpreted as a normalized, convolutive Euclidean distance, such that there are $J = M - N + 1$ element-wise comparisons between a given N-length subsequence γ_i^N and all J N-length subsequences $s_m^N \in \mathbf{S}$. All distances, taken directly from the precomputed pairwise matrix discussed in Subsection 2.3, are then averaged over the J rotations and P subsequences in Γ. Intuitively, the compression measure equals 1 when $\Gamma = \mathbf{S}$ and 0 when $\Gamma \not\subseteq \mathbf{S}$.

3.2 Disjoint Information Measure

In addition to determining how well Γ describes $\mathbf{S}$, it is necessary to also measure the amount of information shared between each pair of subsequences in a set. Conversely, a disjoint information measure $\mathcal{I}(\Gamma)$ seeks to quantify the uniqueness of each subsequence in Γ relative to the rest, defined as follows:

$$\mathcal{I}(\Gamma) = \left(\prod_{i=1}^{P}\prod_{j=i+1}^{P} D_{min}(\phi(\gamma_i^N), \phi(\gamma_j^N))\right)^{\frac{2}{P(P-1)}} \quad (3)$$

We achieve shift-invariance by mapping a sequence of features γ_i^N to a sequence of *shingles* ρ_i^K with length $K = N - L + 1$ where a shingle is defined as the stacking of L adjacent feature frames into a single feature vector. The function ϕ returns the *shingled* version of a subsequence. A modified Euclidean distance function D_{min} then measures the intersection between sequences of shingles, returning the average minimum distance between the u^{th} shingle in

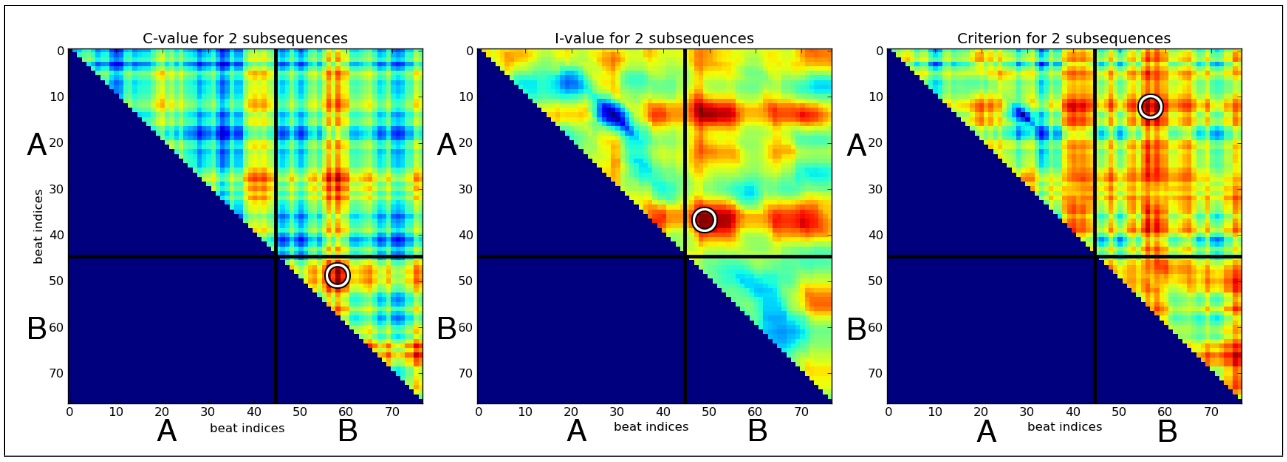


Figure 1. Search space for $\mathcal{C}$, $\mathcal{I}$ and Θ (left, middle, and right respectively) for $P = 2$ subsequences in the first half of a performance of the Mazurka Op. 30 No. 2. Black lines split part A and B. Circles mark the maximum value. Each position in the matrices correspond to a 8-beat subsequence.

ρ_i^K and all v shingles in a different subsequence ρ_j^K, defined as follows:

$$D_{min}(\rho_i^K, \rho_j^K) = \left(\sum_{u=1}^{K} min_v(\rho_i[u] - \rho_j[v])^2 \right)^{1/2} \tag{4}$$

There are two important subtleties that must be observed when calculating this measure. First, distances between shingles are defined by the element-wise L_2 norm based on the same pairwise distance matrix as before. Additionally, $\mathcal{I}(\Gamma)$ is a geometric mean and only produces large values when all pairwise distances are also large; any small distance in the product forces the overall measure toward zero.

3.3 Criterion Definition and Calculation

Having established measures of compression and disjoint information for some Γ, we capture both of these traits by defining a single criterion Θ as follows:

$$\Theta(\mathcal{C}, \mathcal{I}) = \frac{2\mathcal{C}\mathcal{I}}{\mathcal{C} + \mathcal{I}} \tag{5}$$

Noting that $\mathcal{C}$ and $\mathcal{I}$ are constrained on the interval [0,1] and converge to one when optimal, computing the criterion as a harmonic mean enforces the behavior that its value is only large when both measures are as well.

It is worthwhile at this point to make the observation that this criterion can —at least theoretically— be evaluated at every unique combination of subsequences Γ over an entire sequence $\mathbf{S}$. The output of this exhaustive calculation is a P dimensional tensor where each axis is of length J, and the best summary is given simply by the $argmax$ of the resulting data structure. From here onward, we use *optimal criterion* Θ_{max} to refer to the absolute maximum of this tensor, as would be found through a naïve, exhaustive search of the space. Note that for large J and P however, evaluating every cell in this tensor becomes computationally intractable and efficient approximations are necessary (see Section 4).

3.4 Case Example

Here we illustrate the behavior of the audio summary criterion by analyzing by the first half of Frédéric Chopin's Mazurka Op. 30 No. 2, which exhibits a well-defined *AB* structure. For the sake of demonstration, we select a subsequence length of $N = 8$ and define $P = 2$ such that an exhaustive evaluation of Θ produces a $J \times J$ matrix. The result of computing $\mathcal{C}$, $\mathcal{I}$ and Θ over all pairs of subsequences is shown in Figure 1.

The compression measure $\mathcal{C}$ is shown in the left-most matrix of Figure 1. This measure quantifies the extent to which a set Γ explains the overall track independent of any correlation between subsequences. The optimal $\mathcal{C}$ in this matrix corresponds to the two subsequences at beat indices (48, 59) in the *B*-*B* quadrant. These subsequences correspond to repetitions of the same part, making the information in Γ redundant.

The center matrix in Figure 1 corresponds to the disjoint information measure $\mathcal{I}$. This measure captures the degree of uniqueness between subsequences in Γ. It is clear from the plot that the measure behaves as expected: repeated subsequences in the same section (in quadrants *A*-*A* or *B*-*B*) produce significantly lower values of $\mathcal{I}$ than subsequence pairs in *A*-*B*, where the highest $\mathcal{I}$ is found.

Finally, the previous two matrices combine to yield a third, the criterion Θ. In the example the maximum value of $\mathcal{C}$ corresponds to repetitions of the same part, thus making $\mathcal{I}$ to be small and forcing the overall Θ to also be small. Similarly, the position of the maximum value of $\mathcal{I}$ at the boundary between *A* and *B* results in a low $\mathcal{C}$ value, again producing a smaller Θ. In this example, Θ is maximized by the combination of subsequences in *A*,*B* that best balance the two criteria by capturing the midsections of each part.

4. APPROXIMATING THE OPTIMAL SOLUTION

As mentioned in the previous section, naïve calculation of the optimal criterion can, in certain scenarios, become computationally inefficient, impractical, or worse. More specifically, an exhaustive evaluation and parallel search of the full Θ tensor of size $(J/2)^P$ would result in an algorithm of complexity $O((JN \log J)^P)$. In this section, we present a heuristic approach that approximates the optimal solution using a much faster implementation.

4.1 Heuristic Search Algorithm

The main idea behind the fast approach is to assume that the most relevant parts of a song will most likely be uniformly spread across time. The pseudocode is found in Algorithm 1. The method *EquallySpaced()* initializes all P subsequences into equally spaced time indices and stores them in the array Υ. We then iterate over the P subsequences, fixing all of them except the P_i being processed. We use a sliding window, operating over the region between the endpoint of the previous subsequence and the start of the next one, to find the best local music criterion θ by calling the function *ComputeCriterion()*. At every iteration we check if the sliding window is within the correct bounds with the method *CheckBounds()*, and if it is, we update the best index υ in Υ. Finally, the summary Γ is obtained by concatenating the subsequences at the time indices in Υ. This operation is done inside the method *GetSubseqsFromTimeIdxs()*.

Algorithm 1 Heuristic Approach

```
Require: $\mathbf{S} = \{s_1, \ldots, s_M\}, P, N$
Ensure: $\Gamma = \{\gamma_1^N, \ldots, \gamma_P^N\}$
  $\Upsilon \leftarrow$ EquallySpaced$(\mathbf{S}, P, N)$
  for $i = 1 \rightarrow P$ do
    $\theta \leftarrow 0$
    for $j = 1 \rightarrow M$ do
      if CheckBounds$(\Upsilon)$ then
        $\Theta \leftarrow$ ComputeCriterion$(\mathbf{S}, \Upsilon, N, P)$
        if $\Theta > \theta$ then
          $\theta \leftarrow \Theta; \upsilon \leftarrow j$
        end if
        $\Upsilon[i] \leftarrow j$
      end if
    end for
    $\Upsilon[i] \leftarrow \upsilon$
  end for
  $\Gamma \leftarrow$ GetSubseqsFromTimeIdxs$(\mathbf{S}, \Upsilon)$
  return $\Gamma$
```

The complexity in time of this algorithm is $O(PMJ)$, which makes it linear with respect to P. This approach improves the efficiency dramatically and let us explore different hyperparameter values of P and N, as we will see in subsection 5.4.

5. EVALUATION

We now proceed to evaluate multiple facets of the audio summary criterion. We begin by reviewing the dataset used for evaluation before presenting three different experiments.

5.1 Methodology

In our experimentation, we use a collection of solo piano music compiled by the Mazurka Project [2], comprised of 2914 tracks corresponding to different recorded performances of 49 Mazurkas. For clarity, we use *piece* or *work* when referring to a Mazurka, and reserve *track* or *performance* to describe an instance of the work as audio. The motivation for using this dataset is to leverage the several performances of a single work to measure the consistency of our criterion. Additionally, this collection contains 301 tracks with human-annotated, ground truth beat times, which allows us evaluate the impact of beat tracking on various dimensions of performance. It also provides the added benefit that Chopin's Mazurkas are notoriously difficult to beat-track via computational means [3].

5.2 Parameter Sweep & Selection

In the interest of selecting a feature space with which to proceed, an experiment is designed to sweep across the range of free parameters to identify the optimal configuration. There are three questions to address: Is automatic beat tracking sufficient? Do chroma and Tonnetz features perform equivalently, or is one preferable? Does performance vary significantly as a function of codebook size?

These three decisions can be resolved by observing how the optimal criterion behaves across various performances of the same work, comparing between ground truth and estimated beat annotations. Intuitively, a satisfactory audio summary of the same piece would persist across recorded versions, so the summaries themselves should be substantially similar.

For those 301 recordings with ground truth beat annotations, we stratify the tracks into five folds for cross validation such that all but one are used to train the quantizer and the remaining hold-out is reserved as a test set. Sweeping across the two beat annotation sources (ground truth, automatic), two harmonic representations (chroma, Tonnetz), and three codebook sizes (50, 100, 200) produces 12 possible feature space configurations (see Table 1). Summary sets Γ are identified by exhaustively computing Θ_{max} over all possible combinations of subsequences, where segment length N and number P are fixed at 16 and 4, respectively. Additionally, a stride parameter of $N/2$, analogous to a hop size in frame based audio processing, is applied to make the exhaustive search more computationally tractable.

To measure the degree to which summaries of the same work (intra-class distance) are closer than those from other, dissimilar works (inter-class distance), the pairwise distances between summaries of tracks in each fold are computed and the values are treated as empirical distributions of these two classes. The Fisher ratio, defined by (6), provides an estimate of the separation between intra- and inter-class summary distances.

$$F_{ratio} = \frac{\mu_{intra} - \mu_{inter}}{\sigma_{intra}^2 + \sigma_{inter}^2} \tag{6}$$

[2] *http://www.mazurka.org.uk*

k	GT-C	GT-T	A-C	A-T
50	3.64	3.97	2.71	3.89
100	3.84	4.29	2.68	4.20
200	4.09	**4.74**	2.87	**4.45**

Table 1. Parameter Sweep. GT: Ground Truth, A: Automatic, C: Chromagram, T: Tonnetz

Intuitively, higher values of F_{ratio} indicate distinct, well-localized distributions where 'similar' items cluster together, and translates to more consistency across performances. Table 1 shows the results of sweeping free parameters in the feature space. There are a few important observations to make about these results. First, a Tonnetz representation produces consistently better results than chroma features. Additionally, Tonnetz features computed from automatically extracted beat times only marginally trail their ground truth equivalent. Finally, the codebook size K has a non-trivial impact on performance and is positively correlated. Therefore, we can conclude that Tonnetz-features computed with a beat tracking front-end are the best choice going forward, and that the parameter K should be large and ultimately based on practical limitations of the implementation.

5.3 Heuristic Approximation

We evaluate the performance of the heuristic approach by comparing the summaries it produces with the optimal solution obtained through exhaustive computation. A second comparison is made with the expected performance a random algorithm, obtained by averaging across all results observed in the course of computing Θ_{max}. This establishes the upper (optimal) and lower (random) bounds of performance and allows us to determine where on this continuum our heuristic solution lives. We measure the discrepancy between the optimal Θ_{max}, random Θ_{rand}, and heuristic Θ_{heur} solutions by computing the averaged Mean-Squared Error (MSE) across all tracks in the full dataset. To account for local variance resulting for a given track, we normalize the range of $\mathbf{\Theta}$ such that $\Theta_{max} = 1$ and $\Theta_{min} = 0$. The normalized MSE can be expressed formally as follows:

$$\text{MSE}(\mathbf{\Theta}) = \frac{1}{\mathcal{S}} \sum_{i}^{\mathcal{S}} (1 - \mathbf{\Theta}_i)^2 \tag{7}$$

Here, a normalized Θ_{max} is always equals 1, $\mathbf{\Theta}$ represents a vector of normalized criteria obtained by some search strategy, and $\mathcal{S}$ is the number of songs in the Mazurka data set.

Setting the hyperparameters to $P = 4$ and $N = 16$, the MSE of the random baseline is approximately 21%, whereas our heuristic approximation is nearly two orders of magnitude better, achieving a MSE of slightly over 1%. It is evident from this contrast that the heuristic search very closely approximates the results of exhaustive computation, significantly outperforming the random baseline. Therefore we offer the preliminary conclusion that the heuristic approach is a sufficient approximation, allowing a more thorough exploration over the space of hyperparameters.

5.4 Automatically Selecting Hyperparameters

Having gained the efficiency to perform a search across hyperparameters P and N, we can compute Θ_{max} for different combinations and define the maximum over this set as the optimal summary. In this experiment we explore 9 pairs of $P \in [2 : 5]$ and $N \in [16 : 64]$ (constraining N to powers of two), avoiding (P, N) combinations such as $(5, 64)$ or $(2, 16)$ that would produce summaries that are too long or short, respectively. These ranges incorporate prior musical knowledge, as there are typically a small number of distinct parts in a work and meter is predominantly binary. It is worthwhile to mention though the best choice of P and N is signal-dependent and that, in reality, there is no universally optimal combination for all music.

In light of this, the combination of P and N that yields Θ_{max} for a given track provides another statistic that should persist across multiple performances of the same work, as structure and meter are generally invariant to interpretation. We evaluate the criterion further by measuring consistency of the optimal (P, N) pair using the entire Mazurka dataset, and provide qualitative examples of the observed behavior.

5.4.1 Quantitative Evaluation

A consistency distribution resulting from a sweep across combinations of P and N is given in Figure 2. The x-axis represents the proportion of performances for a given Mazurka that produces the most frequent (P, N) pair at Θ_{max}, where a value of 1 indicates complete agreement and 0 complete disagreement. The y-axis represents the number of works that produce a given consistency value, and there are 49 in total.

As illustrated by the plot, there is very high consistency ($\geq$ 90%) for more than half of the data set, resulting in an average consistency of 87%. This shows that our criterion is able to capture high-level information about the structure of a work across various performances, validating its capacity to produce informative audio summaries. Despite a high average overall, it is of special interest to qualitatively analyze the Mazurkas that yield different optimal configurations of the hyperparameters.

5.4.2 Qualitative Evaluation

Importantly, Figure 2 fails to capture is the degree of contrast between Θ_{max} and values for other combinations of P and N. Upon closer inspection, we find that the structure of some works is not clearly defined leading to multiple, equally reasonable interpretations. This manifests explicitly in the data, leading to more than one (P, N) with large Θ values. One such instance of multiple interpretations occurs for Op. 7 No. 2. The form of this work is *ABCA*, but– depending on performance – parts *B* and *C* can be interpreted as one longer part, resulting in an *ABA* structure. Consequently, 62% of these performances produced

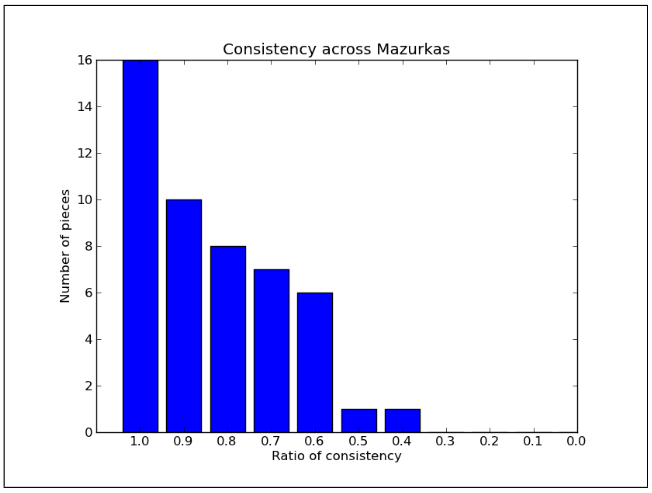


Figure 2. Evaluating consistency across different performances of the same song for the entire Mazurka data-set

a Θ_{max} for $P = 2$, while 31% of performances occurred at $P = 3$.

The other primary cause of inconsistency is due to tempo modulations and the resulting errors and artifacts caused by the beat tracker. An example of this is Op. 41 No. 1, producing the lowest consistency ratio of 49%. Here we observe a lack of well-defined onsets and liberal rhythmic interpretations, both within and between performances. This causes the beat tracker to behave erratically, producing misaligned feature sequences that ultimately yield Θ_{max} values for different pairs of (P, N).

Alternatively, Op. 24 No. 3, which exhibits a clear *ABA* structure and a more stable tempo, achieves 100% consistency for $P = 2$ and $N = 32$. The more noteworthy observation though is that this particular piece is in a ternary meter. Therefore a better summary would likely be obtained with N being a power of 3, and exploring other values of N could potentially improve consistency.

6. DISCUSSION & CONCLUSIONS

We have presented a novel audio summary criterion and established the merit of this approach through data-driven evaluation and qualitative inspection. We have illustrated how our criterion consistently produces informative summaries that capture both meaningful harmonic and high-level structural information. Finally, we have presented a heuristic approach capable of producing audio summaries that closely approximates the absolute maximum.

Complementary to the main body of work itself, the unexpected observation that Tonnetz features definitively yield better results warrants discussion. One possible explanation, as Tonnetz features live in a continuous-valued geometric space, is that any beat estimation errors result in a smooth interpolation of the feature space. Chroma features, acting as a time-varying probability distribution, cannot resolve timing errors in the same way. As a result, a beat tracker does not need to be perfect to be useful given a suitable feature representation.

As part of future work, we identify the potential of audio summaries to be used for various application where the data needs to be time normalized. More related to this work, the next logical step would be to explore the use of variable length subsequences to generate summaries. Finally, several example summaries are made available online [3] .

7. ACKNOWLEDGEMENTS

This work was supported by the Caja Madrid Foundation and the National Science Foundation under grant IIS-0844654, and the authors would like to sincerely thank them.

8. REFERENCES

[1] Mark Bartsch and Gregory Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, 2005.

[2] Matthew Cooper and Jonathan Foote. Summarizing popular music via structural similarity analysis. In *Proc. of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 127–130. IEEE, 2003.

[3] Peter Grosche and Meinard Muller. Extracting predominant local pulse information from music recordings. *Ieee Transactions On Audio Speech And Language Processing*, 19(6):1688–1701, 2011.

[4] Christopher Harte, Mark Sandler, and Martin Gasser. Detecting harmonic change in musical audio. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia AMCMM 06*, volume C, page 21. ACM Press, 2006.

[5] Matthias Mauch and Simon Dixon. Approximate note transcription for the improved identification of difficult chords. In *Proc. of the International Society of Music Information Retrieval*, volume 11, pages 135–140, 2010.

[6] Geoffroy Peeters and Xavier Rodet. Toward Automatic Music Audio Summary Generation from Signal Analysis. In *Proc. of the International Society of Music Information Retrieval*, 2002.

[7] Xi Shao, NC Maddage, and Changsheng Xu. Automatic Music Summarization Based On Music Structure Analysis. In *Proc. of the IEEE International Conference on Acoustics Speech and Signal Processing*, pages 1169–1172. IEEE, 2005.

[3] https://files.nyu.edu/onc202/public/ismir2012

SECOND FIDDLE IS IMPORTANT TOO: PITCH TRACKING INDIVIDUAL VOICES IN POLYPHONIC MUSIC

Mert Bay[2], Andreas F. Ehmann[2], James W. Beauchamp[2], Paris Smaragdis[1,2] and J. Stephen Downie[3]

[1]Department of Computer Science, University of Illinois at Urbana-Champaign
[2]Department of Electrical & Computer Eng., University of Illinois at Urbana-Champaign
[3]Graduate School of Library and Information Science, University of Illinois at Urbana-Champaign
{*mertbay,aehmann,jwbeauch,paris,jdownie*}*@illinois.edu*

ABSTRACT

Recently, there has been much interest in automatic pitch estimation and note tracking of polyphonic music. To date, however, most techniques produce a representation where pitch estimates are not associated with any particular instrument or voice. Therefore, the actual tracks for each instrument are not readily accessible. Access to individual tracks is needed for more complete music transcription and additionally will provide a window to the analysis of higher constructs such as counterpoint and instrument theme imitation during a composition. In this paper, we present a method for tracking the pitches (F0s) of individual instruments in polyphonic music. The system uses a pre-learned dictionary of spectral basis vectors for each note for a variety of musical instruments. The method then formulates the tracking of pitches of individual voices in a probabilistic manner by attempting to explain the input spectrum as the most likely combination of musical instruments and notes drawn from the dictionary. The method has been evaluated on a subset of the MIREX multiple-F0 estimation test dataset, showing promising results.

1. INTRODUCTION

One of the most important classes of information to be retrieved from music is its polyphonic pitch content. Recently, many researchers have attempted multiple-F0 estimation (MFE) [11, 14, 18]. (A review of state-of-the-art MFE systems can be found in [2].) Many current MFE systems such as [14] restrict themselves to the estimation of a certain number of F0s for each frame, while ignoring which instrument/timbre corresponds to each F0. While approaches for tracking solo melodic voice/timbres exist [15], the pitch tracking of additional lines and parts in polyphonic music opens new possibilities. By exposing the individual lines produced by each instrument in polyphonic music, aspects such as counterpoint, the appearance of *leitmotifs* across instruments in a piece, and hidden thematic references across musical pieces can be uncovered. Therefore, instruments and their timbres are very important components in music and should be tracked along with their F0s. Moreover, knowing each instrument's F0 track can be very beneficial for a variety of MIR user applications, such as music transcription, score alignment, audio music similarity, cover song identification, active music listening, melodic similarity, harmonic analysis, intelligent equalization, and F0-guided source separation.

As stated previously, most MFE systems produce low-level representations of the polyphonic pitch content present in music audio which report only what fundamental frequencies are present at each given time. However higher-level representations can attempt to track and link these fundamental frequencies over time to form notes such as described in [3]. It is important to note that such tracking can also improve the accuracy of MFE's based purely on individual frames, as reported reported in [18]. In [12] a classification approach is used to determine singing voice portions in music audio so as to build the pitch-track corresponding to a vocal melody. In [5, 6], a frame level multi-F0 estimation method is used followed by a constrained clustering method that uses harmonic amplitude-based features to cluster pitches into pitch tracks. In these cases, to build instrument or timbre-specific pitch tracks, bottom-up methods were used that first produced frame-based pitch estimates and subsequently sometimes attempted to build note-level representations, which may form solo phrases.

In the method presented in this paper, we use an alternative approach. Instead of building timbre tracks from the pitch content, our proposed approach uses timbre information to guide the formation of instrument-specific pitch tracks. This paper is organized as follows. Section 2 details the proposed method. Section 3 describes the evaluation datasets, measures, and results. The evaluation results are discussed in Section 4 and conclusions and future work covered in Section 5.

2. PROPOSED METHOD

To make a system that tracks pitches attributed to different musical instruments, we borrow an idea from the supervised sound source separation domain: using a spectral library [1, 13]. By training our system on example sounds from different instruments, we can track them in complex sound mixtures. The proposed method is based on probabilistic latent component analysis (PLCA) [17]. PLCA is a variant of non-negative matrix factorization (NMF) and has been used widely to model sounds in the spectral domain. Its probabilistic interpretation makes it extensible to using priors and statistical techniques. We use regular PLCA to build dictionaries of spectra indexed by F0 and instrument where the spectra are analyzed from recordings of individual notes of different musical instruments. We extend the model of [16] to represent each source/instrument not by only one spectral dictionary, but rather with a collection of dictionaries, each of which is an ensemble of spectral basis vectors that have the same F0. We model the input music signal's spectrum as a sum of basis vectors from F0-specfic spectrum dictionaries for different instruments. Update rules are designated to calculate the model parameters, which are estimated probabilities for the occurrence of each spectral basis vector, F0-spectrum dictionary, and instrument in the input mixture at a given time. Finally, we perform the Viterbi algorithm [8] to track the most likely pitch sequence for each instrument.

A sinusoidal model is used for the time-frequency representation because of its compactness and also for representing each note independent of the intonation errors or tuning differences between training and test set performers. We begin by performing a short-time Fourier transform on the audio signal. Peaks in the spectrum are then determined using a frequency-dependent threshold as described in [7]. We then refine each peak's amplitude and frequency using a signal derivative method proposed by [4].

2.1 Model

We model the audio input mixture's spectrum for each frame as a sum of instrument tones where each tone is represented by a dictionary of spectral basis vectors that are learned in advance.

It is assumed that all instrument tones have harmonically related frequencies which are integer multiples of an F0 frequency. It turns out that in a mixture of such tones there is a high probability that harmonics will overlap. The input spectrum can be modeled as a distribution/histogram over a range of frequencies. E.g., each component is viewed as the probability of occurrence of that particular frequency. The magnitude at any particular frequency can be thought as an accumulation of magnitudes from various instruments due to component overlapping. The scaled version of the input mixture spectrum is modeled as a discrete distribution. The generative process is modeled as follows:

$$X_t(f) \cong P_t(f) = \sum_i^I P_t(i) \sum_{n\in p_i}^N P_t(p|i) \sum_{z\in z_{p_i}}^K P_t(z|p) P_{p_i}(f|z) \quad (1)$$

where $X_t(f_j)$ is the spectral magnitude of the peak j at frequency f_j for the observed input mixture spectrum at time t; $P_t(f)$ is an approximation of the input spectrum; $P_t(i)$ is the estimated probability of occurrence of instrument i at time t, whereas $P_t(p|i)$ is the estimated probability that pitch p is produced by instrument i; $P(f|z)$ is the learned spectral basis vector for the pitch p of instrument i; and $P_t(z|p)$ is the probability (weight) of that basis vector. The above model explains the mixture magnitude spectrum hierarchically as the sum of N individual pitches from I different instruments where the dictionary corresponding to each pitch/instrument has K elements. The independence relationships of the model can be represented by the graph I→P→Z→F. The process that generates the frequency components in the observed magnitude spectrum is as follows: First, an individual instrument library is selected from a group of instrument libraries. Second, a spectrum dictionary is drawn for each pitch from the instrument library. Third, a spectral basis vector is drawn from a particular F0-spectrum dictionary for the instrument. Fourth, a spectral component at a particular frequency is drawn from the spectral basis vector. Although it is possible that this spectral component will be generated by only a single instrument, most often it only contributes a fraction to the magnitude of the observed spectrum at that frequency.

2.2 Parameter Estimation

Parameters for the model $\theta = \{P_t(z|p), P_t(p|i), P_t(i)\}$ can be estimated using an expectation-maximization (EM) algorithm. In the E-step, current parameter values θ^{old} are used to calculate the posterior distribution.

$$P_t(i,p,z|f,\theta^{old}) = \frac{P_t(f|i,p,z)P_t(i,p,z)}{P_t(f)} \quad (2)$$

Because of the structure of the model, $P_t(f|i,p,z) = P_t(f|z)$ and $P_t(i,p,z) = P_t(i)P_t(p|i)P_t(z|p)$. We can write the posterior as

$$P_t(i,p,z|f,\theta^{old}) = \frac{P(f|z)P_t(z|p)P_t(p|i)P_t(i)}{\sum\limits_i P_t(i) \sum\limits_{n\in p_i} P_t(p|i) \sum\limits_{z\in z_{p_i}} P_t(z|p)P_{p_i}(f|z)} \quad (3)$$

The posterior is used to calculate the expectation of the complete data log likelihood Q

$$Q(\theta,\theta^{old}) = \sum_f X_f \sum_i \sum_{n\in p_i} \sum_{z\in z_{p_i}} P(i,p,z|f,\theta^{old}) log(P(i,p,z,f|\theta) \quad (4)$$

In the M-step, new parameters are estimated by maximizing the above function according to θ, resulting in the following update rules:

$$P_t(z|p)^{new} \leftarrow \frac{\sum\limits_f P_t(i,p,z|f)X_t(f)}{\sum\limits_{z\in z_{p_i}} \sum\limits_f P_t(i,p,z|f)X_t(f)} \quad (5)$$

$$P_t(p|i)^{new} \leftarrow \frac{\sum\limits_{z\in z_{p_i}} \sum\limits_f P_t(i,p,z|f)X_t(f)}{\sum\limits_{p\in p_i} \sum\limits_{z\in z_{p_i}} \sum\limits_f P_t(i,p,z|f)X_t(f)} \quad (6)$$

$$P_t(i)^{new} \leftarrow \frac{\sum\limits_{p\in p_i} \sum\limits_{z\in z_{p_i}} \sum\limits_f P_t(i,p,z|f)X_t(f)}{\sum\limits_i \sum\limits_{p\in p_i} \sum\limits_{z\in z_{p_i}} \sum\limits_f P_t(i,p,z|f)X_t(f)} \quad (7)$$

We then use the new estimates to calculate the posterior in an iterative manner and repeat until convergence is achieved.

2.3 Sparsity Prior

At any given time, we expect each instrument active to be playing a single pitch. Even though the parameter estimation method would allow multiple pitches per frame for each instrument, in this project our goal is to track a monophonic pitch contour for each instrument. We also expect that not all instruments are active at the same time.

Enforcing sparsity constraints on note and instrument probabilities $P_t(p|i)$'s and $Pt(i)$'s reinforce this behavior. We use the following prior (the normalizing constant is dropped for convenience)

$$P(\phi) = \left(\sum_{\phi} (P(\phi))^{\alpha}\right)^{\beta} \tag{8}$$

Adding the above prior to the expectation of the complete data log-likelihood and maximizing it with respect to $P(i)$ and $P(p|i)$, we arrive at the following update rules

$$P(p|i)^{new} \leftarrow \sum_{z \in z_{p_i}} \sum_{f} P(i,p,z|f) X_f + \frac{\beta p(P_t(p|i))^{\alpha}}{\sum_{p \in p_i} (P_t(p|i))^{\alpha}} \tag{9}$$

$$P_t(i)^{new} \leftarrow \sum_{p \in p_i} \sum_{z \in z_{p_i}} \sum_{f} P_t(i,p,z|f) X_t(f) + \frac{\beta p(P_t(i))^{\alpha}}{\sum_{p \in i} (P_t(i))^{\alpha}} \tag{10}$$

(We have to rescale explicitly so that it sums up to one.)

$$P(p|i)^{new} \leftarrow \frac{P(p|i)^{new}}{\sum_{p \in p_i} P(i)^{new}} \text{ and } P(i)^{new} \leftarrow \frac{P(i)^{new}}{\sum_{i} P(i)^{new}} \tag{11}$$

2.4 Enforcing Continuity

Our goal is to track each instrument's F0 through time. We expect the pitch contour to be smooth, not changing drastically from frame to frame except at note transitions. Using the Viterbi algorithm, we treat the pitches as hidden states and pose the instrument tracking problem as one of inferring the most likely pitch state sequence for each instrument. Above estimated pitch distributions (which maximize the mixture likelihood $P(X_t(f)|i,p,z)$) for each instrument are considered to be the emission probability of the hidden state in a hidden Markov model (HMM). Transition probabilities are modeled as normal distributions given by

$$P(p_t|p'_{t-1}) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\left(f_{0\,t} - f'_{0\,t-1}\right)^2}{2\sigma^2}} \tag{12}$$

where p_t denotes the hidden pitch state for instrument i at time t. $f_{0\,t}$ denotes the F0 associated with the hidden pitch state for instrument i at time t. Transitions are calculated within the same instrument. We empirically choose $\sigma = 7 + f/100Hz$. The above distribution enforces the continuity of the active notes from frame to frame.

3. EVALUATION ON REAL WORLD DATA

We trained a dictionary for each pitch of each instrument using the RWC musical instrument database [9] using non-negative matrix factorization with Kullback-Leibler divergence which is numerically equal to the regular PLCA method in 2 dimensions [17]. For each pitch from the RWC dataset, 20 representative spectra were derived from 27 different tones corresponding to 3 players, 3 dynamics (piano, mezzo, forte), and 3 articulations (normal, staccato, vibrato). In the pitch tracking stage, we limit the number of instrument libraries to choose from by designating the instruments expected to be in the input mixture as input to the algorithm. We also limit the search range for F0 (which F0-spectrum dictionaries to use) of each instrument by only using the peaks estimated by the sinusoidal model that are between 50 Hz to 2500 Hz as pitch candidates.

Evaluations are performed at frame level. Multi-F0 tracking problem is similar to melody extraction problem extended to multiple melodies as opposed multi-F0 estimation problem where the estimated number of F0s for each frame can be different than the ground-truth F0s, which is not the case in the tracking problem, where the number of estimated F0s and the reference F0s are simply equal to total number of frames. We extended the evaluation metrics from MIREX melody extraction task to our problem. Comparing the voiced (nonzero F0) and unvoiced (zero F0) values for each frame of the estimated and ground-truth F0 tracks, the counts for true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) are calculated according to Table 1

		Estimated	
		voiced	unvoiced
Ground	voiced	TP	FN
truth	unvoiced	FP	TN

Table 1. Evaluation

TP's further break down into ones with correct F0 and the ones with incorrect F0 as $TP = TPC + TPI$. Estimated voiced F0 is correct if it is within a quarter tone (+-2.93%) range of a positive ground-truth F0 for that frame.

The precision, recall, F-measure, and accuracy for each input test file is then calculated over all frames and all instruments as:

$$\text{Precision} = \frac{\sum_i \sum_t TPC_{i,t}}{\sum_i \sum_t TP_{i,t} + FP_{i,t}} \tag{13}$$

$$\text{Recall} = \frac{\sum_i \sum_t TPC_{i,t}}{\sum_i \sum_t TP_{i,t} + FN_{i,t}} \tag{14}$$

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{15}$$

$$\text{Acc.} = \frac{\sum_i \sum_t TPC_{i,t} + TN_{i,t}}{\sum_i \sum_t TP_{i,t} + FP_{i,t} + TN_{i,t} + FN_{i,t}} \tag{16}$$

where t is the frame index and i is the instrument index. We tested the proposed method on different datasets. The ground-truths for these datasets were estimated using monophonic pitch estimators (*Wavesurfer, Praat and YIN*) on the single-instrument recordings prior to mixing. The results of the monophonic pitch estimators are manually corrected where necessary.

For preliminary testing and development, we applied the method on a 11 second excerpt taken from a real world performance by bassoon, clarinet and oboe which was taken from a MIREX multitrack dataset (standard woodwind quintet transcribed from L. van Beethoven "Variations for String Quartet", Op.18, N.5). The three separate tracks were mixed to monaural. The results can be seen in Table 2. The proposed method scored 0.83 accuracy on average. Figure 1 shows the multiple-F0 tracks for each instrument. Without tracking, the F0's would not be connected from frame to frame.

	Bassoon	Clarinet	Oboe	Ave.
Accuracy	0.76	0.85	0.89	0.83
Precision	0.76	0.85	0.89	0.83
Recall	0.82	0.92	0.90	0.88
F-measure	0.79	0.88	0.89	0.86

Table 2. Evaluation performances for a 11-s woodwind trio excerpt (bassoon, clarinet, oboe).

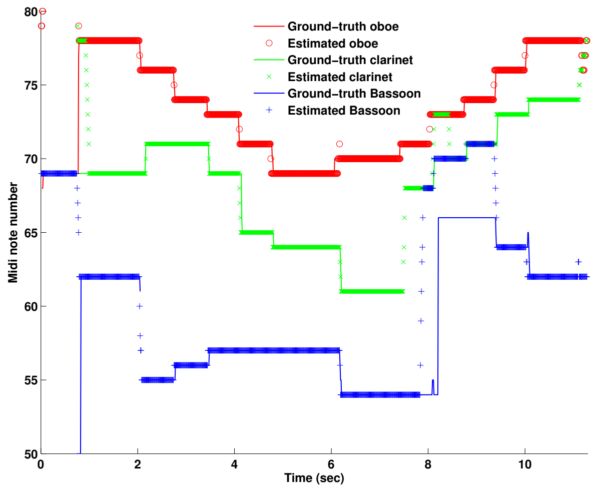


Figure 1. Pitch (in midi note numbers) vs. time using the proposed system on the 11-s woodwind trio excerpt (bassoon (lower), clarinet (middle), oboe (upper)). Thin lines represent the ground-truth.

We then tested the proposed method on two datasets used in the MIREX multiple-F0 task test set [2]. The first one is a multitrack recording of the woodwind quintet mentioned above [2]. The piece is highly contrapuntal as opposed to consisting of a lone melodic voice plus accompaniment. The predominant melodies alternate between instruments. The F0 tracks often cross each other. Five non-overlapping 30-second sections were chosen from the recording. Isolated instruments were mixed from solo tracks to polyphonies ranging from 2 (duo) to 5 (quintet), resulting in a total of 20 tracks (4 different polyphonies times five sections). The average pitch-tracking results over all tracks for polyphony 2 to 5 are shown in Table 3.

Accuracy	Precision	Recall	F-measure
0.52	0.50	0.56	0.53

Table 3. Average performance on the MIREX woodwind quintet dataset.

Figure 2 shows a bar graph of the performance of the method for different polyphonies. Figure 3 shows the average accuracies for different instruments in various polyphonies.

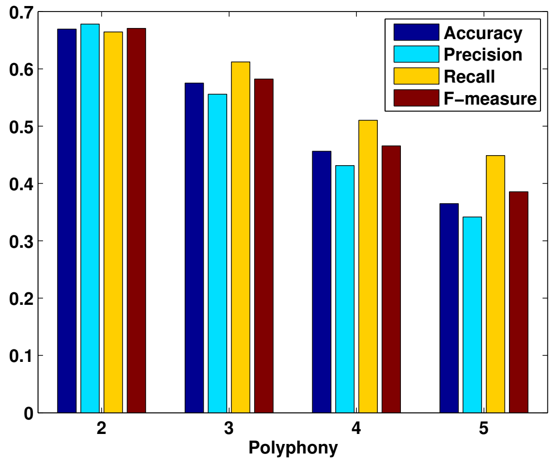


Figure 2. Performance vs. polyphony for the MIREX woodwind quintet dataset (Five 30-s segments per polyphony).

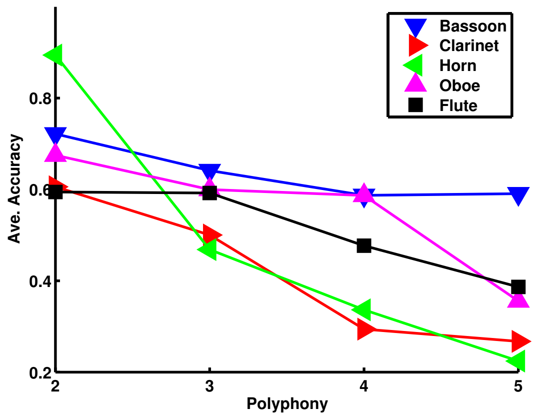


Figure 3. Ave. accuracy of individual instruments for different polyphonies for the MIREX woodwind quintet dataset. (Five 30-s segments)

The second dataset we tested our method on is a recording of a four-part J.S. Bach chorales created by [5] consisting of bassoon, clarinet, saxophone, and viola. Four 30 seconds sections were mixed from duo to quartet, resulting in 12 tracks (2 different polyphonies times 4 sections). The average results over all tracks in this dataset can be seen in Table 4.

Accuracy	Precision	Recall	F-measure
0.59	0.55	0.55	0.55

Table 4. Average performance for the MIREX Bach chorales dataset

Figure 4 shows a bar graph of the performance of the method for different polyphonies. Figure 5 shows the average accuracies of different instruments in various polyphonies. The RWC dataset of the MIREX multiple-F0 task was not evaluated because RWC samples were used for training. Also the piano dataset was not used because our project's goal is to track distinct timbres.

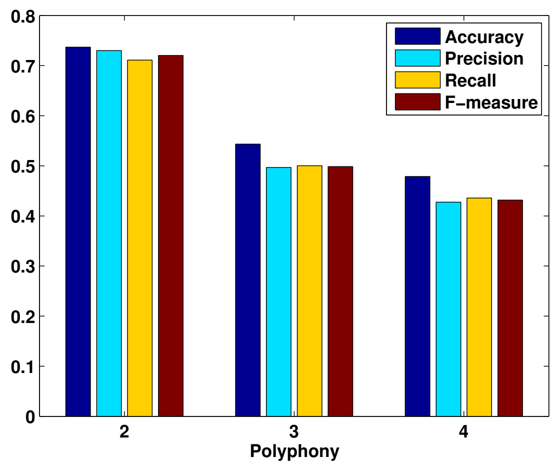


Figure 4. Ave. Performance vs. polyphony for the MIREX Bach chorales dataset. (Four 30-s segments)

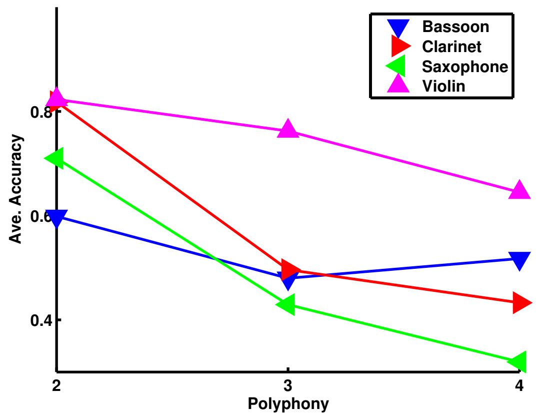


Figure 5. Ave. accuracy of instruments in different polyphonies for the MIREX Bach chorales quintet dataset.(Four 30-s segments)

4. DISCUSSION

The proposed method on average performed with 83% accuracy on identifying the instrument tracks for the trio case (Figure 1) which was used for the development of the algorithm. Accuracies were 52% and 59% for the MIREX woodwind quintet and Bach chorales quartet datasets. By examining the MIREX dataset results, we see that most problems are caused by an inactive instrument following the dominant instrument's F0 track. Some instruments in this dataset are inactive during 70-80% of the entire duration of the input. Accuracy-vs.-instrument results for the MIREX woodwind quintet (Figure 3), indicate that instruments horn and clarinet have lower performance in the 4 and 5 polyphony cases, due to their F0 tracks being highly sparse. In addition to remaining silent much of the time, they often play soft notes when they are active.

The tracking system reports note probabilities for every non-silent frame, which are then used in an HMM to estimate the F0 tracks. Voicing decisions are based strictly on the rms amplitude of the mixture input signal to decide whether the signal is silent. This results in a high number of false positives when an individual instrument is silent in parts of a track, a condition which happens often in the MIREX dataset. This behavior also results in a very low number of true and false negatives (since the system reports very few negatives) resulting in a higher recall than lower precision which can be seen in Figures 2 and 4. In the future, we would like to explore methods to infer whether instruments are active or not at any given point in time.

Another kind of error that frequently occurs is when a less dominant instrument tracks a more dominant one that has a similar timbre. This is probably caused by the instrument spectra having significant correlation with each other. Looking at the performance-vs.-polyphony results for each instrument in Figure 5, we see that instruments like violin, which have a unique timbre, have better average accuracy.

Possible solutions include training the instrument spectrum dictionaries not in isolation but in combination with other instrument spectra, which may assist the basis vectors behaving in a more discriminant way. Methods for discriminant non-negative tensor factorizations are explored in [19]. This issue can also be addressed in the testing part. Pitch probabilities in EM iterations can be estimated to be as maximally different as possible, while still explaining the overall mixture by appropriate use of priors. Another method that might improve this issue would be to use a factorial HMM to jointly estimate the pitch tracks.

On average, the proposed method scored 0.53 for Accuracy on the MIREX dataset, which is an improvement over the only past result (0.21) for the multiple-F0 tracking task evaluated at MIREX [10]. However, we note that the MIREX multiple-F0 task did not offer the opportunity to utilize instrument names for the mixture input test files.

One reason the proposed method works comparatively well for the trio and the Bach chorales case (see Table 1 and Figure 1) is that most instruments were active most of the time. The encouraging results from this case lead us to believe that the pitch tracking problems can be improved effectively by addressing the issues discussed above.

Finally, we note that the restriction that sounds must consist soley of harmonic partials can be relaxed. E.g., pitch estimates for instruments like xylophone, which do not have strict harmonic structures but do have predictable inharmonic structures, can be determined.

5. CONCLUSION

A new method for pitch and instrument tracking of individual instruments in polyphonic music has been designed and evaluated on an established dataset that has previously been used for multiple-F0 estimation under MIREX. Current results are encouraging, but several problems need to be resolved (as described in the Discussion section) for the method to be an effective tool.

As mentioned in the Introduction, knowing the F0 tracks can be very beneficial for a variety of MIR tasks. In the future, we would like to explore the use of voicing detection to determine which instruments are active. We would also like to perform instrument identification in the front-end so the method does not require prior knowledge of the instrumentation. We also plan to experiment with different discriminant learning methods and to re-infer the dictionaries based on the input mixture. Finally, we propose to explore using an automatic key detection algorithm and a more musicologically informed pitch transition matrix for the hidden Markov model.

6. REFERENCES

[1] M. Bay and J. Beauchamp. Harmonic source separation using prestored spectra. *Proc. Independent Component Analysis and Blind Signal Separation*, pages 561–568, 2006.

[2] M. Bay, A.F. Ehmann, and J.S. Downie. Evaluation of multiple-F0 estimation and tracking systems. In *Proc. 10th Int. Conf. for Music Information Retrieval (ISMIR 2009)*, pages 315–320, 2009.

[3] W.C. Chang, A.W.Y. Su, C. Yeh, A. Roebel, and X. Rodet. Multiple-f0 tracking based on a high-order hmm model. In *Proc. 11th Int. Conf. Digital Audio Effects (DAFx-08)*, pages 379–386, 2008.

[4] M. Desainte-Catherine and S. Marchand. High-precision Fourier analysis of sounds using signal derivatives. *J. Audio Eng. Soc*, 48(7/8):654–667, 2000.

[5] Z. Duan, J. Han, and B. Pardo. Harmonically informed multi-pitch tracking. In *Proc. 10th Int. Conf. on Music Information Retrieval (ISMIR 2009)*, pages 333–338, 2009.

[6] Z. Duan, J. Han, and B. Pardo. Song-level multi-pitch tracking by heavily constrained clustering. In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2010)*, pages 57–60, 2010.

[7] M.R. Every and J.E. Szymanski. Separation of synchronous pitched notes by spectral filtering of harmonics. *IEEE Trans. Audio, Speech and Language Processing*, 14(5):1845–1856, 2006.

[8] G.D. Forney Jr. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[9] M. Goto. Development of the RWC music database. In *Proc. 18th Int. Congress on Acoustics (ICA 2004)*, volume 1, pages 553–556, 2004.

[10] IMIRSEL. Multiple fundamental frequency estimation and tracking results wiki. `http://www.music-ir.org/mirex/wiki/2010:Multiple_Fundamental_Frequency_Estimation_%26_Tracking_Results`, 2010.

[11] A. Klapuri. Multipitch analysis of polyphonic music and speech signals using an auditory model. *IEEE Trans. on Audio, Speech, and Language Processing*, 16(2):255–266, 2008.

[12] R. Marxer, J. Janer, and J. Bonada. Low-latency instrument separation in polyphonic audio using timbre models. *Proc. 10th Int. Conf. on Latent Variable Analysis and Signal Separation*, pages 314–321, 2012.

[13] G. Mysore, P. Smaragdis, and B. Raj. Non-negative hidden markov modeling of audio with application to source separation. *Proc. 8th Int. Conf. on Latent Variable Analysis and Signal Separation*, pages 140–148, 2010.

[14] A. Pertusa and J.M. Inesta. Multiple fundamental frequency estimation using gaussian smoothness. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing,(ICASSP 2008)*, pages 105–108, 2008.

[15] 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 Audio, Speech, and Language Processing*, 15(4):1247–1256, 2007.

[16] B. Raj and P. Smaragdis. Latent variable decomposition of spectrograms for single channel speaker separation. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, (WASPAA 2005)*, pages 17–20, 2005.

[17] P. Smaragdis, B. Raj, and M. Shashanka. A probabilistic latent variable model for acoustic modeling. *Proc. 20th Conf. on Neural Information Processing Systems (NIPS 2006)*, 148, 2006.

[18] C. Yeh, A. Roebel, and X. Rodet. Multiple fundamental frequency estimation and polyphony inference of polyphonic music signals. *IEEE Trans. on Audio, Speech, and Language Processing*, 18(6):1116–1126, 2010.

[19] S. Zafeiriou. Discriminant nonnegative tensor factorization algorithms. *IEEE Trans. on Neural Networks*, 20(2):217–235, 2009.

FEATURE LEARNING IN DYNAMIC ENVIRONMENTS: MODELING THE ACOUSTIC STRUCTURE OF MUSICAL EMOTION

Erik M. Schmidt, Jeffrey Scott, and Youngmoo E. Kim
Music and Entertainment Technology Laboratory (MET-lab)
Electrical and Computer Engineering, Drexel University
{eschmidt,jjscott,ykim}@drexel.edu

ABSTRACT

While emotion-based music organization is a natural process for humans, quantifying it empirically proves to be a very difficult task, and as such no dominant feature representation for music emotion recognition has yet emerged. Much of the difficulty in developing emotion-based features is the ambiguity of the ground-truth. Even using the smallest time window, opinions about emotion are bound to vary and reflect some disagreement between listeners. In previous work, we have modeled human response labels to music in the arousal-valence (A-V) emotion space with time-varying stochastic distributions. Current methods for automatic detection of emotion in music seek performance increases by combining several feature domains (e.g. loudness, timbre, harmony, rhythm). Such work has focused largely in dimensionality reduction for minor classification performance gains, but has provided little insight into the relationship between audio and emotional associations. In this work, we seek to employ regression-based deep belief networks to learn features directly from magnitude spectra. Taking into account the dynamic nature of music, we investigate combining multiple timescales of aggregated magnitude spectra as a basis for feature learning.

1. INTRODUCTION

The problem of automated recognition of emotional content (mood) within music has been the subject of increasing attention among the music information retrieval (Music-IR) research community [1]. While there has been much progress in machine learning systems for estimating human emotional response to music, little progress has been made in terms of intuitive feature representations. Current methods generally focus on combining several feature domains (e.g. loudness, timbre, harmony, rhythm) and performing dimensionality reduction techniques to extract the most relevant informaiton. In many cases these methods have failed to provide enhanced classification performance, and they leave much to be desired in terms of understanding the complex relationship between emotional associations and acoustic content.

The Music Information Retrieval Evaluation eXchange (MIREX) [1] audio mood classification task provides an excellent illustration of this. Shown in Figure 1 is the performance of MIREX submissions for each year. The first year MIREX ran the task it received 9 submissions, and the best performing system achieved 61.50% performance on the 6-class problem using a feature space spanning 16-dimensions [2]. Each year the task has received a larger number of submissions, with exponentially larger feature libraries, but have failed to produce significant performance gains. Most recently, in 2010 the task received 36 submissions with the best system mining a 70-dimensional feature space, but achieved only 64.17% [3]. These results perhaps indicate that the data necessary for informing systems for this problem is not present in any current feature set.

Human judgments are necessary for deriving emotion labels and associations, but individual perceptions of the emotional content of a given song or musical excerpt are bound to vary and reflect some degree of disagreement between listeners. This lack of specificity presents significant challenges for developing informative feature representations for content-based music emotion prediction. In previous work we have investigated modeling emotional responses to music as both a singular point [4] as well as a stochastic distribution [5] over the arousal-valence (A-V) space of emotional affect. In this two dimensional representation valence indicates positive versus negative emotion and arousal reflects emotional intensity [6].

The ambiguous nature of musical emotion makes it an especially interesting problem for the application of feature learning. Using deep belief networks (DBNs) [7] we develop methods for learning emotion-based acoustic representations directly from magnitude spectra. In previous work, we have found these models to be powerful methods for generating reduced dimensionality representations of raw input spectra [8]. In that approach, we learned features directly from spectra at the 20msec rate of our windowed Short-Time Fourier Transform (STFT). In doing so, we provided a direct comparison between the DBN model for extracting features to standard acoustic representations such as MFCCs.

[1] http://www.music-ir.org/mirex

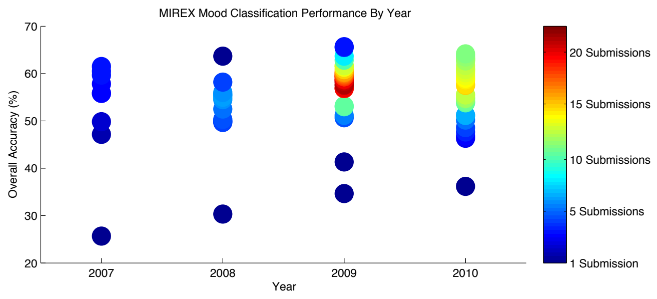


Figure 1. MIREX mood classification task performance by year.

But as humans require a larger time window than 20msec to determine emotions, and building upon that work we seek to improve our performance by informing our learned feature representations with spectra aggregated at multiple timescales. In addition, we investigate a universal background model (UBM) approach to feature learning. As DBN training follows an unsupervised pretraining, we investigate bootstrapping a much larger unlabeled dataset in developing our models. Given the challenges of collecting emotion annotated data, pretraining on a limited dataset is insufficient to form a general music model for finetuning. By bootstrapping a larger dataset we demonstrate significant improvement in using our DBN models for emotion prediction, and modest gains when using our learned features in a separate supervised machine learning approach.

We compare the learned feature representations to other state-of-the-art representations investigated in prior work [4, 5, 9]. In these experiments, we use the DBN hidden layer outputs as features to predict the training labels using a separate linear regression model. In all experiments we show that the features generated by the DBN outperform all other features, and that the topology is especially promising in providing insight into the relationship between acoustic data and emotional associations.

2. BACKGROUND

Feature learning has only recently gained attention in the machine listening community. Lee *et al.* was the first to apply deep belief networks to acoustic signals, employing an unsupervised convolutional approach [10]. Their system employed PCA to provide a dimensionality reduced representation of the magnitude spectrum as input to the DBN and showed slight improvement over MFCCs for speaker, gender, and phoneme detection.

Hamel and Eck applied deep belief networks (DBNs) to the problems of musical genre identification and autotagging [11]. Their approach used raw magnitude spectra as the input to their DBNs, which were constructed from three layers, employing fifty units at each layer. The system was trained using a greedy-wise pre-training and fine-tuned on a genre classification dataset, consisting of 1000 30-second clips. The system took 104 hours to train, and as a result was not cross-validated. Applied to a genre classification task, the learned features achieved a classification accuracy of 0.843, which was an increase over MFCCs at 0.790. The learned model was also used to inform an autotagging algorithm, which scored 0.73 in terms of mean accuracy, a slight improvement over MFCCs at 0.70.

3. GROUND TRUTH DATA COLLECTION

In prior work, we developed an online collaborative annotation activity based on the two-dimensional A-V model [12]. In this activity, participants use a graphical interface to indicate a dynamic position within the A-V space to annotate 30-second music clips. Each subject provides a check against the other, reducing the probability of nonsense labels. The song clips used are drawn from the "uspop2002" database.[2] 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 [13]. The purpose of the MTurk activity was to provide a dataset collected through more traditional means to assess the effectiveness of the game, specifically to determine any biases created though 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 is available to the research community,[3] and is densely annotated, containing $4,064$ label sequences in total, 16.93 ± 2.690 ratings per song. In this work we demonstrate the application of this densely annotated corpus for emotion-based feature learning.

4. ACOUSTIC FEATURE COLLECTION

Since our focus is on learning features that are specifically tuned to emotion prediction, we limit our comparisons to features that performed well in previous work. The features are also commonly used in the machine listening community and provide a reasonable baseline for testing. Our collection (Table 1) consists of the two highest performing features in prior work, Spectral Contrast and MFCCs [4,5], as well as the Echo Nest Timbre (ENT) features.

Feature	Description
Spectral Contrast [14]	Rough representation of the harmonic content in the frequency domain.
Mel-frequency cepstral coefficients (MFCCs) [15]	Low-dimensional representation of the spectrum warped according to the mel-scale. 20 dimensions used.
Echo Nest Timbre features (ENTs)[4]	Proprietary 12-dimensional beat-synchronous timbre feature

Table 1. Acoustic feature collection for music emotion prediction.

[2] http://labrosa.ee.columbia.edu/projects/musicsim/uspop2002.html
[3] http://music.ece.drexel.edu/research/emotion/moodswingsturk
[4] http://developer.echonest.com

5. DEEP BELIEF NETWORKS

A fully trained deep belief network shares an identical topology to a neural network, though they offer a far-superior training procedure, which begins with an unsupervised pre-training that models the hidden layers as restricted Boltzman machines (RBMs) [7,16,17] . A graphical depiction of an RBM is shown in Figure 2. An RBM is a generative model that contains only a single hidden layer, and in simplistic terms they can be thought of two sets of basis vectors, one which reduces the dimensionality of the data and the other that reconstructs it.

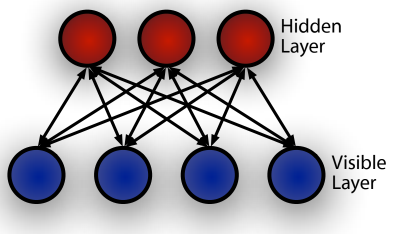


Figure 2. Restricted Boltzman machine topology.

RBMs are Markov random fields (MRFs) with hidden units, in a two layer architecture where we have visible units $\mathbf{v}$ and hidden units $\mathbf{h}$. This has an energy function of the form,

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i \in \text{freq}} b_i v_i - \sum_{j \in \text{features}} c_j h_j - \sum_{i,j} v_i h_j w_{ij} \quad (1)$$

where in this case the input is a spectrogram $\mathbf{v} \in \mathbb{R}^{1\times I}$ and the hidden layer $\mathbf{h} \in \mathbb{R}^{1\times J}$. The model has parameters $\mathbf{W} \in \mathbb{R}^{I\times J}$, with biases $\mathbf{c} \in \mathbb{R}^{1\times J}$ and $\mathbf{v} \in \mathbb{R}^{1\times I}$. During pre-training, we learn restricted Boltzman machines "greedily," where we learn them one at a time from the bottom up. That is, after we learn the first RBM we retain only the forward weights, and use them to create the input for training the next RBM layer.

As in the typical approach to deep learning, after pre-training we form a multi-layer perceptron using only the forward weights of the RBM layers. However, in typical approaches the final step is to attach logistic regression layer to the output of the MLP, and the full system is fine-tuned for classification using gradient descent. Since our goal is to learn feature detectors for a regression problem, we instead attach a simple linear regression layer and report the prediction error for fine-tuning as the mean squared error of the estimators. Squared error is chosen as opposed to Euclidean error for speed and numerical stability, as both functions have the same minimum. Furthermore, we elect to do our fine-tuning using conjugate gradient optimization, which we found to outperform gradient descent for our topology during initial testing.

We trained our DBNs using Theano, [5] a Python-based package for symbolic math compilation, and Scipy's optimization toolbox for the conjugate gradient optimization. Theano is an extremely powerful tool for machine learning problems because it combines the simplicity of Python with the power of compiled C, which can target the CPU or GPU.

[5] http://deeplearning.net/software/theano/

6. EXPERIMENTS AND RESULTS

In the following experiments we investigate employing deep belief networks for emotion-based acoustic feature learning. In all experiments, the model training is cross-validated 5 times, dividing the dataset into 50% training, 20% verification, and 30% testing. To avoid the well-known album-effect, we ensured that any songs that were recorded on the same album were either placed entirely in the training or testing set.

All learned features are then evaluated in the context of multiple linear regression (MLR), as we have investigated in prior work [4,5,18]. MLR provides extremely high computational efficiency, making it ideal for discriminating between relative usefulness of many feature domains.

6.1 Short-time Feature Learning

In the first set of experiments, we investigate learning features directly from short-time magnitude spectra. We have investigated this approach in prior work in the context of a different dataset [4], and we investigate it here to compare performance with the Turk dataset and to provide a baseline for our further work. As with our previous work, we use 3 hidden layers in all experiments, each containing 50 nodes. Furthermore, we run pre-training for 50 epochs with a learning rate of 0.001. During the conjugate gradient fine-tuning stage we attach an additional multiple linear regression (MLR) layer to the output of the DBN. As this stage is supervised, for each input example $\mathbf{x}_i$, we train the model to produce the emotion space parameter vector $\mathbf{y}_i$,

$$\mathbf{y}_i = [\mu_a,\ \mu_v]. \quad (2)$$

Shown in Table 2, are the results for employing the learned features for multiple linear regression. Features are first extracted on 20msec intervals, and then appropriately aggregated to match the one second intervals of our labels. Results for these features which are learned from single frames are shown as DBN-SF. We additionally show the KL-divergence for the Gaussian ground-truth representation used in prior work [5, 18]. Where we develop regressors to predict the parameterization vector $\mathbf{y}_i$ of a two-dimensional Gaussian in A-V space,

$$\mathbf{y}_i = [\mu_a,\ \mu_v,\ \sigma_{aa}^2,\ \sigma_{vv}^2,\ \sigma_{av}^2]. \quad (3)$$

6.2 Multi-frame Feature Learning

While future work on more sophisticated fine-tuning approaches or better stochastic models in pre-training may improve performance, the largest issue is the inherent limitation in using a single short-time window. Human emotional associations necessarily require more than a ∼20ms short-time window, and thus future approaches must take into account the variation of acoustic data over a larger period of time. In these experiments we investigate the development of models that incorporate multiple spectral

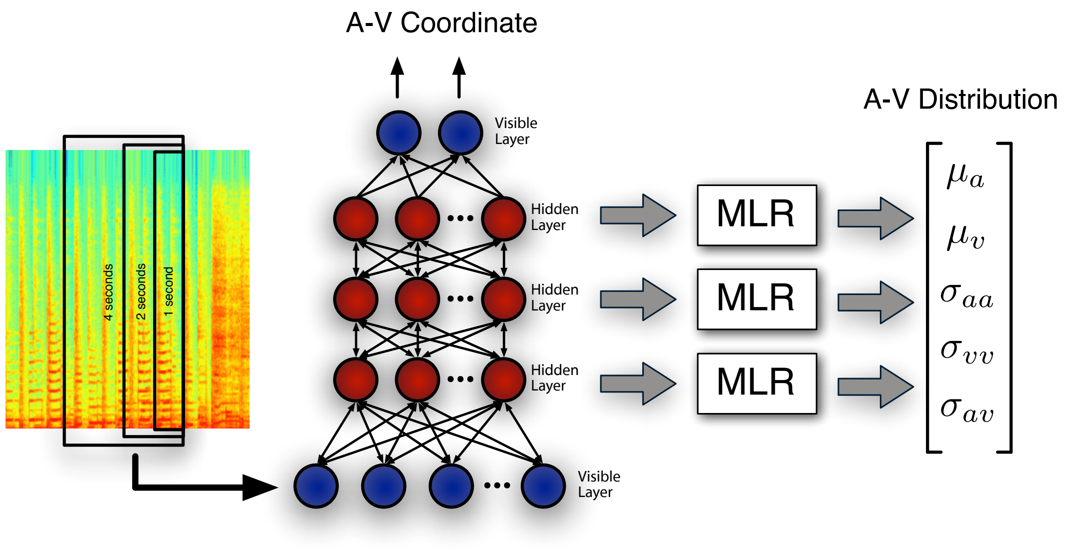


Figure 3. Feature learning system architecture showing the temporal aggregation, deep belief network and subsequent training of linear regressors to predict multi-dimensional A-V distributions.

windows to derive musical emotion. Taking spectral aggregations of the past one second, past two seconds, and past four seconds, we concatenate the resulting vectors as inputs to the system. As each spectrum frame is 257-dimensional vector, the total DBN input is now 771 dimensions. A diagram showing the multi-rate temporal integration, DBN training and linear regression to the emotion space is shown in Figure 3. Results for multi-frame (MF) feature learning can be found in Table 2 labeled as DBN-MF.

For this new approach, we provide visualizations of the learned features. Figure 4 shows the input spectrogram in log-magnitude for proper visualization, though we do not take the log for the actual model input.

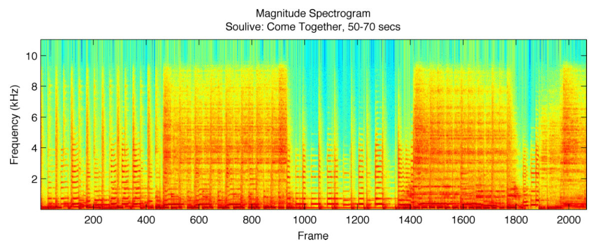


Figure 4. Log-magnitude spectrogram of input audio.

In the original spectrogram (Figure 4) we see the verse transition into the first chorus of the Soulive rendition of the Beatles song *Come Together* starting around frame 500. We see a similar pattern in the spectrogram between frames 1488-1800, which is the only other part of the clip where the percussion includes cymbals. Shown in Figure 5 are the resulting features from the intermediary layer outputs. Note that the structural information in the spectrogram is retained in the hidden layer outputs rendered in Figure 5.

We also wish to investigate the reconstruction of the original input aggregated spectrogram from the hidden layer outputs. Figure 6 depicts this reconstruction which

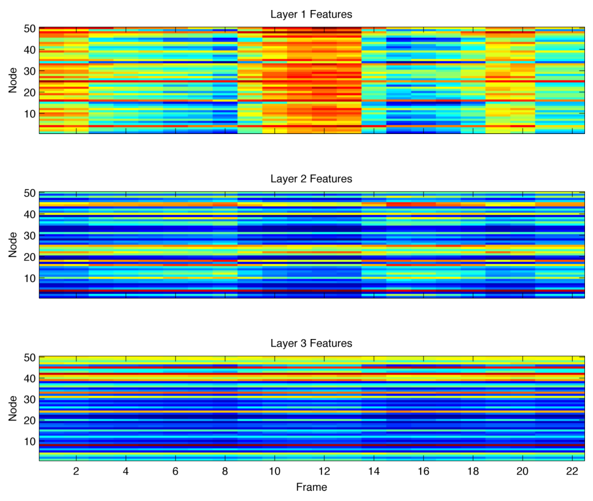


Figure 5. DBN hidden layer outputs using the aggregate spectral frames as input.

was generated using the method outlined in previous work [8]. Due to the concatenations of multiple time scale aggregations, we adjust the y-axis to display the correct frequency values for each. The top contains the last one second aggregations, below that is the aggregation from the last two seconds, and the last four seconds is at the bottom.

6.3 Universal Background Model Feature Learning

In order to improve our results with the multi-frame approach, we seek to harness the power of our much larger unlabeled music dataset. As DBN training relies on a two step training process, the first of which is unsupervised, there is no reason we should not use every piece of available data. In training our RBMs with our larger dataset, we get a much more accurate portrait of the distribution of music, and therefore create a much more accurate music model, which we can then finetune for musical emotion,

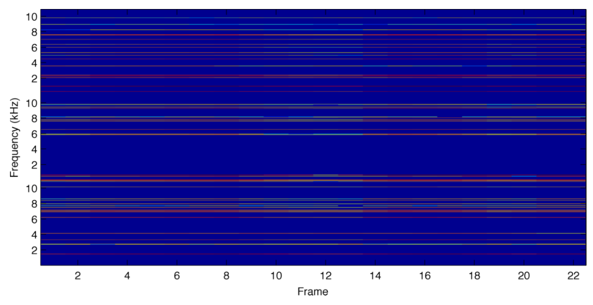


Figure 6. Reconstruction of the original aggregated spectrogram used as the DBN input. Top is last one second aggregates, middle last 2 seconds, bottom last 4 seconds.

or any other supervised machine learning problem. As this model is a general music model, we refer to it as a universal background model (UBM). For our larger dataset we use the uspop2002 dataset in its entirety, which contains nearly 8000 songs. Even after aggregating our spectra at one-second intervals this adds up to ∼26 GB of training data. Results for the universal background model approach are shown in Table 2 as DBN-UBM.

Feature Type	Average Mean Distance	Average KL Divergence
MFCC	0.140 ± 0.005	1.28 ± 0.157
Chroma	0.182 ± 0.006	3.33 ± 0.294
Spectral Shape	0.153 ± 0.006	1.51 ± 0.160
Spectral Contrast	$\mathbf{0.138 \pm 0.005}$	$\mathbf{1.29 \pm 0.160}$
ENT	0.151 ± 0.006	1.41 ± 0.175
DBN-SF Model Error	0.203 ± 0.009	-
DBN-SF Layer 1	0.138 ± 0.005	1.25 ± 0.142
DBN-SF Layer 2	$\mathbf{0.133 \pm 0.004}$	$\mathbf{1.19 \pm 0.129}$
DBN-SF Layer 3	0.133 ± 0.002	1.21 ± 0.180
DBN-MF Model Error	0.194 ± 0.032	-
DBN-MF Layer 1	0.131 ± 0.006	1.15 ± 0.106
DBN-MF Layer 2	0.131 ± 0.004	1.14 ± 0.107
DBN-MF Layer 3	$\mathbf{0.129 \pm 0.004}$	$\mathbf{1.12 \pm 0.114}$
DBN-UBM Model Error	0.140 ± 0.015	-
DBN-UBM Layer 1	0.129 ± 0.006	1.12 ± 0.091
DBN-UBM Layer 2	0.128 ± 0.004	1.13 ± 0.097
DBN-UBM Layer 3	$\mathbf{0.128 \pm 0.004}$	$\mathbf{1.11 \pm 0.090}$

Table 2. Emotion regression results for fifteen second clips. DBN-SF are features learned from single frames (SF), DBN-MF are features learned from multi-frame (MF) aggregations, and DBN-UBM are features learned with a universal background model (UBM) approach to DBN pretraining. KL-divergence is not applicable to model error.

7. DISCUSSION AND FUTURE WORK

In looking at the first set of results for learning features from single frames (DBN-SF), we see second layer features perform best for this method, outperforming spectral contrast, which is the best performing standard feature. This result is consistent with prior work [8], though here we find the DBN-SF features to be better than spectral contrast both in predicting single points and distributions. In that work we found the DBN features to be more accurate in terms of mean prediction and spectral contrast to perform slightly better in terms of KL, though we strongly emphasized an incorrect mean to be a much worse an error than an incorrectly sized or rotated covariance.

In trying to improve our features by including multiple timescales we see improvement in mean error from 0.133 to 0.129, which is encouraging. In analyzing the reconstructed spectra from first layer features, we get a very interesting result, which is similar to our prior work with [8]. The overall representation if very sparse in terms of frequency and seems to target very specific frequencies to contribute to the overall emotion features. Analyzing the features in Figure 5, we note that there is most definitely an emotion change as we progress from the slower and heavily minor sounding verse into the higher tempo rock chorus. We see changes reflected in all three layers' features in that area of the clip. We also note that it appears as if the features don't exactly line up with the spectrogram, which is a result of including past data in our feature computation. When the spectrum changes abruptly it takes several frames for our model to catch up. We do not see this as a limitation as humans have a reaction time too, which perhaps is reflected in the fact that these features are better suited for time-varying emotion prediction.

At a normalized error of 0.128, our simple MLR method with DBN features outperforms two of the three features investigated in our prior work with conditional random fields (CRFs) [19], which is a much more sophisticated method. Furthermore, while the performance increase between the third layer features of DBN-MF and DBN-UMB is small, the performance of the DBN model itself is reduced from 0.194 to 0.140, which we find to be highly encouraging. These results indicate that UBM pretraining is providing us a model that is much better suited for emotion finetuning.

In future work, we plan to investigate shrinking layer sizes in the UBM approach where we can perhaps take better advantage of the dimensionality reduction power of the RBM. Furthermore, we see that it may be interesting to investigate multiple stages of finetuning. We would first follow the approach of [7] for reducing the dimensionality of unlabeled data. It may be possible to gain a more accurate UBM by applying a finetuning stage that involved unraveling the model to reconstruct the unlabeled data. Those model parameters could then be adapted to emotion, or any other type of music prediction.

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] G. Tzanetakis, "Marsyas submissions to MIREX 2007," MIREX 2007.

[3] J.-C. Wang, H.-Y. Lo, S.-K. Jeng, and H.-M. Wang, "Mirex 2010: Audio classification using semantic transformation and classifier ensemble," in *MIREX*, 2010.

[4] 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.

[5] E. M. Schmidt and Y. E. Kim, "Prediction of time-varying musical mood distributions from audio," in *ISMIR*, Utrecht, Netherlands, 2010.

[6] J. A. Russell, "A complex model of affect," *J. Personality Social Psychology*, vol. 39, pp. 1161–1178, 1980.

[7] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, July 2006.

[8] E. M. Schmidt and Y. E. Kim, "Learning emotion-based acoustic features with deep belief networks," in *WASPAA*, New Paltz, NY, 2011.

[9] E. M. Schmidt, M. Prochup, J. Scott, B. Dolhansky, B. G. Morton, and Y. E. Kim, "Relating perceptual and feature space invariances in music emotion recognition," in *CMMR*, London, U.K., 2012.

[10] H. Lee, Y. Largman, P. Pham, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *NIPS*. MIT Press, 2009.

[11] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks," in *ISMIR*, Utrecht, Netherlands, 2010.

[12] Y. E. Kim, E. Schmidt, and L. Emelle, "MoodSwings: A collaborative game for music mood label collection," in *ISMIR*, Philadelphia, PA, September 2008.

[13] 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.

[14] 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*, vol. 1, 2002, pp. 113–116.

[15] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, pp. 357–366, 1980.

[16] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[17] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *NIPS*. MIT Press, 2007.

[18] E. M. Schmidt and Y. E. Kim, "Prediction of time-varying musical mood distributions using Kalman filtering," in *Proc. of the 9th IEEE Intl. Conf. on Machine Learning and Applications (ICMLA)*, Washington, D.C., 2010.

[19] ——, "Modeling musical emotion dynamics with conditional random fields," in *ISMIR*, Miami, FL, 2011.

USER-CENTERED MEASURES VS. SYSTEM EFFECTIVENESS IN FINDING SIMILAR SONGS

Xiao Hu
Faculty of Education
The University of Hong Kong
xiaoxhu@hku.hk

Noriko Kando
National Institute of Informatics
Japan
kando@nii.ac.jp

ABSTRACT

User evaluation in the domain of Music Information Retrieval (MIR) has been very scarce, while algorithms and systems in MIR have been improving rapidly. With the maturity of system-centered evaluation in MIR, time is ripe for MIR evaluation to involve users. In this study, we compare user-centered measures to a system effectiveness measure on the task of retrieving similar songs. To collect user-centered measures, we conducted a user experiment with 50 participants using a set of music retrieval systems that have been evaluated by a system-centered approach in the Music Information Retrieval Evaluation eXchange (MIREX). The results reveal weak correlation between user-centered measures and system effectiveness. It is also found that user-centered measures can disclose difference between systems when there was no difference on system-effectiveness.

1. INTRODUCTION

With the rapid growth of digital music, research on Music Information Retrieval (MIR) has been flourishing in recent years. Many algorithms and systems have been developed to facilitate searching and retrieving music pieces automatically. As a crucial aspect of system development, evaluation of MIR systems has attracted continuous attention among researchers. However, so far, MIR evaluation has been dominated by system-oriented approaches, while users, whom MIR systems would ultimately serve, have rarely been considered in MIR evaluation.

The system-centered evaluation approach, also known as the Cranfield evaluation [3], has been adopted by the Music Information Retrieval Evaluation eXchange (MIREX), a community-based annual evaluation campaign for MIR. Since its inception in 2005, MIREX has evaluated and compared more than a thousand systems on various MIR tasks such as genre classification, artist identification, query-by-humming, etc. [5]. MIREX not only greatly enhances the development of MIR, but also provides rich evaluation data on system effectiveness.

Despite its long tradition and popularity, system-centered approach has been criticized for excluding users from the evaluation process. Researchers argue that the goal of MIR systems is to facilitate users' music information tasks, and thus the evaluation of MIR should inevitably take users into consideration [8]. Furthermore, since music appreciation is more or less a subjective process, users' perceptions about whether a MIR system is useful might be different from a system-centered point of view. However, there have been no formal studies investigating whether there are correlations between user-centered measures and system effectiveness measures in the MIR domain. Thus, people remain puzzled when they see the precision and recall numbers of certain systems. Would they be helpful to users? Would users be satisfied with them? This study aims to fill the research gap and answer the following research question: *to what extent is system effectiveness related to user-centered measures?*

In particular, this study focuses on one MIR task, audio music similarity and retrieval where systems search for songs similar to a given query song. There are two major reasons for choosing this task. First, finding similar songs, as a query-by-example scenario, is a prevailing music information need. For instance, many people search for similar songs to build playlists [4]. Second, the MIREX has the same task and thus provides system-centered measures needed in this study.

2. RELATED WORK

2.1 User Evaluation in MIR

There have been very few studies on formal user evaluation MIR systems. The Philips Research Laboratories in the Netherlands is the leader on this topic. During 2002 to 2005, they conducted a series of controlled user experiments to evaluate their playlists generation systems [9][10][14]. The general approach was to recruit 22 to 24 participants to use a novel system they developed as well as one or two control systems for the task of generating playlists in some pre-defined music listening situations (e.g., "soft music" and "lively music"). The experiments may consist of one to four sessions. The researchers then compared the novel system to control systems using user-centered measures including users' ratings on playlist quality, time spent on the task, number of button presses in accomplishing the task, as well as perceived usefulness, ease-of-use and preference reported by the users.

A more recent study by Hoashi and colleagues [7] compared the effectiveness of three visualization methods for a content-based MIR system. Besides user effectiveness and user satisfaction, they also employed self-reported usability measures on perceived system accuracy, explicitness and enjoyability. It is particularly valuable that the authors also advocated for the user-centered approach as necessary to evaluate MIR systems.

2.2 User and System Effectiveness in Text Information Retrieval

Evaluating retrieval systems from the users' perspective has been active in the domain of text Information Retrieval (IR). Studies have been done to examine the relationship between user-centered measures and system-effectiveness. Hersh et al. investigated this question in the tasks of instance recall [6] and question answering [12]. They conducted user experiments and found user effectiveness and system effectiveness did not yield the same conclusion. More recently, Turpin and Scholer [13] evaluated systems with large differences in system effectiveness and again found no significant relationship between user and system effectiveness for precision-based tasks and a weak relationship for recall-based tasks.

In contrast, there were also studies finding significant correlations. For instance, Allen et al. [1] studied the task of text passage retrieval and found user effectiveness (as measured by task completion time and number of relevant passages) was correlated with system effectiveness when the latter was either low or high, but not in the middle. Last but not least, Al-Maskari et al. [2] controlled the variance of system effectiveness and reported significant correlations between multiple user-centered measures and system effectiveness.

As a first study investigating the relationship between user-centered measures and system effectiveness in the MIR domain, this study is inspired by the aforementioned previous work in text IR. Many of these studies used TREC (Text Retrieval Conference) evaluation results to select systems to be evaluated by users and to obtain data on system effectiveness. In this study, we resort to MIREX, the counterpart of TREC in the MIR domain, for obtaining system effectiveness measures and the underlying MIR systems.

3. METHOD AND RESEARCH DESIGN

3.1 The AMS Task in MIREX

The MIREX has included the Audio Music Similarity and Retrieval (AMS) task every year except for 2008. In this task, systems are given a number of queries (i.e. audio song clips) and a large collection of music audio clips. The goal of the systems is to retrieve clips from the collection that sound similar to the queries. In 2010, the AMS task had 100 queries sampled from 10 different genres, and the candidate collection contained 7,000 music clips also evenly sampled from the 10 genres. There were eight systems evaluated in this task[1] and all of them were considered in this study except for one system (RZ1) which was a random baseline and performed poorly. For each query, the top five song clips retrieved by each system were collected for similarity judgment by human experts. This is much like the pooling method for relevance judgment in TREC [15]. However, unlike TREC, the pooled candidates were deliberately randomized when presented to the human judges so as to eliminate any cues given by the order of candidates. In the 2010 cycle of the AMS task, each query candidate pair was judged by one assessor. Based on the similarity judgments on the pooled candidates, system effectiveness measures were calculated for evaluating and comparing the systems.

This study is built upon the 2010 cycle of the MIREX AMS task. For each query, two systems were selected and user-centered measures on both systems were collected in formal user experiments. Then the research question is answered by comparing the user-centered measures to system effectiveness.

3.2 The Systems

In selecting systems, we adopted the approach proposed by Al-Maskari et al. [2]: different systems are selected for different queries so that the differences of system effectiveness between systems can be better controlled. As there are no previous studies of this kind in the MIR domain, we also followed [2] in using average precision (AP) as the system effectiveness measure. AP is calculated as the mean of precisions at the point of each relevant document in the ranked sequence. This measure rewards relevant documents retrieved at high ranks. In MIREX AMS task, human judges evaluated each candidate using ternary relevance: very similar, somewhat similar, and not similar. In calculating AP scores in this study, we convert the judgments into binary relevance by combining "very similar" and "somewhat similar" into "similar". In the future we will evaluate system effectiveness measures based on ternary relevance.

The AP scores of the seven participating systems vary across queries. For 79 out of the 100 queries, the differences of AP values among systems are from 0.01 to 0.6. For the rest of 21 queries, the systems had exactly the same AP. Unlike [2] where the best and worst performing systems were chosen for each query, we choose the best performing system (denoted as the "superior" system in this paper) and the second best system (denoted as the "inferior" system in this paper) for each query. This is because the systems tend to have lower AP than those in [2] (99% are lower than 0.5 in this study), and the difference between the best and worst performing systems can be very obvious, making it a trivial task to decide system preference. In addition, the worst performing systems sometimes are so bad that our pilot testers felt it was boring to listen to songs very dissimilar to the queries. Finally, using systems with different but close AP scores makes it possible to investigate whether user-centered measures can differentiate system quality when system effectiveness had little difference. For the 21 queries without system difference, two systems were randomly

[1] The MIREX 2010 AMS task results: http://www.music-ir.org/mirex/wiki/2010:Audio_Music_Similarity_and_Retrieval_Results

selected. Figure 1 shows the AP scores of the two selected systems across queries where the queries are ordered by the difference of AP scores between the two systems.

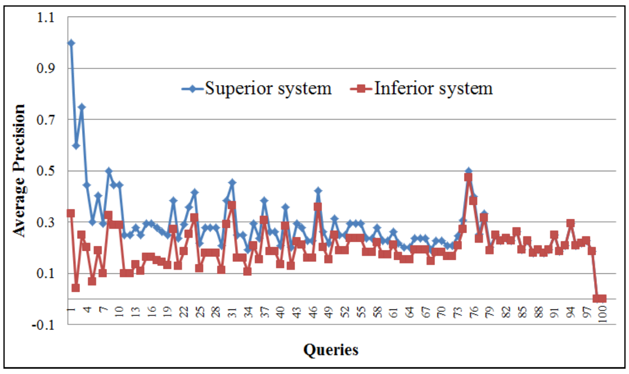


Figure 1. System AP scores across queries.

3.3 Participants

50 Japanese undergraduate and graduate students from 13 different universities were recruited, including 24 females and 26 males. Their average age was 21.7 years old (standard deviation was 4.30, range from 18 to 50). Their majors ranged from engineering, medicine to social sciences and humanities. Statistics of participants' background on music knowledge, computer and English skills, as well as familiarity with the genres of the songs are shown in Table 1. Self-reported English abilities were collected because some of the songs had English lyrics and the pre- and post- experiment questionnaires were written in English. As the songs were associated with American genre classification system, the participants' familiarity levels with the genres were collected.

	Median	Max.	Min.
Music knowledge*	4	6	2
Expertise with computers*	4	6	3
Expertise with online searching*	5	6	2
Ability in reading English*	5	7	3
Ability in listening to English*	4	7	2
Familiarity with the genres ǂ	3	5	0

*: in a Likert scale from 1 to 7. 1: novice, 7: expert; ǂ: in a Likert scale from 1 to 5. 1: very unfamiliar, 5: very familiar

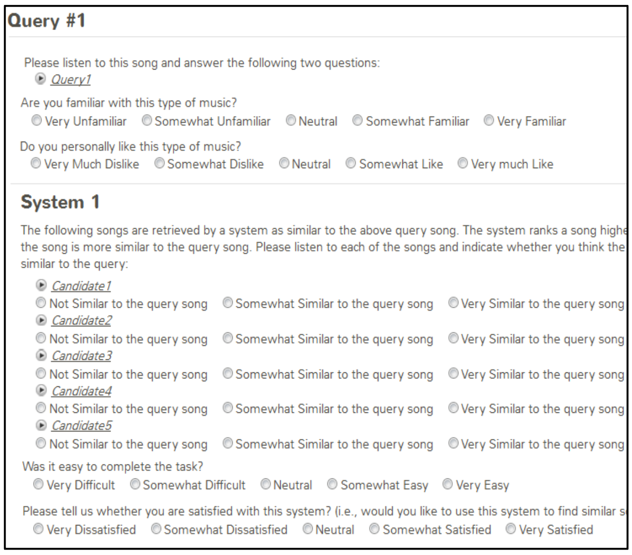
Table 1. Statistics of participants' background.

3.4 Tasks

All of the 100 queries in the MIREX AMS tasks are included in this study. The queries are evenly distributed into ten different genres, namely Baroque, Romantic, Classical, Country, Jazz, Blues, RocknRoll, Rap/HipHop, Metal, and Edance. Each participant was assigned ten queries with one in each genre. Since there are 50 participants, each query was evaluated by five participants. The orders of the ten query genres were distributed to participants using a Latin Square design so as to reduce the effect of genre order on results.

For each query, a participant evaluated the list of candidate songs retrieved by the two selected systems. Specifically, a participant needed to play and listen to the query song and indicate his/her familiarity level with it, as well as his/her personal preference on the query. Then he or she proceeded to listen to each of the five songs returned by one system and indicate whether it sounded similar to the query. Just like in MIREX, we used ternary similarity scale: participants needed to indicate whether a candidate was very similar, somewhat similar or not similar to the query. In this experiment, the candidate songs were presented to users in the original ranked order as retrieved by the systems. This setting mirrors a real life retrieval system where a higher ranked item is expected to be more relevant. In contrast, MIREX human judgments had no information on the rank of the candidates nor did they know which candidates were retrieved by the same system.

After evaluating all the five songs returned by one system, the participant was asked to indicate his/her satisfaction level towards this system and the perceived task easiness in a Likert scale. Then the participant proceeded to the other system. The relative difference of the systems was not revealed to the user and the order of two systems was randomized. After listening to candidates from both systems, the participant was asked to indicate his or her preference between the systems and how easy it was to compare the two systems. The audio of each song (either query or candidate) was 30 second long, but it could be paused and/or replayed at any time. A participant could also change answers when working with one system. However, once proceeding to the other system or the next query, a participant could not go back to change answers. This is to prevent influences from other systems on users' judgments. A screenshot of the working interface is shown in Figure 2.


Query #1

Please listen to this song and answer the following two questions:
Query1

Are you familiar with this type of music?
○ Very Unfamiliar ○ Somewhat Unfamiliar ○ Neutral ○ Somewhat Familiar ○ Very Familiar

Do you personally like this type of music?
○ Very Much Dislike ○ Somewhat Dislike ○ Neutral ○ Somewhat Like ○ Very much Like

System 1

The following songs are retrieved by a system as similar to the above query song. The system ranks a song highe
the song is more similar to the query song. Please listen to each of the songs and indicate whether you think the
similar to the query:

Candidate1
○ Not Similar to the query song ○ Somewhat Similar to the query song ○ Very Similar to the query song
Candidate2
○ Not Similar to the query song ○ Somewhat Similar to the query song ○ Very Similar to the query song
Candidate3
○ Not Similar to the query song ○ Somewhat Similar to the query song ○ Very Similar to the query song
Candidate4
○ Not Similar to the query song ○ Somewhat Similar to the query song ○ Very Similar to the query song
Candidate5
○ Not Similar to the query song ○ Somewhat Similar to the query song ○ Very Similar to the query song

Was it easy to complete the task?
○ Very Difficult ○ Somewhat Difficult ○ Neutral ○ Somewhat Easy ○ Very Easy

Please tell us whether you are satisfied with this system? (i.e., would you like to use this system to find similar s
○ Very Dissatisfied ○ Somewhat Dissatisfied ○ Neutral ○ Somewhat Satisfied ○ Very Satisfied

Figure 2. Screenshot of the evaluation interface.

3.5 Procedure

The experiment was conducted in a batch manner, with 5 to 7 subjects in each batch performing the tasks at the same time. Before the experiment started, each subject read and signed a consent form. After that, she or he filled an online pre-experiment questionnaire with regard to demographic information, music background and search experience. Then, the experiment facilitator introduced the experiment system and the experiment procedure in Japanese. The training sessions lasted about 10 minutes.

According to our pre-tests of the procedure, the participants were given 55 minutes to finish all the 10 assigned queries. Most participants finished the process within 45 minutes. During the process, the experiment system recorded users' interactions including play and pause queries and candidates, answers to each question as well as changes of answers. After all queries were finished, each subject filled an online post-experiment questionnaire which asked for his or her general impression on the evaluated music retrieval systems and the experiment in general. The entire procedure lasted about 1.5 hours and each participant was paid 2000 yen for their participation.

3.6 User-centered Measures

The following user-centered measures were collected and compared to system effectiveness.

User effectiveness:

1) Number of similar songs found using each system. A candidate is "similar" to a query if the user chooses "very similar" or "somewhat similar" option.
2) Task completion time: time spent on making judgments on all candidates of one system.
3) Time spent in finding the first similar song using each system. If there is no similar song found among the five candidates, the time is assumed to be 3 minutes which is the time needed for listening to 6 candidates in full length.
4) Rank of the first similar song using each system. If there is no similar song found among the five candidates, the rank is assumed to be 6.

User perception:

1) Task easiness in evaluating results of each system.
2) User satisfaction with each system.
3) Easiness in comparing two systems.

Each of these measures was on a Likert scale from 1 to 5, with 1 indicating very difficult/very dissatisfied and 5 indicating very easy/very satisfied.

User preference:

1) The system a user prefers: the superior one, inferior one or neither.

3.7 Hypotheses

To answer our research question, we compared AP scores of the two systems to the aforementioned user-centered measures by testing a series of hypotheses:

H1: When the systems' AP scores were different, users were more effective and more satisfied with the superior systems than the inferior systems;

H2: When the systems' AP scores were the same, users were similarly effective and satisfied with both systems;

H3: When the systems' AP scores were different, users preferred the superior systems to the inferior systems;

H4: When the systems' AP scores were the same, users did not have a preference between the systems;

H5: User perceived higher easiness level when comparing systems with AP score difference than comparing those without AP score difference.

To test the correlation between user-centered measures and AP score, we examined the following hypotheses:

H6: User-centered measures are highly correlated with AP score;

H7: When the difference of systems' AP scores gets larger, users would tend to prefer the superior systems and feel it is easier to compare the two systems.

4. RESULS AND DISCUSSIONS

4.1 User Effectiveness and Satisfaction

Table 2 presents means and standard deviations (in parenthesis) of AP scores and user effectiveness and perception measures for the superior and inferior systems on the 79 queries where the two systems had different AP scores. In order to test *H1*, we employed the non-parametric Wilcoxon signed rank sum test because studies have shown that system performance data rarely comply with normal distribution [5] and the Wilcoxon test does not assume normal distribution of tested variables.

Measure	Superior	Inferior	*p* value
average precision	0.30 (0.13)	0.20(0.08)	< 0.001*
number of similar songs	3.53 (0.99)	3.00 (1.07)	< 0.001*
task completion time (seconds)	76.75 (21.99)	77.09 (25.44)	0.688
time finding first similar song (seconds)	19.91 (14.42)	28.55 (26.74)	0.042*
rank of first similar song	1.48 (0.63)	1.72 (0.99)	0.047*
task easiness	3.47 (0.55)	3.48 (0.53)	0.714
user satisfaction	3.44 (0.73)	3.04 (0.77)	< 0.001*

N=79. *: significant at $p < 0.05$ level

Table 2. Measures for queries with different AP scores.

As shown in Table 2, the difference between the AP scores of the superior and inferior systems was significant across the 79 queries. The user-centered measures indicate that, using the superior systems, users found more similar songs, spent less time in finding the first similar song which had a higher rank, and were more satisfied than using the inferior systems. However, there was little difference on the time used to judge all the five candidates of each system. In addition, users perceived the tasks were about the same easiness level when using both systems.

Therefore, hypothesis *H1* is partially supported by four out of six user-centered measures under consideration. In other words, when the AP scores were significantly different, some user-centered measures could also differentiate the systems. The little differences on task completion time and perceived task easiness are related to each other. If a task is difficult, it will likely take more time. The insignificant result indicates systems with higher AP scores did not make the task of music similarity judgment easier.

Table 3 presents means and standard deviations (in parenthesis) of the aforementioned measures and Wilcoxon test results for the two systems on the 21 queries where the two systems had the same AP scores. As can be seen from Table 3, *H2* is also partially supported by four out of six user-centered measures. That is, the two systems had no significant difference on number of similar songs found, task completion time, rank of first similar songs and perceived task easiness. However, the two systems were significantly different in terms of time spent finding the first similar songs and users' satisfaction towards systems, even though the AP scores of the two systems were exactly the same. This evidences that some user-centered measures can tell the differences between systems that system-effectiveness cannot. In particular, the difference on user satisfaction on systems with the same AP scores is remarkable since user satisfaction has been called by IR researchers as a main criterion of IR system evaluation (e.g., [11]).

Measure	System 1	System 2	*p* value
average precision	0.20 (0.07)	0.20 (0.07)	-
number of similar songs	3.59 (0.95)	3.46 (1.01)	0.470
task completion time (seconds)	78.61 (21.19)	78.83 (24.49)	0.776
time finding first similar song (seconds)	14.37(10.83)	24.50 (18.27)	0.009*
rank of first similar song	1.27 (0.47)	1.50 (0.63)	0.197
task easiness	3.22 (0.43)	3.12 (0.54)	0.616
user satisfaction	3.37 (0.64)	3.09 (0.61)	0.032*

N=21. *: significant at $p < 0.05$ level

Table 3. Measures for queries with same AP scores.

4.2 User Preference

Statistics on user preferences between the systems are shown in Table 4. It is interesting to see that users perceived no difference between the two systems 25% of the time while the AP scores of the systems were different. In contrast, for queries where the two systems had exactly the same AP scores, 80% of users thought the systems were different. A Wilcoxon test was conducted on each set of the queries to see if the differences on number of votes of the two systems were significant. The results support both *H3* and *H4*: users preferred the superior systems when the systems' AP scores differed ($p < 0.001$) while users did not have significant preferences when the systems had the same AP scores ($p = 0.06$). However, the low *p* value and the low percentage of "no preference" votes on queries with the same AP scores (20%) indeed suggest that the difference on AP scores may not be a good indicator of system preference.

4.3 Perceived Easiness in Comparing Systems

The test results on hypothesis *H5*, perceived easiness level in comparing the two systems are shown in Table 5. The average easiness score is 3.06 (easier) across the 79 queries with AP difference and 2.81 (harder) across the 21 queries without AP difference. As the two sample sizes are not equal, a two sample unequal variance *t*-test was employed to test the significance of the difference on easiness level. The test result is significant and thus hypothesis *H5* is supported: users perceived it was easier to compare the two systems when there were AP differences between the systems. So far, the results of the analysis generally support our hypotheses *H1* to *H5*. That is, user-centered measures tend to agree with system-effectiveness (as measured by AP score). However, the exceptions in *H1* and *H2* are also noteworthy. In the next subsection, we continue to investigate the correlation between user-centered measures and AP scores.

79 queries with different AP scores	**Superior**	**Inferior**	**No pref.**	**Total**
Number of pref. votes	190	105	100	395
Percentage of preference	48.10%	26.58%	25.32%	100%
21 queries with same AP scores	**System 1**	**System 2**	**No pref.**	**Total**
Number of pref. votes	48	36	21	105
Percentage of preference	45.71%	34.29%	20.00%	100%

Table 4. Votes of system preferences.

	With AP difference	**Without AP difference**	***p* value**
difficulty level	3.06 (1.26)	2.81(1.29)	0.002*
sample size	395	105	

*: significant at $p < 0.05$ level

Table 5. Perceived difficulty in comparing systems

4.4 Correlation between User-centered Measures and System Effectiveness

To test Hypothesis *H6*, Pearson's correlation coefficients were calculated for measures on interval scales: number of similar songs, task completion time and time of finding the first similar songs. For measures on ordinal scales such as rank of first similar songs, task easiness and user satisfaction, Spearman's rank correlation coefficient was calculated. The results are shown in Table 6.

Measure	Coefficient	*p* value
number of similar songs	0.111 (Pearson)	0.059
task completion time	-0.069 (Pearson)	0.177
time finding first similar song	-0.142 (Pearson)	0.022*
rank of first similar song	-0.163 (Spearman)	< 0.021*
task easiness	0.057 (Spearman)	0.423
user satisfaction	0.246 (Spearman)	< 0.001*

N = 200. *: significant at $p < 0.05$ level

Table 6. Correlation between user-centered measures and AP score.

Number of similar songs, task completion time and task easiness have no significant correlation with AP score while the correlation between AP score and other user-centered measures are fairly weak despite being significant. Our hypothesis *H6* is not supported. The fact that there is no significant relationship between perceived task easiness and AP score confirms an earlier finding in Section 4.1 that higher AP scores did not make the task of music similarity judgment easier.

Table 7 shows Spearman's correlation coefficients between the AP scores difference of the two systems and the two user-centered measures related to system comparison: system preference and easiness in system comparison. System preference is encoded as an ordinal variable with values 1, 0, and -1 indicating preferring the superior system, no preference, and preferring the inferior system, respectively. From Table 7 we can see that hypothesis *H7* is not supported: the correlations are either insignificant or fairly weak. The insignificance between system preference and AP score difference helps explain an earlier observation that 80% of the users indicated system preference when there was no difference on the AP scores.

Measure	Correlation with AP difference	*p* value
system preference	0.080 (Spearman)	<0.053
easiness in system comparison	0.174 (Spearman)	<0.001*

N=100. *: significant at $p < 0.05$ level.

Table 7. Correlation between user-centered measures and AP score difference

5. CONCLUSIONS AND FUTURE WORK

This paper presents a user experiment on evaluating results of music similarity retrieval systems in the AMS task in MIREX 2010, with the goal of comparing a well-accepted system effectiveness measure to user-centered measures. Such comparison has rarely been explored in the MIR domain. The results revealed none or weak correlations between system effectiveness and eight user-centered measures. In particular, significant differences on two user-centered measures, including user satisfaction, were found between systems with the same system effectiveness. As a first study on user-centered vs. system-centered measures in MIR, this research prompts many interesting observations for future research. More user behavior measures can be examined such as number of times a query song was played, number of changes a user made to his or her answers, as well as measures based on ternary relevance judgment. In addition, similar evaluations could be done for other MIR tasks such as genre and mood classification in the future.

6. ACKNOWLEGEMENTS

The research is partially supported by the JSPS Grand-in-Aid (#21300096) and the Non-MOU Grant funded by the National Institute of Informatics in Japan, and was conducted when the first author was in the University of Denver. We also thank the IMIRSEL in the University of Illinois for providing the MIREX AMS data.

7. REFERENCES

[1] J. Allan, B. Carterette and J. Lewis: "When will information retrieval be 'good enough'? User effectiveness as a function of retrieval accuracy," *Proceedings of the ACM SIGIR Conference on Information Retrieval*, pp. 433-440, 2005.

[2] A. Al-Maskari, M. Sanderson, P. Clough and E. Airio: "The good and the bad system: Does the test collection predict users' effectiveness?"*Proc. of the ACM SIGIR Conference*, pp. 59-66, 2005.

[3] C. W. Cleverdon and E. M. Keen: "Factors determining the performance of indexing systems," Vol. 1: Design. Vol 2: Results. Cranfield, U.K: *Aslib Cranfield Research Project*, 1966.

[4] 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 7th International Conference on Music Information Retrieval (ISMIR)*, 2006.

[5] J. S. Downie: "The Music Information Retrieval Evaluation eXchange (2005–2007): A window into music information retrieval research," *Acoustical Science and Technology*, 29(4), pp. 247–255. 2008.

[6] W. Hersh, A. Turpin, S. Price, B. Chan, D. Kraemer, L. Sacherek, and D. Olson: "Do batch and user evaluations give the same results?" *Proc. of the ACM SIGIR Conference*, pp. 17–24, 2000.

[7] K., Hoashi, S. Hamawaki, H. Ishizaki, Y. Takishima, and J. Katto: "Usability evaluation of visualization interfaces for content-based music retrieval systems," *Proc. of ISMIR*, 2009.

[8] X. Hu, and J. Liu: "Evaluation of music information retrieval: towards a user-centered approach," *Proceedings of the 4th Workshop on Human-Computer Interaction and Information Retrieval (HCIR)*, 2010.

[9] S. Pauws, and B. Eggen: "PATS: Realization and evaluation of an automatic playlist generator," *Proc. of ISMIR*, 2002.

[10] S. Pauws, and S. van de Wijdeven: "User evaluation of a new interactive playlist generation concept," *Proc. of ISMIR*, 2005.

[11] K. Spärck Jones: *Information Retrieval Experiment*, London, Butterworths. 1981.

[12] A. Turpin and W. Hersh: "Why batch and user evaluations do not give the same results," *Proc. of the ACM SIGIR Conference*, pp. 225-231, 2001.

[13] A. Turpin and F. Scholer: "User performance versus precision measures for simple search tasks," *Proc. of the ACM SIGIR Conference*, pp.11-18, 2006.

[14] F. Vignoli and S. Pauws: "A music retrieval system based on user driven similarity and its evaluation," *Proc. of ISMIR*, 2005.

[15] E. M. Voorhees and D. K. Harman: *TREC: Experiments in Information Retrieval Evaluation*, MIT Press, 2005.

TOWARDS TIME-RESILIENT MIR PROCESSES

Rudolf Mayer and Andreas Rauber
Secure Business Austria
Favoritenstrasse 16, 1040 Vienna, Austria
{mayer, rauber}@sba-research.at

ABSTRACT

In experimental sciences, under which we may likely subsume most research areas in MIR, repeatability is one of the key cornerstones of validating research and measuring progress. Yet, due to the complexity of typical MIR experiments, ensuring the capability of re-running any experiment, achieving exactly identical outputs is challenging at best. Performance differences observed may be attributed to incomplete documentation of the process, slight variations in data (preprocessing) or software libraries used, and others. Digital preservation aims at keeping digital objects authentically accessible and usable over long time spans. While traditionally focussed on individual objects, research is now moving towards the preservation of entire processes. In this paper we present the challenges of preserving a classical MIR process, i.e. music genre classifications, discuss the kinds of context information to be captured, as well as means to validate the re-execution of a preserved process.

1. INTRODUCTION

In many natural science disciplines, complex and data driven experiments form the basis of research. Much of the research activities carried out in the domain of Music Information Retrieval can be attributed to this type of research. Those computationally intensive experiments trigger the need for verification of the results obtained. In many cases, however, only the publication as a final result summarises the entire scientific process, predominantly in the form of results, with frequently (due to space restrictions or the complexity of the underlying process) only superficial information on the actual research and experiment process. In general, the number of experimental studies in Music Information Retrieval constitute a high number of MIR research, however, the comparability of the results is poor, due to complex scenarios, user-dependent evaluation, and the lack of data sharing. But even the re-evaluation and repeatability of experiments is low, due to data or remote services not being available, preprocessing not being documented sufficiently, or code not running or libraries utilised having changed.

A compact illustration of the experiment as it is common in research papers is not sufficient to trace the complete process of scientific research and all the sources that contributed to a result. It more often than not does not provide enough insight to allow for verification of the results obtained. In many situations it is also not possible to re-engineer experiments reported on in the literature – important details such as which exact software stack and which version of it were used, and which parameter settings were applied are often omitted or incomplete.

One step to mitigate this problem was the creation of benchmark environments. These consist minimally of annotated ground truth data as well as evaluation measures and procedures, with MIREX being the most prominent such platform in the music IR community, relying on central evaluation. To facilitate decentralised evaluation, platforms such as those proposed by [1] and [8] have been presented. However, none of these provide sufficient documentation of the process executed during the experiments, and therefore don't allow for re-applying the process to new (larger) data-sets. Publishing source the code of the algorithms used doesn't fully alleviate this problem, as subtle details in the process configuration (such as parameter settings) play an important role.

In order to tackle this increasing complexity and the orchestration of manifold services and systems, the concept of scientific workflows has received increasing attention within the research community. E-Science projects profit from the combination of automated processing steps in workflows in order to perform complex calculations and data transformations. The advantage of workflows is their capability of adding structure to a series of tasks. They can be visualized as graph representations, where nodes denote processes or tasks and edges denote information or data flows between the tasks. This adds a layer of abstraction and helps to clarify interactions between tasks[3]. Different scientific workflow management systems (SWMS) exist that allow scientists to combine services and infrastructure for their research. The most prominent examples of such systems are Taverna[6] and Kepler[4]. In the MIR domain, M2K [2] provides a specialised workflow engine, that allows users to combine certain MIR tasks in a sequence. A similar initiative is the Networked Environment for Music Analysis (NEMA) project [9], which aims at providing an execution environment for evaluation of MIR solutions.

Modelling a process in a workflow system alleviate many of the above mentioned shortcomings of insufficiently detailed experiments, as the exact sequence of processing steps, the software used and the parameter settings become explicit in the workflow definition language used. However, while the repeatability of experiments is in principle enabled by such workflow management systems, many of todays data-intensive experiments depend on a number of services and aspects of the process beyond the control of the workflow system. These may include simple aspects such as system updates – new libraries being deployed may cause experimental results to differ. The problems become even worse when considering external service such as web services. These changes are not under the control of the researcher, and may happen at a system level beyond the awareness of the individual researcher, such as e.g. a new library being installed as part of (automatic) system maintenance. This may lead to different results from the workflow, or render the workflow not executable altogether. Preserving the repeatability of such a process in a changing technological environment is thus a current and emerging topic in Digital Preservation research.

Digital Preservation is a research discipline that traditionally has focused on preserving mostly static digital objects, such as text or multimedia documents. Preservation aims at keeping these digital objects accessible and usable over a long period of time, even when technological change renders e.g. hardware or a specific operating system required unavailable, or file formats obsolete and thus not supported. More recently, digital preservation has taken steps towards preserving more complex and dynamic digital objects, among them also complete processes. The aim is to make processes archivable and allow for a later re-execution in a changed environment, while ensuring authenticity in the process results. Digital preservation of business or E-Science processes requires capturing the whole context of the process, including e.g. dependencies on other computing systems, the data consumed and generated, and more high-level information such as the goals of the process.

In this paper, we will first explore how experiments can be made more repeatable, and will then examine what is needed to preserve these processes over a longer period of time. We do this along a case study of a musical genre classification experiment, where we highlight prototypical aspects of digital preservation. The remainder of this paper is organised as follows. In Section 2, we describe the use case process, for which we then outline aspects of process preservation in Section 3. Finally, we provide conclusions and an outlook on future work in Section 4.

2. USE CASE: MUSICAL GENRE CLASSIFICATION

As an example, we consider a typical process in the MIR research community – musical genre classification, i.e. categorisation of unknown music into one of a set of predefined categories. We also consider data and ground truth

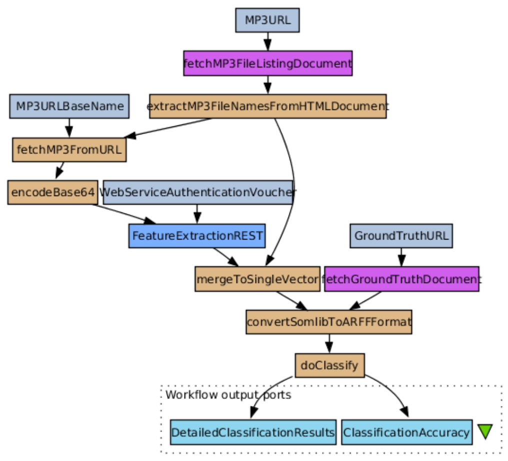


Figure 1: Musical genre classification, including fetching of data, modelled in the Taverna workflow engine

acquisition as part of the experiment, and assume that both are fetched from remote sources, e.g. a content provider such as the Free Music Archive [1] .

To simplify the implementation, we assume in this example that the data source is a simple Apache directory listing on a web-server, and the ground truth file is already compiled and can be fetched from a different web resource via HTTP. The experiment involves the following steps. First, the list of available music is fetched from the server, and each music file is downloaded from the server. For this music data, the genre assignments is downloaded from a different server. Then, a web-service is employed (via REST) to extract features from the audio files; the service accepts one file at the time. A typical example for such a service could be the ones provided by The Echonest [2] Next, the features and the genre assignments are combined into a file with the WEKA ARFF format This file and a set of parameters form the basis for learning a machine learning model with WEKA. Finally, the classification accuracy, and a detailed description of the result, are obtained

These steps are usually carried out as a (more or less defined) sequence of calls to different programs via a Linux shell, also using some constructs built-in into this shell, such as loops and simple file and string processing.

To move towards more sustainable E-Science process, we implement this process in the Taverna workflow engine [6]. Taverna is a system designed specifically to execute scientific workflows. It allows scientists to combine services and infrastructure for modelling their workflows. Services can for example be remote web-services, invoked via WSDL or REST, or local services, in the form of predefined scripts (e.g. for encoding binaries via Base64), or user-defined scripts. The latter are usually implemented by using the Taverna-supported language *beanshell*, which is based on the Java programming language.

Implementing such a research workflow in a system like Taverna yields a complete and documented model of the experiment process – each process step is defined, as is the

[1] `http://freemusicarchive.org/`
[2] `http://the.echonest.com`

sequence (or parallelism) of the steps. Further, Taverna requires the researcher to explicitly specify the data that is input and output both of the whole process, as well as of each individual step. Thus, also parameter settings for specific software, such as the parameters for the classification model or feature extraction, become explicit, either in the form of process input data, or in the script code.

Figure 1 shows the generic process described above as a specific implementation in the Taverna workflow engine. We notice input parameters to the process such as the URL of the MP3 contents and the ground truth, and also an authentication voucher which is needed to authorise the use of the feature extraction service. The latter is a bit of information that is likely to be forgotten frequently in descriptions of this process, as it is rather a technical requirement than an integral part of the scientific process transformations. However, it is essential for allowing re-execution of the process, and may help to identify potential licensing issues when wanting to preserve the process over longer periods of time, requiring specific digital preservation measures.

The fist step is t fetch a list of available MP3s, before each file is downloaded individually. Before sending the binary MP3 data to the web-service, it needs to be encoded via base64 to allow for transport via HTTP. The feature extraction is then called via Taverna's REST service interface, which requires the user to define an URL pattern for invoking the service; parameters to the service become explicit via this definition. The output of the web-service is in text form. Taverna also allows using WSDL, in which case it can infer this information from the service description directly, and the output can be typed. Note that downloading and extraction are independent steps for each file, thus these steps can and are automatically parallelised by Taverna. After a synchronisation point, i.e. when all MP3s are extracted, the features obtained for each file are merged, combined with the groundtruth, and converted to WEKA ARFF format. Finally, the classification step is performed, and the accuracy measure, and a more detailed classification report, are obtained as process outputs.

Implementing for a workflow management system comes with a certain effort, primarily to understanding the system and how process steps can be defined. Another significant effort can be needed to migrate existing scripts into the ones required by the workflow engine, especially if certain functionality is not available in both scripting languages. A positive side-effect of this migration work is that the process, in principle, becomes independent from the original execution platform. The workflow system can in many cases act as a layer of abstraction, kind of like a virtual machine, from the underlying operating system and shell available there.

2.1 Process verifiability with provenance data

During an execution of the workflow, Taverna records so-called *provenance data*, i.e. information about the creation of the objects, on the data transformation happening during the experiment. Taverna stores this information in a database, and allows to export it in the Open-Provenance Model (OPM) [7], or in the *Janus* format, and extension on the OPM that describes more details.

The top section of Listing 1 shows an example of this provenance data for the process output port *ClassificationAccuracy*. The first RDF description element defines the output port and has a reference to the second RDF description element, which contains the actual output value of 80.0, the accuracy measured in percent. The detailed classification result is then depicted in the bottom section of Listing 1. It follows the same structure, i.e. the first block defining the output port, and the second block containing the actual values, which in this case are a listing of the songs tested, and their predicted and actual values.

Such data is recorded for the input and output of each process step. It thus allows to trace the complete data flow from the beginning of the process until the end, thus enabling verification of the results obtained. This is essential for being able to verify system performance upon re-execution, specifically when any component of the process (such as underlying hardware, operating systems, software versions, etc.) have changed.

3. PROCESS PRESERVATION

While representing the process in the workflow engine in principle enables repeatability, and allows for tracing and thus verification of the results, it still does not ensure the longevity of the process. Our example musical genre classification process has several dependencies on software and services that are not under direct control of the researcher. Most prominently, the audio feature extraction web-service is operated by a third party, where changes in the functionality, or even in the availability of the service, may not be communicated at all. These thus constitute possible points of failures that may cause a process execution at a later stage to yield different results, or not being executable at all any more.

Preservation of workflows and processes has gained a lot of attention from researchers in the Digital Preservation community recently. The goal of process preservation is to allow re-executing the process at a later stage of time, when a technological change in the environment of the process has rendered the original instance of it unusable. Digital preservation of business or E-Science processes requires capturing the whole context of the process, including e.g. different or evolved enabling technologies, different system components on both hardware and software levels, dependencies on other computing systems and services operated by external providers, the data consumed and generated, and more high-level information such as the goals of the process, different stakeholders and parties.

To enable digital preservation of business processes, it is therefore required to preserve the set of activities, processes and tools, which all together ensure continued access to the services and software which are necessary to reproduce the context within which information can be accessed, properly rendered and validated.

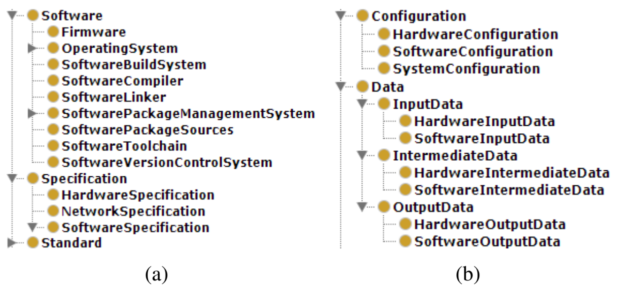
```
<rdf:Description rdf:about="{nsTaverna}/2010/workflow/{idWF}/processor/MusicClassificationExperiment/out/ClassificationAccuracy">
  <janus:has_value_binding rdf:resource="{nsTaverna}/2011/data/{idDataGrp}/ref/{idDataPort0}"/>
  <rdfs:comment rdf:datatype="{nsW3}/2001/XMLSchema#string">  ClassificationAccuracy  </rdfs:comment>
  <janus:is_processor_input rdf:datatype="{nsW3}/2001/XMLSchema#boolean">  false  </janus:is_processor_input>
  <janus:has_port_order rdf:datatype="{nsW3}/2001/XMLSchema#long">  0  </janus:has_port_order>
  <rdf:type rdf:resource="http://purl.org/net/taverna/janus#port"/>
</rdf:Description>

<rdf:Description rdf:about="{nsTaverna}/2011/data/{idDataGrp}/ref/{idDataPort0}">
  <rdfs:comment rdf:datatype="{nsW3}/2001/XMLSchema#string">  80.0 </rdfs:comment>
  <janus:has_port_value_order rdf:datatype="{nsW3}/2001/XMLSchema#long">  1  </janus:has_port_value_order>
  <janus:has_iteration rdf:datatype="{nsW3}/2001/XMLSchema#string">  []  </janus:has_iteration>
  <rdf:type rdf:resource="http://purl.org/net/taverna/janus#port_value"/>
</rdf:Description>

<rdf:Description rdf:about="{nsTaverna}/2010/workflow/{idWF}/processor/MusicClassificationExperiment/out/DetailedClassificationResults">
  <janus:has_value_binding rdf:resource="{nsTaverna}/2011/data/{idDataGrp}/ref/{idDataPort1}"/>
   ...
</rdf:Description>

<rdf:Description rdf:about="{nsTaverna}/2011/data/{idDataGrp}/ref/{idDataPort1}">
  <rdfs:comment rdf:datatype="{nsW3}/2001/XMLSchema#string">
      1   2:Hip-Hop   2:Hip-Hop        0.667  (3.359461)
      2   2:Hip-Hop   2:Hip-Hop        0.667  (3.294687)
      3  1:Classica  1:Classica        0.667  (2.032687)
      4      3:Jazz      3:Jazz        0.667  (2.536849)
      5  1:Classica  1:Classica        0.667  (1.31727)
      6  1:Classica      3:Jazz    +   0.667  (3.46771)
      7      3:Jazz  1:Classica    +   0.333  (2.159764)
      8   2:Hip-Hop   2:Hip-Hop        0.667  (3.127645)
      9      3:Jazz      3:Jazz        0.667  (3.010563)
     10   2:Hip-Hop   2:Hip-Hop        0.667  (4.631316)
  </rdfs:comment>
</rdf:Description>
```

Listing 1: Provenance data recorded by Taverna for the process outputs (cf. Figure 1). The first RDF Description element defines the output *ClassificationAccuracy*, the second element contains the actual value "80.0". The third element defines the output *DetailedClassificationResults*, the fourth element contains the actual value, one entry for each file tested, with the actual class and predicted class. Some identifiers have been abbreviated, marked by {...}



```
▼ Software
    Firmware
  ► OperatingSystem
    SoftwareBuildSystem
    SoftwareCompiler
    SoftwareLinker
  ► SoftwarePackageManagementSystem
    SoftwarePackageSources
    SoftwareToolchain
    SoftwareVersionControlSystem
▼ Specification
    HardwareSpecification
    NetworkSpecification
  ▼ SoftwareSpecification
► Standard
```

(a)

```
▼ Configuration
    HardwareConfiguration
    SoftwareConfiguration
    SystemConfiguration
▼ Data
  ▼ InputData
      HardwareInputData
      SoftwareInputData
  ▼ IntermediateData
      HardwareIntermediateData
      SoftwareIntermediateData
  ▼ OutputData
      HardwareOutputData
      SoftwareOutputData
```

(b)

Figure 2: Sections of the Context Model

To address these challenges, we have devised a context model to systematically capture aspects of a process that are essential for its preservation and verification upon later re-execution. The model consists of approximately 240 elements, structured in around 25 major groups. The model is implemented in the form of an ontology, which on the one hand allows for the hierarchical categorisation of aspects, and on the other hand shall enable reasoning, e.g. over the possibility of certain preservation actions for a specific process instance. The ontology is authored in the Web Ontology Language (OWL). We developed a set of plug-ins for the Protégé ontology editor to support easier working with the model.

This context model corresponds to some degree to the representation information network [5], modelling the relationships between an information object and its related objects, be it documentation of the object, constituent parts and other information required to interpret the object. This is extended to understand the entire context within which a process, potentially including human actors, is executed, forming a graph of all constituent elements and, recursively, their representation information. Two sections of this model are depicted in Figure 2. Each item represents a class of aspects, for which a specific instance of the context model then creates concrete members, which are then related to each other with properties.

Figure 2(a) details aspects on software and specifications. Technical dependencies on software and operating systems can be captured and described for example via CUDF (Common Upgradeability Description Format [3]) for systems which are based on packages, i.e. where there is a package universe (repositories) and a package manager application. Such an approach allows to capture the complete software setup of a specific configuration, which then can be recreated. Related to the software installed, capturing information on the licences associated to them allows for verifying which preservation actions are permissible for a specific scenario. Software and/or its requirements are formally described in specification documents. Specific documents for a process are created as instances of the appropriate class, and related to the software components they describe. Configuration, also depicted in Figure 2(a), is another important aspect, closely related to software (and hardware). Maybe even more than the specific version of a software utilised might influence the process outcome, can the specific configuration applied alter the behaviour of an operating system or software component. Capturing this configuration might not always be easy, but in systems that rely on packages for their software, these packages tend to provide information about default locations for configuration files, which might be a start for capturing tools.

Another important aspect of the context model deals with several types of data consumed and created by a pro-

[3] http://www.mancoosi.org/cudf/

cess, as seen in a section of Figure 2(b). We distinguish between data that originates from hardware or software, and whether this data is input to or output of the process, or created and consumed inside the process, i.e. output from one process step and input for another. Capturing this data is an important aspect in verifying that a re-execution of a process yields the same results as the original process, as we detailed in Section 2. It may be easily captured if the process is formally defined in a workflow engine, and this engine provides provenance data, as it is the case with Taverna. In other cases, it may be more difficult to obtain, e.g. by observing network traffic or system library calls.

Other aspects of the model cover for example human resources (including e.g. required qualifications for a certain role), actors, or legal aspects such as data protection laws. Location and time-based aspects need to be captured for processes where synchronisation between activities is important. Further important aspects are documentation and specifications, on all different levels, from high-level design documents of the process, use-case specifications, down to test documents, etc.

While the model is very extensive, it should be noted that a number of aspects can be filled automatically – especially if institutions have well-defined and documented processes. Also, not all sections of the model are equally important for each type of process. Therefore, not every aspect has to be described in most detail.

3.1 Context of the MIR process

We modelled the scientific experiment in the above presented context model. Figure 3 gives an overview on the concrete instances and their relations identified as relevant aspects of the process context.

As this experiment, as most experiments in the MIR domain, is a process mostly focusing on data processing, the majority of the identified aspects are in the technical domain – software components, external systems such as the web service to extract the numerical audio features from, or data exchanged and their format and specification. However, also goals and motivations are important aspects, as they might heavily influence the process. As such, the motivation for the providers of the external systems is relevant, as it might determine the future availability of these services. Commercial systems might be more likely to sustain than services operated by a single person for free.

Another important aspect in this process are licences – depending on which licence terms the components of our process are released under, different options of preservation actions might be available or not. For closed-source, proprietary software, migration to a new execution platform might be prohibited.

A central aspect in the scientific process is the AudioFeatureExtractionService, i.e. the remote web-service that provides the numeric representation for audio files. The service needs as input files encoded in the MP3 format (specified by the ISO standard 11172-3). More specifically, as they are binary files, they need to be further encoded with Base64, to allow for data exchange over the HTTP protocol. The web-service further accepts a number of parameters that control the exact information captured in the numeric representation; they are specified in the AudioFeatureExtractionSpecification, which for example also covers a detailed information on how the extraction works. The service requires an authorisation key. The operator of the web-service provides the service for free, but grants authorisation keys that are non-transferable between different researchers. Finally, the feature extraction service provides the numeric description as ASCII file, following the SOMLib format specification.

As a software component used locally, the WEKA machine learning toolkit requires a Java Virtual Machine (JVM) platform to execute. The JVM in turn is available for many operating systems, but has been specifically tested on a Linux distribution, Ubuntu "Oneiric" 11.04. WEKA requires as input a feature vector in the ARFF Format, and a set of parameters controlling the learning algorithm. These parameters are specified in the WEKA documentation. As output result, the numeric performance metric "accuracy" is provided, as well as a textual, detailed description of the result. WEKA is distributed under the terms of the open-source GNU Public License (GPL) 2.0, which allows for source code modifications.

After this experimentation process, a subsequent process of result analysis and distillation is normally performed, taking input from the experiment outcomes, and finally leading to a publication of the research in the form of e.g. a conference or journal paper. This, again, may be modelled either as a single information object (the paper) connected to the process, and thus to all data and processing steps that led to the results published, or as a more complex process in its own, specifically if a paper reports on meta-studies across several experiment runs.

3.2 Preservation Actions and Evaluation

Preservation Actions are executed to regain or improve access to digital information. For process preservation, preservation actions could be cross compilation of software modules, to enable to run the process on a different platform, or code migration if the former is not (easily) possible. Also the emulation of hardware or software utilised in the process might be a viable option. Further preservation actions include the (file format) migration of specifications and documents. For external services such as web-service digital preservation approaches still need to be developed. For once, web-services should allow the user to query for a version, to identify whether something has changed. To ensure process continuity if a service has indeed changed or disappeared, re-implementing the service is only a viable option if the specification is known. In other cases, capturing provenance data as described in Section 2.1 allows to create mock-up service that can replay previously recorded process executions.

Evaluation of the process is enabled by comparing the provenance data recorded during the original execution (cf. Section 2.1) with the one recorded from a modified process.

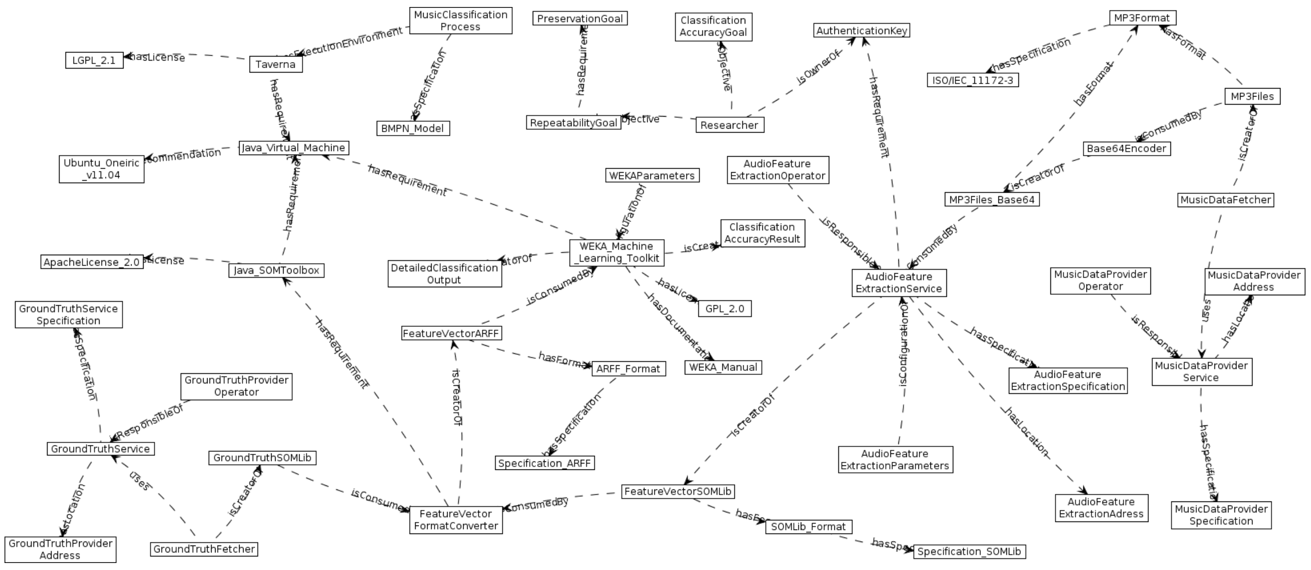


Figure 3: Context Model of musical genre classification process

4. CONCLUSIONS

There is an urgent need to move towards more sustainable process in the Music Information Retrieval domain. Principles of experimental science and traditions are well-established in other disciplines (biology, chemistry, crystallography). This is very complex to achieve in MIR, where legal issues associated with the data analysed are a significant obstacle, but more specifically, fast-changing technology has a huge impact. In this paper, we have presented approaches from the Digital Preservation domain for preserving processes, so that a later execution is enabled. We discussed on the example of a typical musical genre classification process how this can be applied to MIR tasks. Future work will focus on an integration of digital preservation methods into benchmark environments such as the ones proposed by [1] and [8], and evaluation campaigns such as MIREX, forming research infrastructures for MIR research.

5. ACKNOWLEDGMENTS

Part of this work was supported by the project TIMBUS, partially funded by the EU under the FP7 contract 269940.

6. REFERENCES

[1] Timothy G. Armstrong, Alistair Moffat, William Webber, and Justin Zobel. Improvements that don't add up: ad-hoc retrieval results since 1998. In *Proceedings of the ACM Conference on Information and Knowledge Management*, CIKM '09, New York, USA, 2009.

[2] J. Stephen Downie, Joe Futrelle, and David K. Tcheng. The international music information retrieval systems evaluation laboratory: Governance, access and security. In *Proceedings of the International Conference on Music Information Retrieval*, Barcelona, Spain, 2004.

[3] Juliana Freire, David Koop, Emanuele Santos, and Cláudio T. Silva. Provenance for computational tasks: A survey. *Computing in Science and Engineering*, 10(3):11–21, May 2008.

[4] Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. Scientific workflow management and the Kepler system. *Concurrency and Computation: Practice and Experience*, 18(10):1039–1065, 2006.

[5] Yannis Marketakis and Yannis Tzitzikas. Dependency management for digital preservation using semantic web technologies. *International Journal on Digital Libraries*, 10:159–177, 2009.

[6] Paolo Missier, Stian Soiland-Reyes, Stuart Owen, Wei Tan, Alexandra Nenadic, Ian Dunlop, Alan Williams, Tom Oinn, and Carole Goble. Taverna, reloaded. In *Proceedings of the 22nd international conference on Scientific and Statistical Database Management*, Berlin, Heidelberg, June 2010. Springer-Verlag.

[7] Luc Moreau, Juliana Freire, Joe Futrelle, Robert E. Mcgrath, Jim Myers, and Patrick Paulson. Provenance and annotation of data and processes. chapter The Open Provenance Model: An Overview, pages 323–326. Springer-Verlag, Berlin, Heidelberg, 2008.

[8] Julián Urbano. Information retrieval meta-evaluation: Challenges and opportunities in the music domain. In *Proceedings of the International Society for Music Information Retrieval Conference*, Miami, USA, 2011.

[9] Kris West, Amit Kumar, Andrew Shirk, Guojun Zhu, J. Stephen Downie, Andreas F. Ehmann, and Mert Bay. The Networked Environment for Music Analysis (NEMA). In *6th World Congress on Services*, pages 314–317, Miami, USA, July 5-10 2010.

HYPERGRAPH MODELS OF PLAYLIST DIALECTS

Brian McFee
Computer Science and Engineering
University of California, San Diego

Gert Lanckriet
Electrical and Computer Engineering
University of California, San Diego

ABSTRACT

Playlist generation is an important task in music information retrieval. While previous work has treated a playlist collection as an undifferentiated whole, we propose to build playlist models which are tuned to specific categories or *dialects* of playlists. Toward this end, we develop a general class of flexible and scalable playlist models based upon hypergraph random walks. To evaluate the proposed models, we present a large corpus of categorically annotated, user-generated playlists. Experimental results indicate that category-specific models can provide substantial improvements in accuracy over global playlist models.

1. INTRODUCTION

Playlist generation, the automated construction of sequences of songs, is a central component to online music delivery services. Because users tend to consume music sequentially in listening sessions, the quality of a playlist generation algorithm can significantly impact user satisfaction.

Recently, it has been proposed that playlist generation algorithms may be best viewed as probabilistic models of song sequences [11]. This viewpoint, borrowed from the statistical natural language processing literature, enables the automatic evaluation and optimization of a model by computing the likelihood of it generating examples of user-generated playlists. For this method to work, the practitioner must provide a large collection of example playlists, both for model evaluation and parameter optimization.

Of course, numerous subtleties and difficulties arise when working with user-generated playlist data. For example, the data is often noisy, and the author's intent may be obscure. In extreme cases, users may compose playlists by randomly selecting songs from their libraries. More generally, different playlists may have different intended uses (*e.g.*, *road trip* or *party mix*), thematic elements (*break up* or *romantic*), or simply contain songs only of specific genres. While previous work treats the universe of user-generated playlists as a single *language*, building effective global models has proven to be difficult [11].

To better understand the structure of playlists, we advocate a more subtle approach. Rather than viewing naturally occurring playlists as a single language, we propose to model playlists as a collection of *dialects*, each of which may exhibit its own particular structure. Toward this end, we develop dialect-specific playlist models, and evaluate on a large corpus of annotated, user-generated playlists.

The proposed approach raises several natural questions:

- Is it beneficial to individually model playlist dialects?
- Are some dialects easier to model than others?
- Which features are important for each dialect?

Answering these questions will hopefully provide valuable insight into the underlying mechanics of playlist generation.

1.1 Our contributions

In this work, our contributions are two-fold. First, we develop a flexible, scalable, and efficient class of generative playlist models based upon hypergraph random walks. Second, we present a new, large-scale, categorically annotated corpus of user-generated playlist data.

2. HYPERGRAPH RANDOM WALKS

Over the last decade, several researchers have proposed playlist generation algorithms based upon random walks [9, 11,12]. [1] Random walk playlist models consist of a weighted graph $G = (\mathcal{X}, E, w)$, where the vertices $\mathcal{X}$ represent the library of songs, and the edges E and weights w encode pairwise affinities between songs. A playlist is then generated by following a random trajectory through the graph, where transitions $x_t \rightsquigarrow x_{t+1}$ are sampled according to the weights on edges incident to x_t.

Random walk models, while simple and efficient, carry certain practical limitations. It is often unclear how to define the weights, especially when multiple sources of pairwise affinity are available. Moreover, relying on pairwise interactions can severely limit the expressive power of these models (if each song has few neighbors), or scalability and precision (if each song has many neighbors).

To overcome these limitations, we propose a new class of playlist algorithms which allow for more flexible affinities between songs and sets of songs.

2.1 The user model

To motivate our playlist generation algorithm, we propose a simple model of user behavior. Rather than selecting songs

[1] There are many approaches beyond random walk models; see [5, chapter 2] for a survey of recent work.

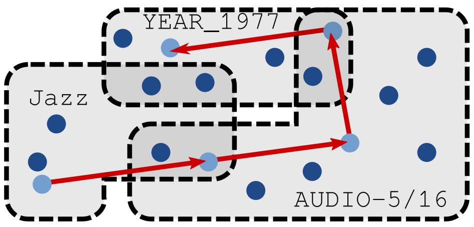


Figure 1. An example random walk on a song hypergraph: vertices represent songs, and edges are subsets of songs. Each transition $x_t \rightsquigarrow x_{t+1}$ must lie within an edge.

directly from the entire collection $\mathcal{X}$, we assume that the user first narrows her selection to a subset $e \subseteq \mathcal{X}$ (*e.g.*, *jazz* songs), from which a song $x_0 \in e$ is chosen uniformly at random. For each subsequent transition $x_t \rightsquigarrow x_{t+1}$, the user selects a subset containing the current song x_t, and then selects x_{t+1} uniformly from that subset.

This user model is exactly characterized by a random walk on a *hypergraph*. Hypergraphs generalize undirected graphs by allowing an edge $e \in E$ to be an arbitrary subset of the vertices, rather than a pair (Figure 1). For example, a hypergraph edge may be as general as *jazz songs*, or as specific as *funk songs from 1977*. Edge weights can be used to encode the importance of a subset: for example, a model of *jazz* playlists would assign high weight to an edges containing *jazz* songs.

This model has several practically beneficial properties. First, it is efficient and scalable, in that the only information necessary to describe a song is its membership in the edge sets. Similarly, it naturally supports extension to new songs without having to significantly alter the model parameters (edge weights). Second, the model can easily integrate disparate feature sources, such as audio descriptors, lyrics, tags, *etc*, as long as they can be encoded as subsets. Moreover, the model degrades gracefully if a song only has partial representation (*e.g.*, audio but no lyrics or tags). Finally, the model is *transparent*, in that each transition can be explained to the user simply in terms of the underlying edge taken between songs. As we will see in Section 3, these edges often have natural semantic descriptions.

2.2 The playlist model

To formalize our model, let $H = (\mathcal{X}, E, w)$ denote a hypergraph over vertices (songs) $\mathcal{X}$, edges $E \subseteq 2^{\mathcal{X}}$, and non-negative weights $w \in \mathbb{R}_+^{|E|}$. We assume that the song library $\mathcal{X}$ and edge set E are given, and our goal is to optimize the edge weights w. We denote by $x_t^e := \mathbb{1}[x_t \in e]$ the indicator that the song x_t is contained in the edge e.

Because the selection of the next song x_{t+1} depends only on the previous song x_t and edge weights w, the model is a first-order Markov process. The likelihood of a playlist $s = (x_0 \rightsquigarrow x_1 \rightsquigarrow \cdots \rightsquigarrow x_T)$ thus factors into likelihood of the initial song, and each subsequent transition:

$$\mathbf{P}(x_0 \rightsquigarrow x_1 \rightsquigarrow \cdots \rightsquigarrow x_T \mid w) = \mathbf{P}(x_0 \mid w) \prod_{t=0}^{T-1} \mathbf{P}(x_{t+1} \mid x_t, w).$$

Given the edge weights w, the distribution over the initial song x_0 can be characterized by marginalizing over edges:

$$\mathbf{P}(x_0 \mid w) := \sum_{e \in E} \mathbf{P}(x_0 \mid e) \mathbf{P}(e \mid w) = \sum_{e \in E} \frac{x_t^e}{|e|} \frac{w_e}{\sum_{f \in E} w_f}.$$

Similarly, the probability of a transition $x_t \rightsquigarrow x_{t+1}$ is defined by marginalizing over edges incident to x_t:

$$\begin{aligned}\mathbf{P}(x_{t+1} \mid x_t, w) &:= \sum_{e \in E} \mathbf{P}(x_{t+1} \mid e, x_t) \mathbf{P}(e \mid x_t, w) \\ &= \sum_{e \in E} \frac{\mathbb{1}[x_{t+1} \neq x_t] \cdot x_{t+1}^e}{|e| - 1} \cdot \frac{x_t^e w_e}{\sum_{f \in E} x_t^f w_f}.\end{aligned}$$

Finally, to promote sparsity among the edge weights and resolve scale-invariance in the model, we assume an IID exponential prior on edge weights w_e with rate $\lambda > 0$:

$$\mathbf{P}(w_e) := \lambda \cdot \exp(-\lambda w_e) \cdot \mathbb{1}[w_e \in \mathbb{R}_+].$$

2.3 Learning the weights

Given a training sample of playlists $\mathcal{S} \subset \mathcal{X}^*$,[2] we would like to find the maximum a posteriori (MAP) estimate of w:

$$\begin{aligned} w &\leftarrow \operatorname*{argmax}_{w \in \mathbb{R}_+^{|E|}} \log \mathbf{P}(w \mid \mathcal{S}) \\ &= \operatorname*{argmax}_{w \in \mathbb{R}_+^{|E|}} \sum_{s \in \mathcal{S}} \log \mathbf{P}(s \mid w) + \sum_{e \in E} \log \mathbf{P}(w_e). \qquad (1)\end{aligned}$$

The MAP objective (1) is not concave, and it is generally difficult to find a global optimum. Our implementation uses the L-BFGS-B algorithm [2] to solve for w, and converges quite rapidly to a stationary point. Training typically takes a matter of seconds, even for the large playlist collections and edge sets described in Section 3.

3. DATA COLLECTION

Previous work on playlist modeling used the Art of the Mix[3] (AotM) collection of Ellis, et al. [4]. The existing AotM dataset was collected in 2002, and consists of roughly 29K playlists over 218K songs, provided as lists of plain-text song and artist names. In this work, we expand and enrich this dataset into a new collection, which we denote as AotM-2011.[4] This section describes our data collection, pre-processing, and feature extraction methodology.

3.1 Playlists: Art of the Mix 2011

To expand the AotM playlist collection, we crawled the site for all playlists, starting from the first indexed playlist (1998-01-22) up to the most recent at the time of collection (2011-06-17), resulting in 101343 unique playlists. Each playlist contains not only track and artist names, but a timestamp and categorical label (*e.g.*, *Road Trip* or *Reggae*).

[2] $\mathcal{X}^*$ denotes the Kleene star operation.
[3] `http://www.artofthemix.org`
[4] `http://cosmal.ucsd.edu/cal/projects/aotm2011/`.

To effectively model the playlist data, the plain-text song and artist names must be resolved into a common namespace. Following previous work, we use the Million Song Dataset (MSD) as the underlying database [1, 11]. Rather than rely on the Echo Nest text-search API to resolve song identifiers, we instead implemented a full-text index of MSD song and artist names in Python with the Whoosh [5] library. This allowed both high throughput and fine-grained control over accent-folding and spelling correction. Each (*artist, song*) pair in the raw playlist data was used as a query to the index, and resolved to the corresponding MSD song identifier (if one was found). In total, 98359 songs were matched to unique identifiers.

Because not every song in a raw playlist could be correctly resolved, each playlist was broken into contiguous segments of two or more matched song identifiers. Finally, playlist segments were grouped according to category. Table 1 lists each of the 25 most popular categories by size.

3.2 Edge features

To fully specify the playlist model, we must define the edges of the hypergraph. Because edges can be arbitrary subsets of songs, the model is able to seamlessly integrate disparate feature modalities. We use the following collection of edge features, which can be derived from MSD and its add-ons.

Audio To encode low-level acoustic similarity, we first mapped each song i to a vector $x_i \in \mathbb{R}^{222}$ using the optimized vector quantized Echo Nest Timbre (ENT) descriptors provided by [1, 11]. Audio descriptors were clustered via online k-means, and cluster assignments were used to produce k disjoint subsets. Repeating this for $k \in \{16, 64, 256\}$ provided multiple overlapping edges of varying degrees of granularity. All 98K songs receive audio representations.

Collaborative filter To capture high-level similarities due to user listening patterns, we construct edges from the taste profile data used in the MSD Challenge [10]. We used the Bayesian Personalized Ranking (BPR) algorithm [6, 13] to factor the users-by-songs (1M-by-380K) feedback matrix into latent feature vectors $x_i \in \mathbb{R}^{32}$. The BPR regularization parameters were set to $\lambda_1 = \lambda_2 = 10^{-4}$. Edges were constructed by cluster assignments following the procedure described above for audio features. 62272 songs (63%) coincide with the taste profile data.

Era The era in which songs are released can play an important role in playlist composition [3, 8]. To model this, we use the MSD meta-data to represent each song by its year and half-overlapping decades. For example, the song *Parliament - Flash Light* maps to edges `YEAR-1977`, `DECADE-1970` and `DECADE-1975`. 77884 songs (79%) were mapped to era descriptors.

Familiarity Previous studies have noted the importance of song- or artist-familiarity when composing playlists [3, 11]. We used the artist familiarity data provided with MSD, which maps each song to the range $[0, 1]$ (0 being unfamiliar, 1 being very familiar). Edges were constructed by estimating the 25th and 75th percentiles of familiarity, and mapping each song to `LOW`, `MEDIUM`, or `HIGH` familiarity.

Lyrics Previous studies have shown the importance of lyrics in playlist composition [8]. To compute lyrical similarity, we applied online latent Dirichlet allocation (LDA) [7] with $k = 32$ to the musiXmatch lyrics database. [6] We then constructed three sets of 32 edges (one edge per topic): the first matches each song to its most probable topic, the second matches each song to its top three topics, and the third set to its top five topics. 53351 songs (56%) were found in the musiXmatch data.

Social tags Previous work incorporated semantic information by using the total similarity between bag-of-tags vectors of songs to determine similarity [11]. Here, we take a more flexible approach, and model each tag separately. Using the Last.fm [7] tags for MSD, we match each song to its top-10 most frequent tags. Each tag induces an edge (the songs assigned to that tag). [8] 80396 songs (82%) matched to tag edges.

Uniform shuffle Because the features described above cannot model all possible transitions, we include a `uniform` edge that contains all songs. A transition through the uniform edge can be interpreted as a random restart of the playlist. The uniform shuffle also provides a standard baseline for comparison.

Feature conjunctions Some of the features described above may be quite weak individually, but when combined, become highly descriptive. For example, the tag `rock` and era `YEAR-1955` are both vague, but the conjunction of these two descriptors — `rock-&--YEAR-1955` — retains semantic interpretability, and is much more precise. We therefore augment the above collection of edges with all pair-wise intersections of features. Note that this induces general cross-modal feature conjunctions, such as `Lyrics topic #4-&-Audio cluster #17`, resulting in an extremely rich set of song descriptors.

4. EXPERIMENTS

To evaluate the proposed method, we randomly partitioned each of the top-25 categories listed in Table 1 into ten 75/25 train/test splits. For each split, the train (test) sets are collected across categories to form a global train (test) set *ALL*, which is used to train a global model. After fitting a

[5] `http://packages.python.org/Whoosh/`

[6] `http://labrosa.ee.columbia.edu/millionsong/musixmatch`

[7] `http://last.fm/`

[8] A similar tag-hypergraph model was proposed by Wang, et al. [14].

Category	Playlists	Segments	Songs
Mixed	41798	101163	64766
Theme	12813	31609	35862
Rock-Pop	4935	13661	20364
Alternating DJ	4334	10493	18083
Indie	4528	10333	13678
Single Artist	3717	9044	17715
Romantic	2523	6269	8873
Road Trip	1846	4817	8935
Punk	1167	3139	4936
Depression	1128	2625	4794
Break Up	1031	2512	4692
Narrative	964	2328	5475
Hip Hop	1070	1958	2505

Category	Playlists	Segments	Songs
Sleep	675	1487	2957
Electronic Music	611	1131	2290
Dance-House	526	1117	2375
Rhythm and Blues	432	1109	2255
Country	398	908	1756
Cover	447	833	1384
Hardcore	268	633	1602
Rock	215	565	1866
Jazz	295	512	1089
Folk	241	463	1137
Reggae	183	403	831
Blues	165	373	892
Top-25	86310	209485	97411

Table 1. The distribution of the top 25 playlist categories in AotM-2011. Each playlist consists of one or more segments of at least two contiguous MSD songs. 948 songs do not appear within the top 25 categories, but are included in the model.

Feature	# Edges
Audio	204
Era	56
Lyrics	82
Uniform	1

Feature	# Edges
Collaborative filter	93
Familiarity	3
Tags	201
All features	640
Feature conjunctions	6390

Table 2. Summary of edges after pruning.

model to each training set, we compute the average (length-normalized) log-likelihood of the test set $\mathcal{S}'$:

$$\mathcal{L}(\mathcal{S}' \mid w) := \frac{1}{|\mathcal{S}'|} \sum_{s \in \mathcal{S}'} \frac{1}{|s|} \log \mathbf{P}(s \mid w).$$

For comparison purposes, we report performance in terms of the relative gain over the uniform shuffle model w_{u} (all weight assigned to the uniform edge):

$$G(w) := 1 - \frac{\mathcal{L}(\mathcal{S}' \mid w)}{\mathcal{L}(\mathcal{S}' \mid w_{\text{u}})}.$$

To simplify the model and reduce over-fitting effects, we pruned all edges containing fewer than 384 (98359/256) songs. Similarly, we pruned redundant conjunction edges that overlapped by more than 50% with either of their constituent edges. Table 2 lists the number of edges retained after pruning. On average, each song maps to 76.46 ± 57.79 edges, with a maximum of 218. In all experiments, we fix the prior parameter $\lambda = 1$.

4.1 Experiment 1: Does dialect matter?

In the first set of experiments, we compare the global model to category-specific models. Figure 2 illustrates the relative gain over uniform across all categories for four different model configurations: tags, tags with pairwise conjunctions, all features, and all features with conjunctions.

Several interesting trends can be observed from Figure 2. First, in all but two cases — *Narrative* and *Rock* under the all features with conjunctions model — category-specific models perform at least as well as the global model, and are often substantially better. As should be expected, the effect is most pronounced for genre-specific categories that naturally align with semantic tags (*e.g.*, *Hip Hop* or *Punk*).

Note that the larger categories overlap more with *ALL*, leaving less room for improvement over the global model. Not surprisingly, the *Mixed* category appears to be difficult to model with similarity-based features. The fact that it is the single largest category (Table 1) may explain some of the difficulties observed in previous studies when using a global model [11]. Similarly, several other categories are quite broad (*Theme*, *Narrative*, *Rock*), or may be inherently difficult (*Alternating DJ*, *Mixed*).

Also of note are differences across model configurations. Feature conjunctions generally provide a modest improvement, both in the global and category-specific models. Due to the large parameter space, some over-fitting effects can be observed in the smallest categories (*Folk*, *Reggae*, *Blues*). Interestingly, several categories benefit substantially from the inclusion of all features compared to only tags (*e.g.*, *Hip Hop*, *Punk*, *Jazz*).

4.2 Experiment 2: Do transitions matter?

Given the flexibility of the model, it is natural to question the importance of modeling playlist continuity: could a model which ignores transition effects perform as well as the random walk model? To test this, we split each playlist $s = (x_0 \rightsquigarrow x_1 \rightsquigarrow \cdots \rightsquigarrow x_T)$ into singletons $s_0 = (x_0), \cdots, s_T = (x_T)$. With this modified corpus, the model treats each song in a playlist as an independent draw from the initial song distribution $\mathbf{P}(x_0 \mid w)$. Consequently, a model trained on this corpus can fit global trends across playlists within a category, but cannot enforce local continuity.

Figure 3 illustrates the relative gain for each category under the stationary distribution with all features and conjunctions. The results are qualitatively similar for alternate model configurations. Compared to Figure 2 (bottom-right), the results are substantially worse for most categories. In many cases, the stationary model performs worse than the uniform shuffle. This reflects the importance of transition effects when modeling playlists, even when the corpus is confined to genre-specific categories.

4.3 Experiment 3: Which features matter?

As illustrated in Figure 2, certain categories seem to benefit substantially from the inclusion of non-tag features. To investigate this effect, Figure 4 illustrates the aggregated weight for each feature type under each of the category models. Note that weight is aggregated across feature con-

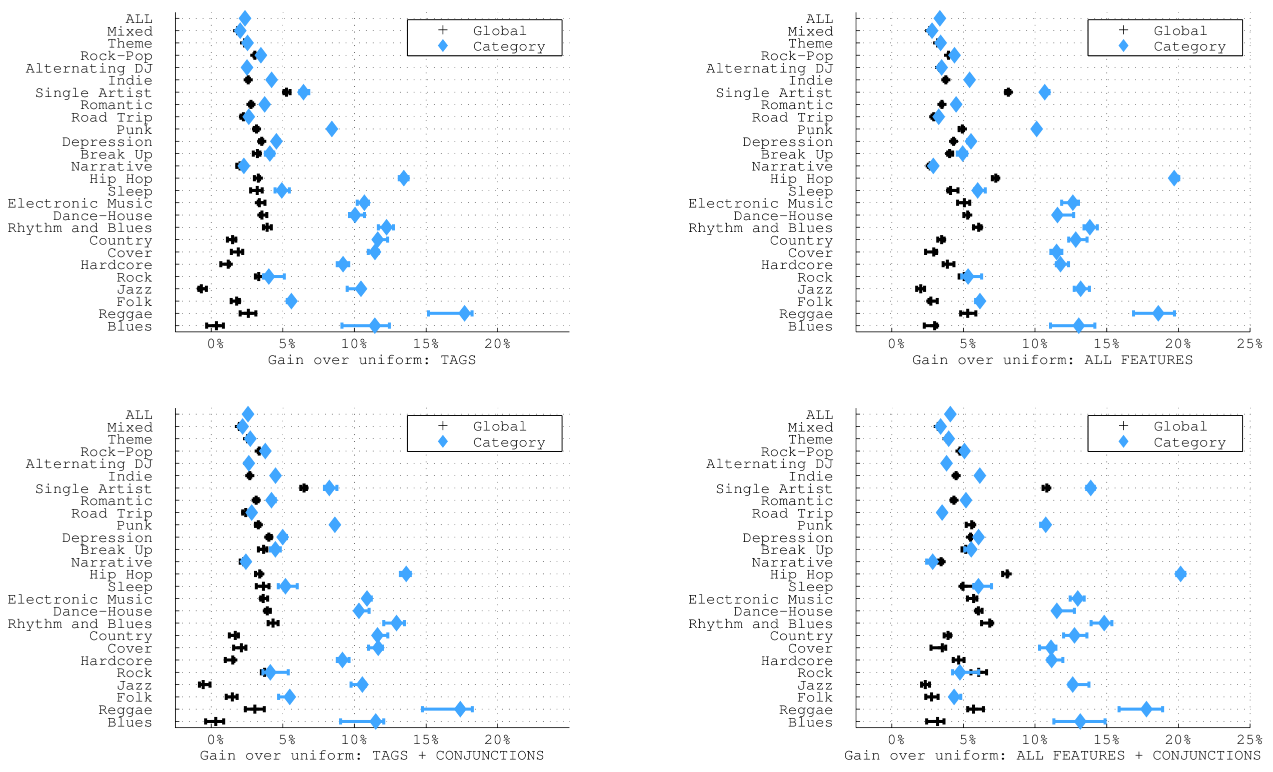


Figure 2. The median gain in log-likelihood over the uniform shuffle model, aggregated over ten random splits of the data. Error bars span the 0.25–0.75 quantiles. Category-specific models generally outperform global models.

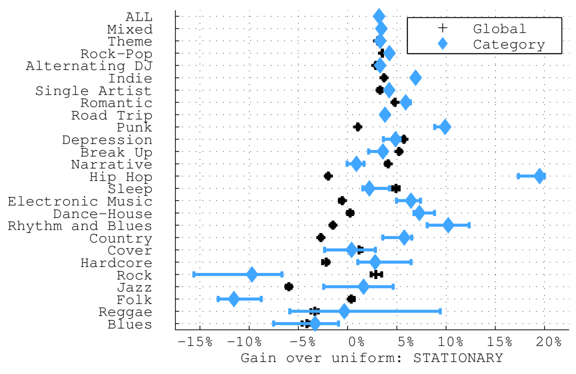


Figure 3. Log-likelihood gain over uniform with the stationary model (all features and conjunctions). Ignoring temporal structure significantly degrades performance.

junctions, so the weight for edge DECADE_1955-&-Rock counts both for *Era* and *Tag*.

Tags receive the most weight (64% on average) across all categories. Audio features appear to be most useful in *Hip Hop*, *Jazz* and *Blues* (43%–44%, compared to 26% average). This is not surprising, given that these styles feature relatively distinctive instrumentation and production qualities. Lyrical features receive the most weight in categories with salient lyrical content (*Folk*, *Cover*, *Narrative*, *Hardcore*, *Break Up*) and low weight in categories with little or highly variable lyrical content (*Electronic Music*, *Dance-House*, *Jazz*). Era and familiarity receive moderate weight (on average, 22% and 15% respectively), but the majority (20% and 14%) is due to conjunctions.

4.4 Example playlists

Table 3 illustrates samples drawn from category-specific feature conjunction models. For generative purposes, the uniform edge was removed after training. The generated playlists demonstrate both consistency within a single playlist and variety across playlists. Each transition in the playlist is explained by the corresponding (incoming) edge, which provides transparency to the user: for example, *Cole Porter - You're the Top* follows *Django Rheinhardt - Brazil* because both songs belong to the conjunction edge AUDIO-3/16--&-jazz, and share both high- and low-level similarity.

5. CONCLUSION

We have demonstrated that playlist model performance can be improved by treating specific categories of playlists individually. While the simple models proposed here work well in some situations, they are far from complete, and suggest many directions for future work. The first-order Markov assumption is clearly a simplification, given that users often create playlists with long-term interactions and global thematic properties. Similarly, the uniform distribution over songs within an edge set allows for an efficient and scalable implementation, but allowing non-uniform distributions could also be an avenue for future improvement.

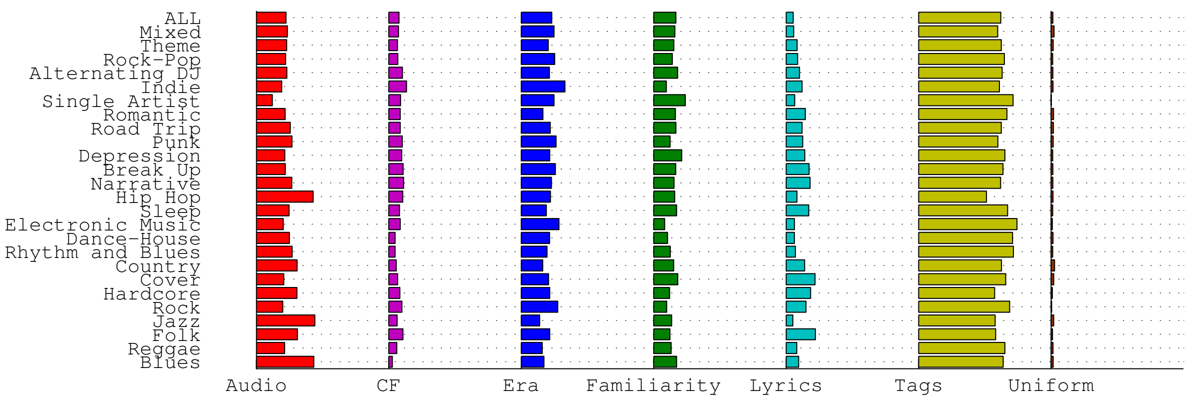


Figure 4. Distribution of learned edge weights for each playlist category. Weight is aggregated across feature conjunctions.

Category	Edge	Playlist
Hip Hop	`AUDIO-149/256`	*Eminem - The Conspiracy (Freestyle)*
	`AUDIO-149/256`	*Busta Rhymes - Bounce*
	`DECADE-2000-&-rap`	*Lil' Kim (Featuring Sisqo) - How Many Licks?*
	`old school`	*A Tribe Called Quest - Butter*
	`DECADE_1985-&-Hip-Hop`	*Beastie Boys - Get It Together*
	`AUDIO-12/16`	*Big Daddy Kane - Raw [Edit]*
Electronic Music	`AUDIO-11/16-&-downtempo`	*Everything But The Girl - Blame*
	`DECADE_1990-&-trip-hop`	*Massive Attack - Spying Glass*
	`AUDIO-11/16-&-electronica`	*Björk - Hunter*
	`DECADE_2000-&-AUDIO-23/64`	*Four Tet - First Thing*
	`electronica-&-experimental`	*Squarepusher - Port Rhombus*
	`electronica-&-experimental`	*The Chemical Brothers - Left Right*
Rhythm and Blues	`70s-&-soul`	*Lyn Collins - Think*
	`AUDIO-14/16-&-funk`	*Isaac Hayes - No Name Bar*
	`DECADE_1965-&-soul`	*Michael Jackson - My Girl*
	`AUDIO-6/16-&-soul`	*The Platters - Red Sails In The Sunset*
	`FAMILIARITY_MED-&-60s`	*The Impressions - People Get Ready*
	`soul-&-oldies`	*James & Bobby Purify - I'm Your Puppet*
Jazz	`AUDIO-14/16-&-jazz`	*Peter Cincotti - St Louis Blues*
	`jazz`	*Tony Bennett - The Very Thought Of You*
	`vocal jazz`	*Louis Prima - Pennies From Heaven*
	`jazz-&-instrumental`	*Django Reinhardt - Brazil*
	`AUDIO-3/16-&-jazz`	*Cole Porter - You're The Top*
	`jazz`	*Doris Day - My Blue Heaven*

Table 3. Example playlists generated by various dialect models. *Edge* denotes the incoming edge to the corresponding song, which for transitions, is shared by the previous song. Feature conjunctions are indicated by `X-&-Y`.

6. ACKNOWLEDGMENTS

The authors acknowledge support from Qualcomm, Inc., eHarmony, Inc., Yahoo! Inc., Google, Inc., and NSF Grants CCF-0830535, IIS-1054960, and EIA-0303622.

7. REFERENCES

[1] T. Bertin-Mahieux, D.P.W. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *ISMIR*, 2011.

[2] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16(5):1190–1208, Sep. 1995.

[3] S.J. Cunningham, D. Bainbridge, and A. Falconer. “More of an art than a science”: Supporting the creation of playlists and mixes. In *ISMIR*, 2006.

[4] D. Ellis, B. Whitman, A. Berenzweig, and S. Lawrence. The quest for ground truth in musical artist similarity. In *ISMIR*, 2002.

[5] B. Fields. *Contextualize Your Listening: The Playlist as Recommendation Engine*. PhD thesis, Goldsmiths, University of London, April 2011.

[6] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: A free recommender system library. In *RecSys*, 2011.

[7] M. Hoffman, D. Blei, and F. Bach. Online learning for latent dirichlet allocation. In *NIPS*. 2010.

[8] J.H. Lee. How similar is too similar?: Exploring users' perceptions of similarity in playlist evaluation. In *ISMIR*, 2011.

[9] B. Logan. Content-based playlist generation: exploratory experiments. In *ISMIR*, 2002.

[10] B. McFee, T. Bertin-Mahieux, D.P.W. Ellis, and G.R.G. Lanckriet. The million song dataset challenge. In *AdMIRe*, 2012.

[11] B. McFee and G. R. G. Lanckriet. The natural language of playlists. In *ISMIR*, 2011.

[12] R. Ragno, C. J. C. Burges, and C. Herley. Inferring similarity between music objects with application to playlist generation. In *7th ACM SIGMM international workshop on Multimedia information retrieval*, 2005.

[13] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*, 2009.

[14] F. Wang, X. Wang, B. Shao, T. Li, and M. Ogihara. Tag integrated multi-label music style classification with hypergraph. In *ISMIR*, 2009.

LEARNING TO EMBED SONGS AND TAGS FOR PLAYLIST PREDICTION

Joshua L. Moore, Shuo Chen, Thorsten Joachims
Cornell University, Dept. of Computer Science
{jlmo|shuochen|tj}@cs.cornell.edu

Douglas Turnbull
Ithaca College, Dept. of Computer Science
dturnbull@ithaca.edu

ABSTRACT

Automatically generated playlists have become an important medium for accessing and exploring large collections of music. In this paper, we present a probabilistic model for generating coherent playlists by embedding songs and social tags in a unified metric space. We show how the embedding can be learned from example playlists, providing the metric space with a probabilistic meaning for song/song, song/tag, and tag/tag distances. This enables at least three types of inference. First, our models can generate new playlists, outperforming conventional n-gram models in terms of predictive likelihood by orders of magnitude. Second, the learned tag embeddings provide a generalizing representation for embedding new songs, allowing it to create playlists even for songs it has never observed in training. Third, we show that the embedding space provides an effective metric for matching songs to natural-language queries, even if tags for a large fraction of the songs are missing.

1. INTRODUCTION

Music consumers can store thousands of songs on their computer or smart phone. In addition, cloud-based services like Rhapsody or Spotify give instant and on-demand access to millions of songs. While these technologies provide powerful new ways to access music, they can also overwhelm users by giving them too much choice [15].

This has created substantial interest in automatic playlist algorithms that can help consumers explore large collections of music. Companies like Apple and Pandora have developed successful commercial playlist algorithms, but relatively little is known about how these algorithms work and how well they perform in rigorous evaluations. Comparably little scholarly work has been done on automated methods for playlist generation (e.g., [1, 4, 10, 12, 14]), and the results to date indicate that it is far from trivial to operationally define what makes a playlist coherent.

Most approaches to automatic playlist creation rely on computing some notion of music similarity between pairs of songs. Numerous similarity functions have been proposed and are often based on the analysis of audio content [9, 13], social tag information [8], web document mining [6], preference-based ratings data [11], or some combination of these data sources. Given a music similarity algorithm, a playlist is created by finding the most similar songs to a given seed song or set of seed songs.

In this paper, we explore the idea of learning a playlist model that does not require an external similarity measure and that is trained directly on the data of interest, namely historical playlists. In particular, we extend the *Logistic Markov Embedding* (LME) [3] approach to probabilistic sequence modeling to incorporate social tags, unifying song and tag embeddings in a single Euclidean space. This provides a probabilistically well-founded and constructive way to compute meaningful distances between pairs of songs, pairs of tags, and songs and tags. We show that this joint embedding is useful not only for probabilistically sound playlist generation, but also for a variety of other music information retrieval tasks such as corpus visualization, automatic tagging, and keyword-based music retrieval.

An efficient C implementation, a demo, and data are available at `http://lme.joachims.org`.

2. RELATED WORK

Automatically generated playlists are a key component in several commercial systems. For example, Pandora relies on content-based music analysis by human experts [16] while Apple iTunes Genius relies on preference ratings and collaborative filtering [2]. What is not known is the mechanism by which the playlist algorithms are used to *order* the set of relevant songs, nor is it known how well these playlist algorithms perform in rigorous evaluations.

In the scholarly literature, two recent papers address the topic of playlist prediction. First, Maillet et al. [10] formulate the playlist ordering problem as a supervised binary classification problem that is trained discriminatively. Positive examples are pairs of songs that appeared in this order in the training playlists, and negative examples are pairs of songs selected at random which do not appear together in order in historical data. Second, McFee and Lanckriet [12] take a generative approach by modeling historical playlists as a Markov chain. That is, the probability of the next song in a playlist is determined only by acoustic and/or social-tag similarly to the current song. Our approach is substantially different from both [10] and [12], since we do not require any acoustic or semantic information about the songs.

While relatively little work has been done on explicitly modeling playlists, considerably more research has focused on embedding songs (or artists) into a similarity-based music space (e.g., [4, 9, 14, 18].) For example, Platt et al. use semantic tags to learn a Gaussian process kernel function between pairs of songs [14]. More recently, Weston et al. learn an embedding over a joint semantic space of audio features, tags and artists by optimizing performance metrics for various music retrieval tasks [18]. Our approach, however, differs substantially from these existing methods, since it explicitly models the sequential nature of playlists in the embedding. Recently and independently, [1] also proposed a sequential embedding model. However, their model does not include tags.

Modeling playlists as a Markov chain connects to a large body of work on sequence modeling in natural language processing (NLP) and speech recognition. Smoothed n-gram models (see e.g. [5]) are the most commonly used method in language modeling, and we will compare against such models in our experiments.

3. PROBABILISTIC EMBEDDING OF PLAYLISTS

Our goal is to estimate a generative model of coherent playlists, which will enable us to efficiently sample new playlists. More formally, given a collection $\mathcal{S} = \{s_1, ..., s_{|\mathcal{S}|}\}$ of songs s_i, we would like to estimate the distribution $\Pr(p)$ of coherent playlists $p = (p^{[1]}, ..., p^{[k_p]})$. Each element $p^{[i]}$ of a playlist refers to one song from S.

A natural approach is to model playlists as a Markov chain, where the probability of a playlist $p = (p^{[1]}, ..., p^{[k_p]})$ is decomposed into the product of transition probabilities $\Pr(p^{[i]}|p^{[i-1]})$ between adjacent songs $p^{[i-1]}$ and $p^{[i]}$.

$$\Pr(p) = \prod_{i=1}^{k_p} \Pr(p^{[i]}|p^{[i-1]}) \tag{1}$$

For ease of notation, we assume that $p^{[0]}$ is a dedicated start symbol. Such bi-gram (or, more generally, n-gram) models have been widely used in language modeling for speech recognition and machine translation with great success [5]. In these applications, the $O(|\mathcal{S}|^n)$ transition probabilities $\Pr(p^{[i]}|p^{[i-1]})$ are estimated from a large corpus of text using sophisticated smoothing methods.

While such n-gram approaches can be applied to playlist prediction in principle, there are fundamental differences between playlists and language. First, playlists are less constrained than language, so that transition probabilities between songs are closer to uniform. This means that we need a substantially larger training corpus to observe all of the (relatively) high-probability transitions even once. Second, and in contrast to this, we have orders of magnitude less playlist data to train from than we have written text.

To overcome these problems, we propose a Markov-chain sequence model that produces a *generalizing representation* of songs, song sequences, and social tags. Unlike n-gram models that treat words as atomic units without metric relationships between each other, our approach seeks to model coherent playlists as paths through a latent space. In particular, songs are embedded as points in this space so that Euclidean distance between songs reflects the transition probabilities. Similarly, each social tag is represented as a point in this space, summarizing the average location of songs with that tag. The key learning problem is to determine the location of each song and tag using existing playlists as training data. Once songs and tags are embedded, our model can assign meaningful transition probabilities even to those transitions that were not seen in the training data, and it can also reason about tagged songs that were never seen before.

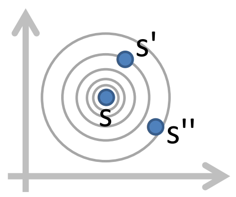
In the following we start by reviewing the basic LME model of $\Pr(p)$, and then extend this model to incorporate social tags.

3.1 Embedding Model for Songs

The basic LME model [3] represents each song s as a single vector $X(s)$ in d-dimensional Euclidean space $\mathcal{M}$. The key assumption of our model is that the transition probabilities $\Pr(p^{[i]}|p^{[i-1]})$ are related to the Euclidean distance $||X(p^{[i]}) - X(p^{[i-1]})||_2$ between $p^{[i-1]}$ and $p^{[i]}$ in $\mathcal{M}$ through the following logistic model:

$$\Pr(p^{[i]}|p^{[i-1]}) = \frac{e^{-||X(p^{[i]})-X(p^{[i-1]})||_2^2}}{\sum_{j=1}^{|S|} e^{-||X(s_j)-X(p^{[i-1]})||_2^2}} \tag{2}$$

This is illustrated in the figure to the right, showing that transitioning from s to a nearby point s' is more likely than transitioning to a point s'' that is further away. We will typically abbreviate the partition function in the denominator as $Z(p^{[i-1]})$, and the distance $||X(s) - X(s')||_2$ between two songs in embedding space as $\Delta(s, s')$ for brevity. Using a Markov model with this transition distribution, we can now define the probability of an entire playlist of a given length k as



$$\Pr(p) = \prod_{i=1}^{k_p} \Pr(p^{[i]}|p^{[i-1]}) = \prod_{i=1}^{k_p} \frac{e^{-\Delta(p^{[i]},p^{[i-1]})^2}}{Z(p^{[i-1]})}. \tag{3}$$

The LME seeks to discover an embedding of the songs into this latent space which causes "good" playlists to have high probability of being generated by this process. This is inspired by collaborative filtering methods such as [7], which similarly embed users and items into a latent space to predict users' ratings of items. However, our approach differs from these methods in that we wish to predict paths through the space, as opposed to independent item ratings.

In order to learn the embedding of songs, we use a sample $D = (p_1, ..., p_n)$ of existing playlists as training data and take a maximum a posteriori (MAP) approach to learning. Denoting with X the matrix of embedding vectors for all songs in the collection $\mathcal{S}$, this leads to the following training problem

$$X = \operatorname*{argmax}_{X\in\Re^{|S|\times d}} \prod_{p\in D} \prod_{i=1}^{k_p} \frac{e^{-\Delta(p^{[i]},p^{[i-1]})^2}}{Z(p^{[i-1]})} \cdot \prod_{i=1}^{|S|} e^{-\lambda||X(s_i)||_2^2}, \tag{4}$$

where we also added a zero-mean Normal prior as regularizer to control overfitting (see term after the dot). The parameter λ controls how heavily the embedding is regularized. While the optimization problem is not concave, we have already shown in [3] how to efficiently and robustly find good optima using a stochastic gradient approach.

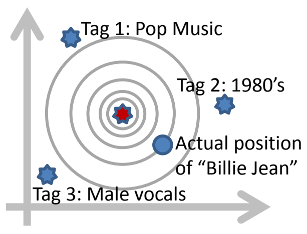
3.2 Embedding Model for Songs and Tags

The previous model is very general in that it does not require any features that describe songs. However, this is also a shortcoming, since it may ignore available information. We therefore now extend the LME to include tags as prior information. The new model will provide reasonable embeddings even for songs it was not trained on, and it will define a unified metric space for music retrieval based on query tags.

The key idea behind the new model is that the tags $T(s)$ of song s inform the prior distribution of its embedding location $X(s)$. In particular, each tag t is associated with a Normal distribution $\mathcal{N}(M(t), \frac{1}{2\lambda} I_d)$ with mean $M(t)$. Here, I_d is the d by d identity matrix and we will see soon that λ again behaves like a regularization parameter. For a song with multiple tags, we model the prior distribution of its embedding as the average of the Normal distribution of its tags $T(s)$, while keeping the variance constant.

$$\Pr(X(s)|T(s)) = \mathcal{N}\left(\frac{1}{|T(s)|}\sum_{t\in T(s)} M(t), \frac{1}{2\lambda} I_d\right) \qquad (5)$$

Note that this definition of $\Pr(X(s)|T(s))$ nicely generalizes the regularizer in (4), which corresponds to an "uninformed" Normal prior $\Pr(X(s)) = \mathcal{N}(0, \frac{1}{2\lambda} I_d)$ centered at the origin of the embedding space. The tag-based prior distribution is illustrated in the figure to the right. In this example, the song "Billie Jean" has the three tags "pop music", "male vocals" and "1980s". Each tag has a mean $M(t)$ as depicted, and $\Pr(X(s)|T(s))$ is centered at the average of the tag means, providing the prior for the embedding of "Billie Jean". Without any training data, the most likely location is the center of the prior, but with more observed training data the embedding may move further away as necessary.



Let M be the matrix of all tag means, we obtain the following maximum a posteriori estimate for the tag-based LME analogous to the basic LME model:

$$(X, M) = \underset{X,M}{\operatorname{argmax}} \Pr(D|X)\cdot\Pr(X|M) \qquad (6)$$

$$= \underset{X,M}{\operatorname{argmax}} \prod_{p\in D}\prod_{i=1}^{k_p} \frac{e^{-\Delta(p^{[i]},p^{[i-1]})^2}}{Z(p^{[i-1]})} \cdot \prod_{i=1}^{|S|} e^{-\lambda||X(s) - \frac{\sum_{t\in T(s)} M(t)}{|T(s)|}||_2^2}$$

Note that we now optimize jointly over the song locations $X(s)$ and tag locations $M(t)$. In this way, the tag-based LME model yields a meaningful probabilistic interpretation of distances not only among songs, but also among songs and tags. The following experiments exploit this for locating new songs and for tag-based music retrieval.

	yes_small	*yes_big*
Appearance Threshold	20	5
Num of Songs	3,168	9,775
Num of Train Trans	134,431	172,510
Num of Test Trans	1,191,279	1,602,079

Table 1: Statistics of the playlists datasets.

4. EXPERIMENTS

The playlists and tag data we used for our experiments are respectively crawled from *Yes.com* and *Last.fm*.

Yes.com is a website that provides radio playlists from hundreds of radio stations in the United States. By using the web based API [1], one can retrieve the playlist record of a specified station for the last 7 days. We collected as many playlists as possible by specifying all possible genres and getting playlists from all possible stations. The collection lasted from December 2010 to May 2011. This lead to a dataset of 75,262 songs and 2,840,553 transitions. To get datasets of various sizes, we pruned the raw data so that only the songs with a number of appearances above a certain threshold are kept. We then divide the pruned set into a training set and a testing set, making sure that each song has appeared at least once in the training set. We report results for two datasets, namely *yes_small* and *yes_big*, whose basic statistics are shown in Table 1.

Last.fm provides tag information for songs, artists and albums that is contributed by its millions of users. For each of the songs in our playlists dataset, we query the Last.fm API [2] for the name of the artist and the song, retrieving the top tags. We then prune the tag set by only keeping the top 250 tags with the most appearances across songs. Note that Last.fm did not provide any tags for about 20% of songs.

Unless noted otherwise, experiments use the following setup. Any model (either the LME or the baseline model) is first trained on the training set and then tested on the test set. We evaluate test performance using average log-likelihood as our metric. It is defined as $\log(\Pr(D_{\text{test}}))/N_{\text{test}}$, where N_{test} is the number of transitions in test set.

4.1 What does the embedding space look like?

Before starting the quantitative evaluation of our method, we first want to give a qualitative impression of the embeddings it produces. Figure 1 shows the two-dimensional embedding of songs and tags according to (6) for the *yes_small* dataset. The top 50 genre tags are labeled, and the lighter points represent songs.

Overall, the embedding matches our intuition of what a semantic music space should look like. The most salient

[1] `http://api.yes.com`
[2] `http://www.last.fm/api`

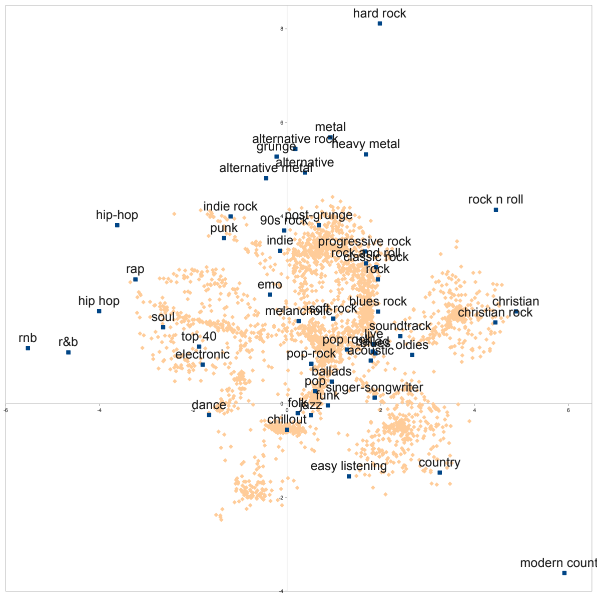


Figure 1: 2D embedding for *yes_small*. The top 50 genre tags are labeled; lighter points represent songs.

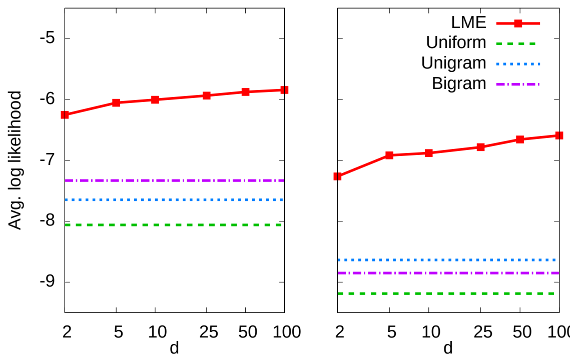


Figure 2: Log-likelihood on the test set for the LME and the baselines on *yes_small* (left) and *yes_big* (right).

observation is that the embedding of songs does not uniformly cover the space, but forms clusters as expected. The location of the tags provides interesting insight into the semantics of these clusters. Note that semantically synonymous tags are typically close in embedding space (e.g. “christian rock” and “christian”, “metal rock” and “heavy metal”). Furthermore, location in embedding space generally interpolates smoothly between related genres (e.g. “rock” and “metal”). Note that some tags lie outside the support of the song distribution. The reason for this is twofold. First, we will see below that a higher-dimensional embedding is necessary to accurately represent the data. Second, many tags are rarely used in isolation, so that some tags may often simply modify the average prior for songs.

To evaluate our method and the embeddings it produces more objectively and in higher dimensions, we now turn to quantitative experiments.

4.2 How does the LME compare to n-gram models?

Our first quantitive experiment explores how the generalization accuracy of the LME compares to that of traditional n-gram models from natural language processing (NLP). The simplest NLP model is the **Unigram Model**, where

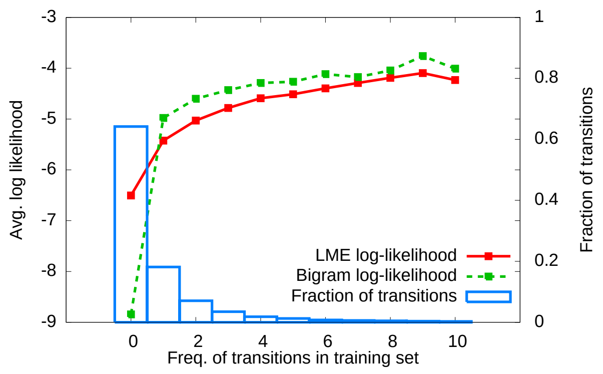


Figure 3: Log-likelihood on testing transitions with respect to their frequencies in the training set for *yes_small*.

the next song is sampled independently of the previous songs. The probability $p(s_i)$ of each song s_i is estimated from the training set as $p(s_i) = \frac{n_i}{\sum_j n_j}$, where n_i is the number of appearances of s_i.

The **Bigram Model** conditions the probability of the next song on the previous song similar to our LME model. However, the transition probabilities $p(s_j|s_i)$ of each song pair are estimated separately, not in a generalizing model as in the LME. To address the the issue of data sparsity when estimating $p(s_j|s_i)$, we use Witten-Bell smoothing (see [5]) as commonly done in language modeling.

As a reference, we also report the results for the **Uniform Model**, where each song has equal probability $1/|\mathcal{S}|$.

Figure 2 compares the log-likelihood on the test set of the basic LME model to that of the baselines. The x-axis shows the dimensionality d of the embedding space. For the sake of simplicity and brevity, we only report the results for the model from Section 3.1 trained without regularization (i.e. $\lambda = 0$). Over the full range of d the LME outperforms the baselines by at least two orders of magnitude in terms of likelihood. While the likelihoods on the big dataset are lower as expected (i.e. there are more songs to choose from), the relative gain of the LME over the baselines is even larger for *yes_big*.

The tag-based model from Section 3.2 performs comparably to the results in Figure 2. For datasets with less training data per song, however, we find that the tag-based model is preferable. We explore the most extreme case, namely songs without any training data, in Section 4.4.

Among the conventional sequence models, the bigram model performs best on *yes_small*. However, it fails to beat the unigram model on *yes_big* (which contains roughly 3 times the number of songs), since it cannot reliably estimate the huge number of parameters it entails. Note that the number of parameters in the bigram model scales quadratically with the number of songs, while it scales only linearly in the LME model. The following section analyzes in more detail where the conventional bigram model fails, while the LME shows no signs of overfitting.

4.3 Where does the LME win over the n-gram model?

We now analyze why the LME beats the conventional bigram model. In particular, we explore to what extent

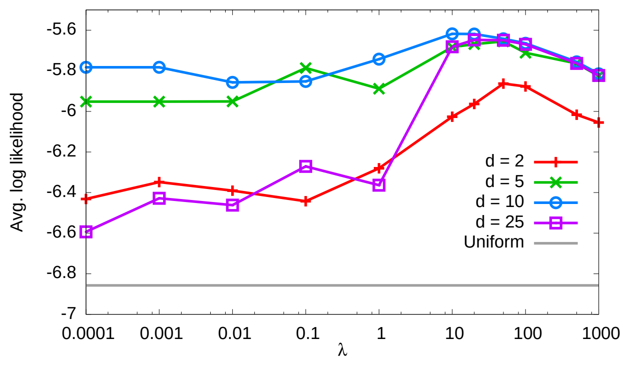


Figure 4: Log-likelihood of predicting transitions for new songs for different d and λ.

the generalization performance of the methods depends on whether (and how often) a test transition was observed in the training set. The ability to produce reasonable probability estimates even for transitions that were never observed is important, since even in *yes_small* about 64% of test transitions were not at all observed in our training set.

For both the LME and the bigram model, the lines in Figure 3 show the log-likelihood of the test transitions conditioned on how often that transition was observed in the training set of *yes_small*. The bar graph illustrates what percentage of test transitions had that given number of occurences in the training set (i.e. 64% for zero). It can be seen that the LME performs comparably to the bigram model for transitions that were seen in the training set at least once, but it performs substantially better on previously unseen transitions. This is a key advantage of the generalizing representation that the LME provides, since it provides an informed way of assigning transition probabilities to all pairs of songs.

4.4 Can the tag model coldstart new songs?

Any playlist generator will encounter new songs it has not been trained on. Fortunately, it is easy to impute an embedding for new songs in our tag-based LME model. Given a new song s with tags $T(s)$, the most likely embedding location according our probabilistic model is

$$X(s) = \frac{1}{|T(s)|} \sum_{t \in T(s)} M(t). \tag{7}$$

To evaluate performance on new songs, we take the *yes_small* dataset and randomly withhold a subset of 30% (951) of the songs which have at least one tag each. We test on these songs and train the tag-based LME on the remaining songs. In particular, we test on transitions from training to test songs, having our model predict based on the imputed test-song location which one of the 951 songs was played.

The only valid baseline for this experiment is the uniform model, since we have no history for the testing songs. The results are shown in Figure 4 for various dimensionalities and regularization parameters λ. Over all parameter settings, the LME outperforms the baseline substantially. Comparing Figure 4 with Figure 2, the gain over uniform for new songs is still roughly half of that for songs that

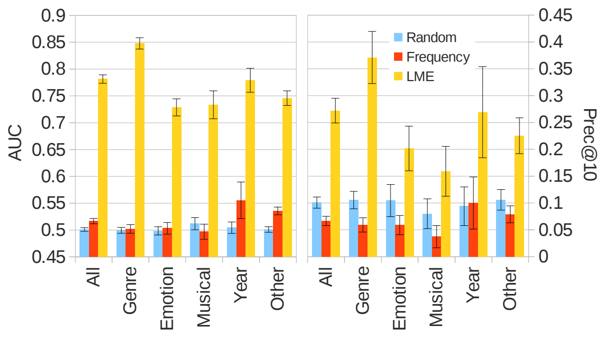


Figure 5: Average AUC (left) and precision at 10 (right) across tag categories for random and frequency baselines and LME. Error bars indicate +/- 1 standard error.

the LME was trained on. This demonstrates that the embedding of the tags captures a substantial amount of the playlist semantics, generalizing well even for new songs.

4.5 Can the embedding space be used for retrieval?

As already demonstrated in the previous section, a powerful property of our model is that it results in a similarity metric that unifies tags and songs – namely, the Euclidean distance of the corresponding points in the embedding. This leads to a natural method for retrieval of songs based on query tags: rank songs by their Euclidian distance to the query tag(s). Note that this method can retrieve songs even though they are not manually tagged with any of the query tags.

To evaluate the effectiveness of the embedding space for retrieval, we now evaluate how well **untagged** songs can be retrieved using queries that consist of a single tag. The experiment is set up as follows. We pooled the train and test partitions of the *yes_small* dataset and then randomly split all songs with at least one tag into 5 partitions. Following a 5-fold cross-validation setup, we removed the tags from the songs in one of the partitions, trained the tag-based LME on the now untagged songs plus the tagged songs from the other 4 partitions, and then computed the query-tag rankings over the untagged songs. For each query tag, we computed the average (over folds) ROC Area (AUC) and Precision@10.

Figure 5 shows the results for the LME and for two baselines: a random ranking of all held-out songs and a ranking of the held-out songs in order of decreasing frequency of appearance in the data set. We separated (by hand) each of the 250 query tags into one of five categories: genre tags (91 tags like rock, hip hop, etc.), emotion tags (35 tags: sad, happy, dark, upbeat etc.), musical and instrumental tags (23 tags: male vocalist, guitar, major key tonality...), years and decades (17 tags), and other tags (84 tags including awesome, loved, catchy, and favorites). For brevity, we only report results for a model with dimension 25 and $\lambda = 10$. However, similar to the results in Figure 4, we find that the exact choice of these parameters is not crucial. For example, the best unregularized model was never more than 4 percentage points worse in AUC than the best regularized model (though naturally for higher dimensions regularization becomes more important).

Our method significantly and substantially outperforms both baselines in every category. Matching our intuition, it does the best for genre queries, with an AUC of nearly 0.85 and Precision@10 of about 37%. The emotion and musical categories prove the most difficult, while the year and other categories are the easiest after genre.

Note that the performance values reported in Figure 5 are extremely conservative estimates of the actual retrieval quality of our method. This is for three reasons: First, social tags can be noisy since they result from ad-hoc labeling practices by non-experts [17]. Second, we made no attempt to identify lexicographically similar tags as the same. For example, consider the following ranking that our method produces for the tag-query "male vocals", with a relevant subset of the tags given for each song:

Daughtry - Home: male vocalists, male vocalist, male
Allen - Live Like We're Dying: male vocalists, male vocalist
The Fray - How To Save A Life: male vocalists, male vocalist
Aerosmith - Rag Doll: male vocalist, malesinger
Lifehouse - Hanging By A Moment: male vocalists, male vocalist, male vocals

Here, all five songs are clearly relevant to the query, but only the last song was considered relevant for the purposes of our experiments. Third, we only test our method on songs for which *no tags at all* were seen during training. For these reasons, it is important to keep in mind that the results we report are strict lower bounds on the actual retrieval performance of our method.

5. CONCLUSIONS

We presented a method for learning to predict playlists through an embedding of songs and tags in Euclidian space. The method not only provides a well-founded probabilistic model for playlist generation, it also produces a distance metric with a probabilistic meaning for song/song, song/tag, and tag/tag distances. We show that the method substantially outperforms conventional sequence models from NLP, that it can sensibly impute the location of previously unseen songs, and that its distance metric is effective for music retrieval even of untagged songs.

The flexibility of the LME approach provides exciting opportunities for future work, since the model leaves open the possibility of more complex representations of songs. For example, instead of representing each song as a single $X(s)$, one can use two embedding vectors $U(s)$ and $V(s)$ to model the beginning and ending of a song respectively. This allows modeling that the ending of song s is compatible with the beginning of song s', but that the reverse may not be the case. Another interesting direction for future work is the modeling of long-range dependencies in playlists. Such long-range dependencies could capture the amount of redundancy/repetition that a user may seek, versus how much a playlist provides variety and explores new music.

This research was supported in part by NSF Awards IIS-1217686, IIS-0812091 and IIS-0905467.

6. REFERENCES

[1] N. Aizenberg, Y. Koren, and O. Somekh. Build your own music recommender by modeling internet radio streams. In *WWW*, 2012.

[2] L. Barrington, R. Oda, and G. Lanckriet. Smarter than genius? human evaluation of music recommender systems. *ISMIR*, 2009.

[3] Shuo Chen, J. L. Moore, D. Turnbull, and T. Joachims. Playlist prediction via metric embedding. In *SIGKDD*, 2012.

[4] D. F. Gleich, L. Zhukov, M. Rasmussen, and K. Lang. The World of Music: SDP embedding of high dimensional data. In *Information Visualization 2005*, 2005.

[5] D. Jurafsky and J.H. Martin. Speech and language processing, 2008.

[6] P. Knees, T. Pohle, M. Schedl, D. Schnitzer, and K. Seyerlehner. A document-centered approach to a natural language music search engine. In *ECIR*, 2008.

[7] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[8] M. Levy and M. Sandler. A semantic space for music derived from social tags. In *ISMIR*, 2007.

[9] B. Logan. Content-based playlist generation: exploratory ex- periments. *ISMIR*, 2002.

[10] F. Maillet, D. Eck, G. Desjardins, and P. Lamere. Steerable playlist generation by learning song similarity from radio station playlists. In *ISMIR*, 2009.

[11] B. McFee, L. Barrington, and G. Lanckriet. Learning content similarity for music recommendation. *IEEE TASLP*, 2012.

[12] B. McFee and G. R. G. Lanckriet. The natural language of playlists. In *ISMIR*, 2011.

[13] E. Pampalk. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, TU Wien, Vienna, Austria, 2006.

[14] J. C. Platt. Fast embedding of sparse music similarity graphs. In *NIPS*, 2003.

[15] B. Schwartz. *The Paradox of Choice: Why More is Less*. Ecco, 2003.

[16] D. Tingle, Y. Kim, and D. Turnbull. Exploring automatic music annotation with "acoustically-objective" tags. In *ICMR*, 2010.

[17] D. Turnbull, L. Barrington, and G. Lanckriet. Five approaches to collecting tags for music. In *ISMIR*, 2008.

[18] J. Weston, S. Bengio, and P. Hamel. Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval. *Journal of New Music Research*, 2011.

EXTRACTING SEMANTIC INFORMATION FROM AN ONLINE CARNATIC MUSIC FORUM

Mohamed Sordo[1], Joan Serrà[2], Gopala K. Koduri[1] and Xavier Serra[1]
[1] Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain.
[2] Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Barcelona, Spain.
{mohamed.sordo, gopala.koduri, xavier.serra}@upf.edu, jserra@iiia.csic.es

ABSTRACT

By mining user-generated text content we can obtain music-related information that could not otherwise be extracted from audio signals or symbolic score representations. In this paper we propose a methodology for extracting music-related semantic information from an online discussion forum, rasikas.org, dedicated to the Carnatic music tradition. We first define a dictionary of relevant terms within categories such as raagas, taalas, performers, composers, and instruments, and create a complex network representation by matching such dictionary against the forum posts. This network representation is used to identify popular terms within the forum, as well as relevant co-occurrences and semantic relationships. This way, for instance, we are able to learn the instrument played by a performer with 95% accuracy, to discover the confusion between two raagas with different naming conventions, or to infer semantic relationships regarding lineage or musical influence. This contribution is a first step towards the automatic creation of ontologies for specific musical cultures.

1. INTRODUCTION

Understanding music requires (also) understanding how listeners perceive music, how they consume it or enjoy it, and how they share their tastes among other people. The online interaction among users results in the emergence of online communities. These interactions generate digital content that is very valuable for the study of many topics, in our case for the study of music. According to [11], an online community can be defined as a persistent group of users of an online social media platform with shared goals, a specific organizational structure, community rituals, strong interactions and a common vocabulary. Our aim in this paper is to study and analyze an online community dedicated to the Carnatic music tradition, rasikas.org.

Carnatic music is the art music of south India [12]. This is a very old and alive tradition with a very engaged and active community. The music lovers of Carnatic music are known as rasikas [1], and their involvement in music related activities and events is fundamental for the preservation and evolution of this music. Interestingly, the interactions between artists and rasikas can influence the evolution of the music concepts in the tradition. For instance, raagas [2], often described as collections of phrases, evolve over time (hence, it is often said that a given raaga today is not the same as it was a hundred years ago). When a performer experiments with a new phrase, rasikas respond to show their appreciation if they believe that the phrase enriched their experience of the raaga.

Websites and online forums have become very relevant venues with which to support and sustain the Carnatic music tradition. Online communities have emerged in which groups or rasikas share music content and discuss among them. Rasikas.org is one such forum in which users engage in many types of discussions, some of them quite engaged, covering most relevant Carnatic music related topics. It clearly does not reflect the whole community of rasikas, but it is an interesting forum from which to learn about Carnatic music and about the opinions of some very passionate and active rasikas. We will be using rasikas.org to perform the experiments in this paper.

Extracting semantic information from online forums has become an important area of research in the last few years. For instance, Yang et al. [15] proposed a method to extract structured data from all types of online forums. Weimer et al. [13] and Chen et al. [2] proposed models to identify high quality posts and topics, respectively. Zhu et al. [16], on the other hand, generated relation networks for topic detection and opinion-leader detection. In addition, a considerable number of approaches devoted to mining user-generated text content have been proposed in the music information retrieval (MIR) community (e.g. [1, 4, 8, 14]). Nevertheless, to the best of our knowledge, none of the highlighted approaches in MIR has exploited the inner structure of online discussion forums.

In this paper we propose a method for extracting semantic information from an online Carnatic music forum, specifically rasikas.org. We define a dictionary of Carnatic music terms and create a complex network representation of the online forum by matching such dictionary against

[1] Rasika, in sanskrit, literally means "the one who derives pleasure".
[2] A raaga is the fundamental melodic framework for composition and improvisation in Indian classical music.

the forum posts. We study different network measures related to the aforementioned network, including node relevance, node co-occurrence and term relations via semantically connecting words. This allows us to obtain meaningful information from the forum's discussions.

The rest of the paper is organized as follows. Section 2 presents the studied online forum, rasikas.org, and the generated Carnatic music dictionary. The methodology for creating a complex network representation of the forum text content is described in Section 3. We present and discuss results related to the aforementioned network measures in Section 4 and conclude in Section 5.

2. DATA GATHERING

2.1 Dictionary

We first build a dictionary that will help us identify and extract Carnatic music terms from a text. For that we gather the editorial metadata of an extensive list of Carnatic music CD's from MusicBrainz.org, an open music encyclopedia. The metadata includes names of recordings, releases, works (compositions), composers/lyricists and performers, and also information about raagas and taalas [3], two key concepts in Carnatic music. To improve our dictionary we also consider the English Wikipedia as an additional source of information. Similar to [10], we obtain a list of Carnatic music terms from dbpedia.org, a machine-readable representation of Wikipedia. We start from the seed category "Carnatic_music" and explore the inherent structure of the dbpedia categorization in order to get all the terms related to the seed. The final dictionary is then created by merging MusicBrainz metadata and Wikipedia categories, and stored as a flat taxonomy of category terms (e.g. raaga–bhairavi, intrument–mdridangam, etc.).

The main problem of this dictionary is that it suffers from noisiness/misspelling errors, mainly due to the diverse transliterations to English of Indian languages terms. For instance, the name *Tyagaraja* (a legendary composer of Carnatic music) can also be written as *Thayagaraja*, *Thiagaraja*, *Tyagayya*, *Thiyagaraja*, *Thagraja*, etc. In order to clean the dictionary, we apply a string matching method based on a linear combination of the longest common subsequence and Levenshtein algorithms [3] to find all duplicate terms, which we manually filter to maintain a single common description for each of them.

2.2 The rasikas.org forum

The text we analyze is extracted from rasikas.org, a dedicated forum of Carnatic music lovers. As many discussion forums, rasikas.org is divided into different sub-forums, 20 in this case. Each forum contains a list of threads and each thread has a number of posts. Typically, a thread is considered to contain a topic which is discussed in all the posts of that thread. It is interesting to note the ratio of number of posts per thread (Table 1). A median value of 5 means that half of the threads have only 5 posts or less. Regarding

[3] Complementary to raaga, taala is the rhythmic framework for composition and improvisation.

Num. topics	$16,595$
Num. posts	$192,292$
Posts per thread	$\mu = 11.59$, $\sigma = 34.49$, $median = 5$
Num. active threads	1362 active in the last 12 months
Num. users	$4,332$ (with at least one post)
Num. active users	929 active in the last 12 months

Table 1. Some statistics of rasikas.org forum.

the forum users, even though there are 4,332 users in total, only a subset of them, 929, have been active in the last 12 months (a user is considered to be active if she has written at least one post in the last N months and M posts overall).

We crawled the entire rasikas.org forum and stored it locally. Not all the sub-forums are of our interest, though. A few of the sub-forums are not directly related to music, whilst some others discuss topics from other music cultures (e.g. Hindustani). We finally selected a subset of the sub-forums that we considered relevant for our study. This made a total of 11 sub-forums, 14,309 threads and 172,249 posts.

3. METHODOLOGY

3.1 Step 1: Text processing

To extract musically-related semantic information from rasikas.org we generate a complex network representation of it. Nonetheless, before building the network, we apply some text processing techniques to match our Carnatic music dictionary against the forum posts. Hence, in the first step, we iterate over the posts of all the topics for a given subset of sub-forums. For each post, the text is tokenized with Penn Treebank, a classical tokenizing technique. The words are then tagged using the Maxent Treebank part-of-speech (POS) tagger. We use the NLTK toolkit [4] implementation of both the tokenizer and the POS tagger. These methods and implementations are classical choices in natural language processing and computational linguistics [5].

Once the text is tokenized and tagged, the method proceeds to match the dictionary of Carnatic music terms against the list of tagged tokens. Given that some terms in the dictionary are word n-grams (i.e. terms with more than one word), the dictionary is sorted by the number of words, matching the longest terms first. Additionally, stemmed adjectives and nouns provided by the POS tagging are also included, except for stop words and words with less than 3 characters. The non-matched words are not removed from the list of tokens, but rather marked as non-eligible. For example, the sentence "the difference between AbhEri and dEvagAndhAram" is converted to "** difference ** AbhEri ** dEvagAndhAram", where ** denotes a non-eligible word.

[4] `http://www.nltk.org`

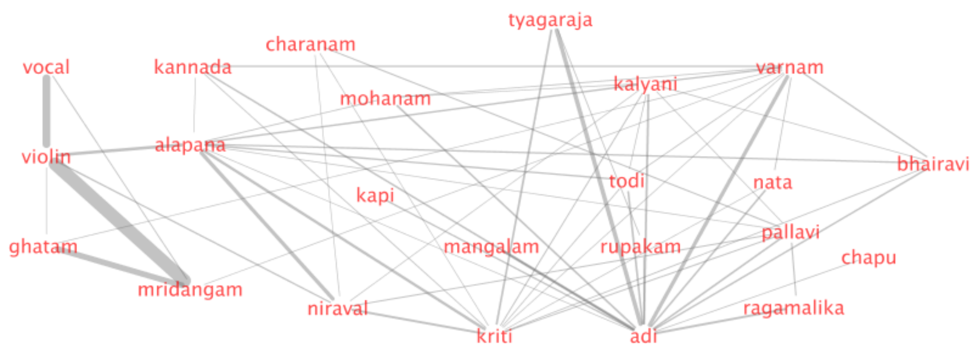


Figure 1. A plot of a subnetwork containing Carnatic music terms with the highest degree. The thickness of the edges represents their weight.

Rank	Raagas	Taalas	Instruments	Performers	Composers
1	Nata	Adi	Violin	Chembai	Tyagaraja
2	Kalyani	Rupakam	Mridangam	Madurai Mani Iyer	Annamacharya
3	Bhairavi	Chapu	Vocal	Charulatha Mani	Purandara Dasa
4	Ragamalika	Jhampa	Ghatam	Kalpakam Swaminathan	Swati Tirunal
5	Kannada	Misram	Morsing	Lalgudi Jayaraman	Papanasam Sivan

Table 2. Top-5 nodes with highest betweenness centrality in the network, organized by category.

3.2 Step 2: Network creation

In the second step, an undirected weighted network is created by iterating over the processed posts. Each matched term is assigned to a node in the network, and an edge/link between two nodes is added if the two terms are close in the text. The link weight accounts then for the number of times two matched terms appear close in the text.

Text closeness is defined as the number of intermediate words between two terms. Thus, we introduce a distance parameter L that will determine which terms are associated with each other. Keeping the non-matched words in the posts (although they are not finally eligible) is important for calculating this distance. Using the example from Step 1, *AbhEri* and *dEvagAndhAram* are considered to be at a distance of $L = 2$. Our assumption is that words that are closer in text are more likely to be related.

3.3 Step 3: Network cleaning

The resulting network from step 2 is very dense (it has $24,420$ nodes and $1,564,893$ links). The average degree is 128.16, which is very high for such a small network. In addition, we find that the network contains a lot of noise. In particular, we find the presence of many spurious nouns and adjectives. Therefore, we introduce a frequency threshold F, which filters out the nouns and adjectives that appear less than F times.

Thresholds L and F yield a more sparse network. However, it could still be possible that some non-statistically significant term relations were reflected in the network links. Thus, we decide to apply a sensible filter to the network topology, the disparity filter [9]. The disparity filter is a local filter that compares the weights of all links attached to a given node against a null model, keeping only the links that cannot be explained by the null model under a certain confidence level α[5]. This confidence level α can be thought of as a p-value ($p = 1 - \alpha$) assessing the statistical significance of a link.

3.4 Network statistics and parameter configuration

After applying the three aforementioned steps, we obtain several network configurations depending on the different parameter values L, F and p. After preliminary assessment, which we omit due to space constraints, we decided to use $L = 5$, $F = 10$ and $p = 0.01$. Fig. 1 shows a subset of the obtained network. With this configuration we obtain a weighted undirected network of $10,928$ nodes (including nouns and adjectives) and $39,067$ edges, resulting in an average degree of 7.15. Furthermore, the network has an average clustering coefficient of 0.309 and a shortest path of 3.658, which are very common values for many other real-world networks [6].

4. RESULTS AND DISCUSSION

4.1 Node betweennes centrality

The resulting weighted network can be analyzed by using different network measures. One of such measures is betweenness centrality. The betweenness centrality measures the importance of a node to the network. It does so by finding the number of shortest paths from all the nodes to all other nodes that pass through that node [6].

[5] The null model assumes that the strength of a given node is homogeneously distributed among all its links.

Parameter configuration	**Num. matched performers**	**Num. matched performer-instrument pairs**	**Hit %**	**MRR**
$F = 10, L = 5, p = 0.01$	104	63	95.24	95.24
$F = 10, L = 10, p = 0.01$	114	70	80.00	85.48

Table 3. Predicted performer/instrument pairs using frequent co-occurrences.

Table 2 illustrates the top-5 nodes with highest betweenness centrality for each term category. Regarding the raagas, *Kalyani* and *Bhairavi* (two major raagas in Carnatic music) are two frequent choices of artists in order to do a Raagam-Taanam-Pallavi, a complete exposition of their skills in a given raaga [12]. *Ragamalika* is not exactly a raaga per se, but a performance where the performer sings more than a raaga in a single composition. In practice, however, since all the combinations are often named as just *Ragamalika*, without referring to the constituent raagas, the term *Ragamalika* ends up being one of the most frequent ones. *Kannada* is unfortunately an ambiguous term. It can refer to a raaga or to the official language of the south Indian state Karnataka. This is probably why the term might have acquired more weight than, for instance, the raaga *Thodi*, which is in principle more popular. As for taalas, *Adi* is the most preferred taala in Carnatic compositions (511 out of 917 recordings in our Carnatic music collection in MusicBrainz are performed in *Adi* taala). In the case of Carnatic music performers, *Lalgudi Jayaraman* is one of the violin trinity of Carnatic music, the three violinists who are considered the irrefutable masters of the art. Moreover, he is also a renowned composer in the modern times, which can explain why he was ranked high in the betweenness centrality. Finally, regarding Carnatic composers, *Tyagaraja*'s compositions constitute a significant proportion of the Carnatic music repertoire (102 out of the 293 works in our Carnatic music collection were composed by *Tyagaraja*). *Annamacharya* is also a composer with a large number of *keerthanas*, many of which are sung either at the beginning or towards the end of a Carnatic music concert. Keerthana is different from, and generally not as elaborate as, the compositional form called Kriti. Kriti forms the crux of the compositional repertoire in Carnatic music, and is considered to have evolved from Keerthana. *Purandara Dasa* is called the father of Carnatic music, both for his compositions and his systematization of Carnatic music learning process. *Papanasam Sivan* was a prominent composer from the state of Tamilnadu. Indeed, he is often referred to as Tamil Tyagaraja. It happens that many users of rasikas.org are from that region too.

4.2 Term co-occurrences

In this section we analyze co-occurrences of Carnatic music terms. There are two possible ways to measure co-occurrence of terms in a network. We distinguish between frequent and relevant co-occurrences.

4.2.1 Frequent co-occurrences

By assuming that terms that co-occur most frequently have a strong relation we can obtain much knowledge from the network. As an example, we will show that network co-occurrent terms allow for correctly guessing the instrument of a performer. This is a non-trivial task whose accuracy can be evaluated objectively, since ground truth data is relatively easy to obtain. In particular, we use the same sources from which we derived the Carnatic music dictionary, Wikipedia and MusicBrainz. From MusicBrainz, we get all the relations between performers and recordings. In the case of Wikipedia, we match a list of instruments (defined in our dictionary) against the text of Wikipedia abstracts. The ground truth is finally filtered by removing the cases where a performer has more than one instrument (vocals are also considered instruments here).

To evaluate the instrument-performer pairing task we use the weight of the links to rank the list of instrument neighbors for each performer (i.e. to rank connected nodes representing an instrument). Two measures are then used to evaluate the predicted instruments in the network: hit percentage and mean reciprocal rank (MRR). The hit percentage refers to correctly predicting the instrument in the first rank. Sometimes, however, the correct answer does not fall in the first rank, but rather in the second, third, or other ranks. This can be measured with MRR, which we define as

$$MRR = \frac{100}{N} \sum_{n=1}^{N} r_n, \qquad (1)$$

where N is the number of co-occurring instruments and r_n is 1 if and only if the n-th instrument is the correct one for a given performer.

As we can observe in Table 3, by considering simple co-occurrences in the network we already achieve an accuracy of 95%, which means that we correctly predict 60 out of the 63 performer-instrument pairs available. For comparative purposes, we also include the results obtained by increasing the link threshold parameter L, which increases the density of the network substantially. Even though the latter configuration increases the number of matched performers and instruments per performers, the accuracy of the predicted instruments drops significantly, meaning that a larger value of L is also adding noise to the network.

4.2.2 Relevant co-occurrences

Apart from evaluating co-occurrences by their frequency, we also compute a relevance score for the co-occurrence.

Raaga	**Raaga**	*Relevance*
Kedaram	Gowla	0.121
Bhavani	Bhavapriya	0.109
Manavati	Manoranjami	0.092
Kalavati	Yagapriya	0.088
Nadanamakriya	Punnagavarali	0.081

Table 4. Relevant co-occurrences of raagas with other raagas.

Raaga	**Composer**	*Relevance*
Abhang	Tukaram	0.159
Yaman kalyani	Vyasa Raya	0.149
Pharaz	Dharmapuri Subbarayar	0.143
Reethi Gowlai	Subbaraya Sastri	0.122
Andolika	Muthu Thandavar	0.108

Table 5. Relevant co-occurences of raagas with composers.

In the network, this means that we compute a relevance weight for the edge between a pair of nodes. The relevance score $R_{i,j}$ for a link between nodes i and j is obtained by

$$R_{i,j} = \frac{w_{i,j}}{\frac{1}{2}(d_i + d_j)}, \tag{2}$$

where $w_{i,j}$ is the weight of the link and d_x is the degree of node x. This score is giving more relevance to the nodes that are more probable to have some relationship [6, 9].

We apply this relevance measure of co-occurrence to combinations of the previously-mentioned term categories. In this experiment we study two such combinations: raaga-raaga and raaga-composer. Tables 4 and 5 show the top-5 more relevant co-occurrences of these two combinations, which we now comment.

Raaga-raaga By looking back at the rasikas.org discussion forum, we can confirm that the co-occurrences found in Table 4 are related to discussions between different raagas, or similar raagas with different naming conventions. For instance, *Bhavapriya* raaga is also called *Bhavani* by disciples of the famous composer Muthuswami Deekshitar. For the co-occurrence of *Kalavati* and *Ragapriya*, a discussion in a forum thread indicates that *Kalavati* and *Ragapriya* are often confused with each other due to a few historic reasons (e.g. naming convention). Some members of the forum posted several facts and technical arguments that are tied to both raagas [6]. It should be noted that although our method helps us to find these important discussions in the forum, it does not have the knowledge that they are in fact discussions. We believe that extracting contextual information of the co-occurrences in the text could help to predict the type of relation between two terms.

[6] `http://www.rasikas.org/forum/viewtopic.php?t=14435`

Raaga-composer Intuitively, a relevant co-occurrence of a raaga and a composer might mean that a composer is known by or uses more frequently a particular raaga. Indeed, that is the case of the relations in Table 5. *Vyasa Raya*'s most famous composition, "Krishna Nee Bagane" is in *Yaman Kalyani* raaga. *Dharmapuri Subbarayar* has a popular composition in *Pharaz* raaga called "Smara Sundaranguni". It is also the case of "Sevikka Vendumayya" composition by *Muthu Thandavar* which is in *Andolika* raaga . Contrasting a little bit with these good agreements, the most relevant raaga-composer co-occurrences include *Abhang*, which is a devotional poetry and not a raaga. This is due to a misleading tag in our vocabulary suggesting that *Abhang* is a raaga name, which suggests that a more accurate cleaning process is needed for some terms.

4.3 Term semantic relations

The aim of this last experiment is to extract semantically meaningful relationships between pairs of Carnatic music terms. From the network perspective, given a pair of nodes, we want to find another node that is connected to both nodes, and that corresponds to a semantically meaningful relationship concept. We call this node a connecting word. For this experiment we use a predefined list of connecting words, including concepts of lineage or family (mother, father, husband, uncle, etc.) and musical influence (guru or disciple) to identify the relationship between pairs of composers and/or performers.

A straightforward approach is to use the same network as before and match the list of predefined connecting words in the common neighbors of a pair of nodes. However, the global nature of the network does not allow to capture the connecting words correctly, since a connecting word can be related to any of the two compared terms separately. Thus, another approach has to be considered. A possible solution is to apply the proposed methodology locally. That is, instead of creating a single, global network, the method described in Sec. 3 can be applied for each post text individually. For each generated small network, we identify all the common neighbors of a pair of composers and/or performers that are related to the concepts of lineage and musical influence. This experiment was evaluated manually using Wikipedia and by asking a Carnatic music expert.

From a total of 24 relations found in the network, our method correctly infers 14 (58% of accuracy). A closer look at the misclassified term relations reveals that many of the wrong predictions were due to ambiguity problems of natural language. For example, our method assigns the relation *guru* for the following pairs of performers: (*Karaikudi Mani*, *G. Harishankar*) and (*Karaikudi Mani*, *Mysore Manjunath*). However, the term *guru* can also be used as a prefix name of a person, in this case *Guru Karaikudi Mani*. Finally, given for instance the following sentence, "Abhisek is grandson of Palghat Raghu and disciple of P.S. Nayaranaswamy", and since the name *Abishek* is not included in our dictionary, our method incorrectly infers a relation of disciple between *Palghat Raghu* and *P.S.*

Nayaranaswamy. It is clear that more advanced natural language processing techniques should be applied in order to solve these ambiguities. Nevertheless, our method is already inferring some relations with a certain amount of confidence, using very simple heuristics.

5. CONCLUSION AND FUTURE WORK

We presented a method for extracting musically-meaningful semantic information from an online discussion forum, dedicated to the Carnatic art music tradition. For that, we defined a dictionary of Carnatic music terms (from concepts such as raagas, taalas, performers, etc.), and created an undirected weighted network by matching such dictionary against the forum posts. Three experiments were ran to study different characteristics of the resulting network, extracting valuable information such as term relevance, term co-occurrence and term relations via semantically connecting words. In the first experiment, nodes with higher betweenness centrality were usually highly correlated with the popularity of Carnatic music terms. In the second experiment, we showed that our method is able to predict the instrument of a performer with a 95% of accuracy. Furthermore, we were able to identify node pairings that are relevant discussions in the forum, using a relevant co-occurrence measure. Even though these discussions were found, our methodology does not have the knowledge that the co-occurrences are in fact discussions. Extracting contextual information of the co-occurrences in the text could help to predict the type of relation between two terms. In the last experiment, we showed that with simple heuristics one can predict semantic relations related to lineage and musical influence with a certain level of confidence.

There are many avenues for future work. Besides the extraction of contextual information and the use of more sophisticated natural language processing techniques, we plan to explore methods that can capture users' opinions using, for instance, algorithms from sentiment analysis [7]. Regarding the forum structure, not all the posts or topics are relevant enough to be added to the network. Therefore, we want to find techniques to impose a confidence value per post, depending on the users' relevance to the forum. Another relevant issue to be tackled is the use of a more complete Carnatic music vocabulary. For that, we plan to manually improve the metadata that can be found in MusicBrainz and to use more resources to increment the size of the dictionary. In order to ease reproducibility of our work and to stimulate further research on the topic, the data and code used for these experiments are publicly-available [7].

6. ACKNOWLEDGMENTS

This research was partly funded by the European Research Council under the European Union's Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583). Joan Serrà acknowledges AEDOC069/2010 from Consejo Superior de Investigaciones Científicas, 2009-SGR-1434 from Generalitat de Catalunya, TIN2009-13692-C03-01 from the Spanish Government, and EU Feder Funds.

[7] `http://www.dtic.upf.edu/\%7Emsordo/research/ismir2012/index.html`

7. REFERENCES

[1] O. Celma, P. Cano, and P. Herrera. Search Sounds: An audio crawler focused on weblogs. In *Proc. of 7th Intl. Conf. on Music Information Retrieval*, Victoria, Canada, 2006.

[2] Y. Chen, X.Q. Cheng, and Y.L. Huang. A wavelet-based model to recognize high-quality topics on web forum. In *Web Intelligence and Intelligent Agent Technology, 2008. IEEE/WIC/ACM Intl. Conf. on*, 2008.

[3] D. Gusfield. *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press, 1997.

[4] P. Lamere. Social tagging and Music Information Retrieval. *Journal of New Music Research*, 37(2):101–114, 2008.

[5] C.D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999.

[6] M.E.J. Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[7] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[8] M. Schedl and T. Pohle. Enlightening the Sun: A User Interface to Explore Music Artists via Multimedia Content. *Multimedia Tools and Applications: Special Issue on Semantic and Digital Media Technologies*, 49(1):101–118, 2010.

[9] M. Serrano, M. Boguñá, and A. Vespignani. Extracting the multiscale backbone of complex weighted networks. *Proc. of the National Academy of Sciences of the USA*, 2009.

[10] M. Sordo, F. Gouyon, and L. Sarmento. A Method for Obtaining Semantic Facets of Music Tags. In *1st Workshop On Music Recommendation And Discovery, ACM RecSys*, 2010.

[11] K. Stanoevska-Slabeva. Toward a community-oriented design of internet platforms. *Intl. Journal of Electronic Commerce*, 6(3):71–95, 2002.

[12] T. Viswanathan and M.H. Allen. *Music in South India*. Oxford University Press, 2004.

[13] M. Weimer and I. Gurevych. Predicting the perceived quality of web forum posts. In *Proc. of the 2007 Conf. on Recent Advances in Natural Language Processing*, 2007.

[14] B. Whitman and S. Lawrence. Inferring Descriptions and Similarity for Music from Community Metadata. In *Proc. of Intl. Computer Music Conf.*, 2002.

[15] J. Yang, R. Cai, Y. Wang, J. Zhu, L. Zhang, and W. Ma. Incorporating Site-Level Knowledge to Extract Structured Data from Web Forums. In *Proc. of the 18th Intl. Conf. on World Wide Web*, 2009.

[16] T. Zhu, B. Wu, and B. Wang. Extracting relational network from the online forums: Methods and applications. In *Emergency Management and Management Sciences, IEEE Intl. Conf. on*, 2010.

RANKING LYRICS FOR ONLINE SEARCH

Robert Macrae
Centre for Digital Music
Queen Mary University of London
robert.macrae@eecs.qmul.ac.uk

Simon Dixon
Centre for Digital Music
Queen Mary University of London
simon.dixon@eecs.qmul.ac.uk

ABSTRACT

When someone wishes to find the lyrics for a song they typically go online and use a search engine. There are a large number of lyrics available on the internet as the effort required to transcribe and post lyrics is minimal. These lyrics are promptly returned to the user with customary search engine page ranking formula deciding the ordering of these results based on links, views, clicks, etc. However the content, and specifically, the accuracy of the lyrics in question are not analysed or used in any way to determine the rank of the lyrics, despite this being of concern to the searcher. In this work, we show that online lyrics are often inaccurate and the ranking methods used by search engines do not distinguish the more accurate annotations. We present an alternative method for ranking lyrics based purely on the collection of lyrics themselves using the Lyrics Concurrence.

1. INTRODUCTION

Multiple copies of song lyrics are available on the internet for almost any song. Due to this free availability, search engines have become the common tool for finding lyrics. As lyrics are relatively easy to mine from the web, and given that the words to a song contain rich semantic information, lyrics are also used for information retrieval such as for karaoke data production, song-browsing, and thumbnailing [2, 3, 12, 15, 16]. The content of a song's lyrics can indicate the topic of the song [4], which genre it belongs to [13], or be used for music indexing and artist similarity [8]. Another example of lyric based information retrieval uses natural language processing to extract language, structure, categorisation, and similarity from lyrics [11].

A contributing factor to the abundance of lyrics and a potential problem for research in this area is the lack of requirements, such as training or language knowledge, that are typically necessary for professionally annotating lyrics. Due to these issues, there is a high potential for song lyrics to contain errors. This can lead to inaccurate lyrics being presented to those using search engines to find lyrics as well as music information retrieval researchers who wish to mine the rich semantic content within lyrics.

In this paper we are concerned with ranking web based song lyrics. Whilst previous work has focused on using multiple sequence alignment to determine the single most accurate lyrics for a song [5, 6], ours is concerned with ranking lyrics, so that users can apply their own selection should the first result not be appropriate. To the best of our knowledge, lyrics ranking has only previously been attempted as part of more generalised web resource ranking methods [14]. In order to evaluate song lyrics ranking we first describe a test data set for this purpose and we then proceed to mine the web for lyrics of the songs in this dataset. We then formulate a metric to compare each lyric to the ground truth, as an accuracy measurement, and to other versions to calculate the Lyrics Concurrence, an adaptation of the Chords Concurrence and Structure Concurrence used to rank guitar tablature [10]. We then adapt the ranking methods outlined previously to evaluate these methods by measuring their correlation with the lyrics' accuracy.

2. TEST DATA: THE MUSIXMATCH DATASET

The Million Song Dataset (MSD) [1] is a collection of metadata for a million popular music tracks [1] produced by LabROSA in collaboration with The Echo Nest. A subset of this data, called the musicXmatch Dataset (MXMD), [2] consists of 237,662 lyrics to songs within the MSD provided in a Bag-of-words format with the 5000 most common (stemmed) words.

2.1 Bag-of-words Format

The Bag-of-words format (BOW) is primarily a means of summarising text by listing the unique words with the number of occurrences of each word in the text, with all punctuation removed. These word and count pairs are ordered by their count with the most common coming first. For example:

"On mules we find two legs behind and two we find before. We stand behind before we find what those behind be for."

can be represented in BOW format as:

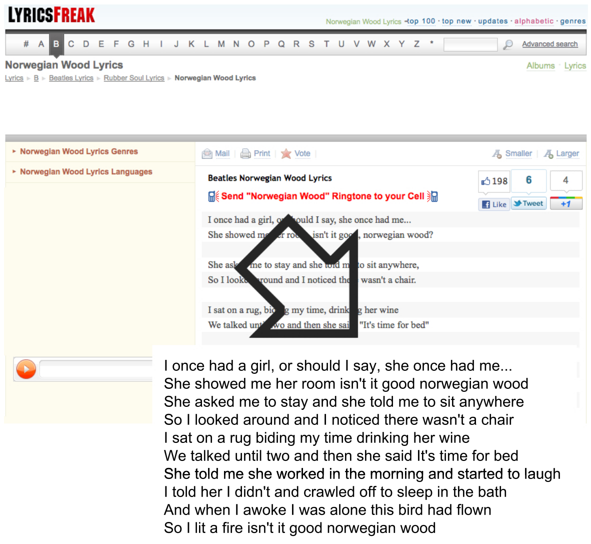
"we:4, find:3, behind:3, two:2, before:2, on:1, mules:1, legs:1, and:1, stand:1, what:1, those:1, be:1, for:1"

[1] http://labrosa.ee.columbia.edu/millionsong/
[2] http://labrosa.ee.columbia.edu/millionsong/musixmatch

Additionally the words are stemmed [9] so that words with different endings are reduced to their root form, reducing the number of unique words. Using this BOW format avoids copyright issues with sharing lyrics for the purposes of research.

3. LYRICS MINING

For each of the 237,662 tracks in the MXMD we searched DogPile [3] for lyrics using the following terms: "<artist><song>lyrics -video". DogPile was chosen as it returns results from all the popular search engines and yet is more easy to data mine. Previous web mining approaches have used the Google Web API in a similar fashion [5, 6], however we required a search engine with an unrestricted number of searches. From the list of URLs returned by this search we selected only those that contained the song title in the URL. This set of URLs provides a similar representation of the URLs a user might select when manually searching for lyrics. 888,745 URLs were found using this method for the 237,662 tracks. In order to extract the lyrics from the URLs we separated and analysed each line to determine whether it contained lyrics-like text and then selected the longest sequence of lyrics-like text lines in the page. Any lyrics that were less than three lines or over 200 lines long were discarded. As we are interested in comparing with Concurrence, we discarded songs and their lyrics if they had less than three lyrics associated with the song. The lyrics extraction process is demonstrated in Figure 1.


LYRICSFREAK
Norwegian Wood Lyrics
Beatles Norwegian Wood Lyrics
Send "Norwegian Wood" Ringtone to your Cell

I once had a girl, or should I say, she once had me...
She showed me her room isn't it good norwegian wood
She asked me to stay and she told me to sit anywhere
So I looked around and I noticed there wasn't a chair
I sat on a rug biding my time drinking her wine
We talked until two and then she said It's time for bed
She told me she worked in the morning and started to laugh
I told her I didn't and crawled off to sleep in the bath
And when I awoke I was alone this bird had flown
So I lit a fire isn't it good norwegian wood

Figure 1. An example lyrics web page and the lyrics extracted from it.

4. LYRICS EVALUATION METRICS

In this section we give an overview of the metrics used in judging the accuracy and similarity of lyrics. The first method, Levenshtein Edit Distance, is a well known Dynamic Programming method for comparing strings. We use the Levenshtein Edit Distance to judge the similarity of lyrics and this is used later in the Lyrics Concurrence ranking method.

[3] http://www.dogpile.com/

4.1 Levenshtein Edit Distance

The Levenshtein Edit Distance (LED) [7] counts the number of "edits" required to transform one string into another. An edit is classed as an insertion, deletion, or substitution of a single character. LED uses a cost of 0 for matches and 1 for any edit (insertion, deletion or alteration). As such the LED of "sun" and "sing" is 2 (substitution of the letter 'u' for 'i' and insertion of the letter 'g'). The LED cost is found by calculating a path $P(U,V) = (p_1, p_2, ..., p_W)$ through a matrix of costs between strings $U = (u_1, u_2, ..., u_M)$ and $V = (v_1, v_2, ..., v_N)$. This cost matrix is described as $d_{U,V}(m,n)$ where $m \in [1 : M]$ and $n \in [1 : N]$ where each position in the path is designated as $p_k = (m_k, n_k)$. A simple bottom-up algorithm for calculating the LED in $O(N^2)$ time and space is shown in Algorithm 1. In this example a matrix of edit costs is calculated between two strings, so that the cell in the final row and column would contain the total number of required edits. Additionally, an example of the "cost matrix" and the solution this algorithm produces can be seen in Table 1.

```
**Input**: String $A$, String $B$
**Output**: Levenshtein Edit Distance $LED$
Matrix $m$; $m[0,0] := (A[0] == B[0]?\ 0 : 1)$;
**for** $a \in [1..A.length]$ **do**
    $m[a,0] := (A[a] == B[0]?\ 0 : 1) + m[a-1,0]$;
**end**
**for** $b \in [1..B.length]$ **do**
    $m[0,b] := (B[b] == A[0]?\ 0 : 1) + m[0,b-1]$;
**end**
**for** $a \in [1..A.length]$ **do**
    **for** $b \in [1..B.length]$ **do**
        $m[a,b] := (A[a] == B[b]?\ m[a-1,b-1] :$
        $1 + min(m[a-1,b], m[a-1,b-$
        $1], m[a,b-1]))$;
    **end**
**end**
**return** $LED := m[A.length, B.length]$;
```

Algorithm 1: The Levenshtein Edit Distance.

4.2 Lyric Accuracy (LA)

In order to calculate the accuracy of the lyrics we first convert the lyrics to the BOW format with the 5000 most common stemmed words (as designated by the MXMD set) using the same stemming code the MXMD set used. We describe the ground truth MXMD BOW $G = (g_1, g_2, ..., g_M)$ and the lyrics BOW $L = (l_1, l_2, ..., l_N)$ as sets of word (w_i) and count (x_i) pairs where $g_i = (w_i, x_i)$. Each word in the ground truth BOW G is looked for in the lyrics BOW L so that if a match is found *i.e.* $g_m(w) = l_n(w)$. Therefore each ground truth word yields an expected word count $g_m(x)$ and a found word count of $l_k(x)$ if the word was present in the lyrics BOW and 0 if not. If the found word

String A: all the other kids
String B: with their pumped up kicks

	a	l	l	t	h	e	o	t	h	e	r	k	i	d	s
w	**1**	2	3	4	5	6	7	8	9	10	11	12	13	14	15
i	2	**2**	3	4	5	6	7	8	9	10	11	12	12	13	14
t	3	**3**	3	3	4	5	6	7	8	9	10	11	12	13	14
h	4	4	**4**	4	3	4	5	6	7	8	9	10	11	12	13
t	5	5	5	**4**	4	4	5	5	6	7	8	9	10	11	12
h	6	6	6	5	**4**	5	5	6	5	6	7	8	9	10	11
e	7	7	7	6	5	**4**	5	6	6	5	6	7	8	9	10
p	8	8	8	7	6	**5**	5	6	7	6	6	7	8	9	10
u	9	9	9	8	7	6	**6**	6	7	7	7	7	8	9	10
m	10	10	10	9	8	7	7	**7**	7	8	8	8	8	9	10
p	11	11	11	10	9	8	8	8	**8**	8	9	9	9	9	10
e	12	12	12	11	10	9	9	9	9	**8**	9	10	10	10	10
d	13	13	13	12	11	10	10	10	10	**9**	9	10	11	10	11
u	14	14	14	13	12	11	11	11	11	**10**	10	10	11	11	11
p	15	15	15	14	13	12	12	12	12	11	**11**	11	11	12	12
k	16	16	16	15	14	13	13	13	13	12	12	**11**	12	12	13
i	17	17	17	16	15	14	14	14	14	13	13	12	**11**	12	13
c	18	18	18	17	16	15	15	15	15	14	14	13	**12**	12	13
k	19	19	19	18	17	16	16	16	16	15	15	14	13	**13**	13
s	20	20	20	19	18	17	17	17	17	16	16	15	14	14	**13**

Table 1. An example of a Levenshtein Edit Distance (LED) requiring 13 edits (with spaces removed).

count is greater than the expected word count, the found count is replaced as the expected count minus the difference or 0 if this difference is greater than the expected count. The LA is calculated as the sum of the found word counts divided by the sum of the expected word counts multiplied by 100 and divided by the sum of the ground truth counts *expected*, so as to be expressed as a percentage. Equation 1 shows this calculation and Table 2 shows an example of the LA measurement.

$$\text{LA}(G,L) = \frac{\sum \max(g_m(x) - |g_m(x) - l_k(x)|, 0)}{\sum g_m(x)} \times 100 \quad (1)$$

Ground Truth: "Are we human or are we dancer? My sign is vital, my hands are cold"
Lyrics: "Are we human or are we dancers? My signs are vital, my hands are cold"
Lyrics Accuracy (LA): $(12/14) \times 100$ = 85.7%
(wrong count for "is" and wrong count for "are")

Table 2. Lyrics Accuracy (LA) example.

4.3 Lyrics Similarity (LS)

The Lyrics Similarity is a measure of how similar two lyrics, L_1 and L_2 are. We use the LED of the entire sequence of characters in both lyrics, not stemmed and with all the punctuation included. We convert the LED to a similarity score by normalising to the perfect score, then inverting and multiplying by 100 to give a value from 0 to 100:

$$\text{LS}(L_1, L_2) = \left(1 - \frac{\text{LED}(L_1, L_2)}{\max(L_1, L_2)}\right) \times 100 \quad (2)$$

For the Lyrics Ranking experiments we additionally tried a variation of the LS called LS_{ns} where spaces are removed from the input lyrics L_1 and L_2. The incentive for removing spaces is that, as the average english word length is 5 characters, spaces make up roughly $\frac{1}{6}$ of the text and possibly contain less relevant information than the rest of the text. As the LED has quadratic costs, reducing the input sequences by $\frac{1}{6}$ reduces the processing time and memory requirements of this method by 31%.

Lyrics 1: "On mules we find two legs behind and two we find before."	
Lyrics 2: "We stand behind before we find what those behind be for."	
Lyrics Similarity (LS):	43.8%
Lyrics Similarity no spaces (LS_{ns}):	45.7%
Lyrics 1: "Are we human or are we dancer? My sign is vital, my hands are cold' '	
Lyrics 2: "Are we human or are we dancers? My signs are vital, my hands are cold"	
Lyrics Similarity (LS):	92.9%
Lyrics Similarity no spaces (LS_{ns}):	90.9%
Lyrics 1: "Scaramouche, Scaramouche, will you do the Fandango"	
Lyrics 2: "Scallaboosh, Scallaboosh, will you to the banned tango"	
Lyrics Similarity (LS):	69.1%
Lyrics Similarity no spaces (LS_{ns}):	66.0%
Lyrics 1: Radiohead - High and Dry (azlyrics.com/lyrics/radiohead/highdry.html)	
Lyrics 2: Jamie Cullum - High and Dry (azlyrics.com/lyrics/jamiecullum/highanddry.html)	
Lyrics Similarity (LS):	86.6%
Lyrics Similarity no spaces (LS_{ns}):	86.0%

Table 3. Lyrics Similarity (LS) examples.

5. LYRICS STATISTICS

The final list of lyrics included 358,535 lyrics for 67,156 songs with an average Lyrics Accuracy of 38.6%. The distribution of the lyrics over these songs can be seen in Figure 2. This lyrics distribution shows a quick drop off in the number of lyrics per song after the songs with less than three lyrics were removed. The range of lyrics accuracy results can be seen in the histogram in Figure 3. The large number of low accuracy lyrics and the low average Lyrics Accuracy suggest the lyrics mining procedure failed to filter out all the non-text lyrics, however, this is not a trivial task for users browsing the web either and so we allow these non lyrics to be considered within the ranking method experiments as one purpose of these methods is to differentiate between lyrics and non-lyrics. In Section 7.1 we examine the possibility of removing these non-lyrics to judge their effect on the ranking experiments. Table 4 shows the top twenty lyrics domains based on their average Lyrics Accuracy. The increase in Lyrics Accuracy of these domains over the average suggests that a simple filter restricting the results to known accurate lyrics domains would remove most of the non-lyrics.

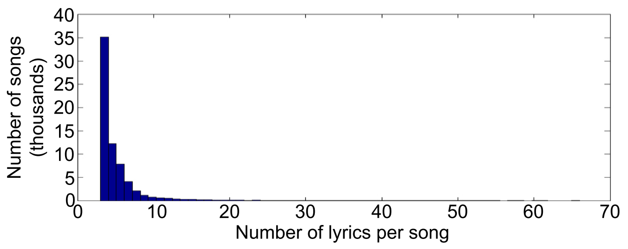


Figure 2. A histogram showing the distribution of lyrics for the 61,755 songs.

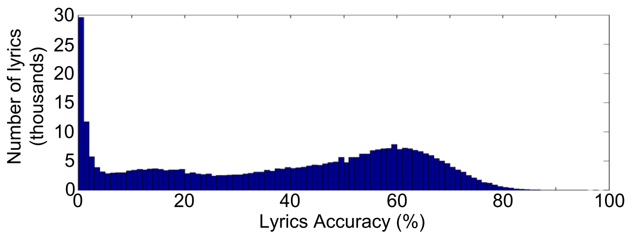


Figure 3. A histogram showing the distribution of the lyrics accuracies.

LA	Domain	Lyrics
55.82%	www.alivelyrics.com	123
52.75%	www.sing365.com	15798
52.53%	www.popular-lyrics.com	142
52.43%	www.plyrics.com	127
52.34%	www.musicsonglyrics.com	3307
52.33%	www.lyricspond.com	535
52.25%	www.songteksten.nl	1178
51.97%	www.lyricsdepot.com	3301
51.93%	www.azlyrics.com	7006
51.30%	www.1songlyrics.com	253
51.11%	www.absolutelyrics.com	1360
51.02%	www.lyricsondemand.com	2909
50.85%	www.sarkisozum.gen.tr	138
50.72%	www.christian-lyrics.net	167
50.62%	www.lyricsdomain.com	925
50.57%	www.lyricstop.com	235
50.084%	www.cowboylyrics.com	1656
49.26%	www.lyriczz.com	682
49.08%	www.lyricsreg.com	1877
49.01%	www.lyricmania.com	155

Table 4. Average accuracy rates for different lyrics domains.

6. LYRICS RANKING METHODS

The following methods describe how we apply the ranking methods to the lyrics.

6.1 Search Engine Results Page Rank

The lyric's Search Engine Results Page Rank (SERP Rank) corresponds to where the URL of the lyric is found in the ordered list of DogPile's ranked search results. Values range from 1 (best) to 100 (worst known), as our mining was restricted to the top 100 results (see Section 3). All the lyrics were mined using DogPile and as such had an associated SERP Rank.

6.2 Date Modified

The Date Modified value is expressed as the number of milliseconds since 00:00:00 January 1, 1970 GMT. 137,875 of the 358,535 lyrics had an associated last date modified that was greater than 0. Any value of 0 is ignored as it was presumed that such a date was unknown.

6.3 Lyrics Concurrence

To determine the extent to which lyrics of songs agree with a set of lyrics, we measure the Lyrics Concurrence as the average of the Lyrics Similarities between a lyric L_k and the other lyrics of the same song $L_i (i \neq k)$.

$$\mathrm{LC}(L_k) = \sum_{i=1, i \neq k}^{n} \mathrm{LS}(L_k, L_i)/(n-1) \qquad (3)$$

6.4 Lyrics Concurrence NS (LC$_{ns}$)

Additionally, we measure the Lyrics Concurrence No Spaces as the average of the LS$_{ns}$ between a lyrics' L_k and the other Lyrics of the same song $L_i (i \neq k)$.

$$\mathrm{LC}_{ns}(L_k) = \sum_{i=1, i \neq k}^{n} \mathrm{LS}_{ns}(L_k, L_i)/(n-1) \qquad (4)$$

7. LYRICS RANKING EVALUATION

In order to measure correlation we use two alternative measurements, the Pearson Product-Moment Correlation Coefficient (PCC), and the Spearman's Rank Correlation Coefficient (SCC). Table 5 shows the correlations found between the lyrics LA and the 4 ranking methods described above. Figure 4 shows scatter graphs of the accuracy and rank of the lyrics using two of the methods: SERP Rank and Lyrics Concurrence. The correlations show the Lyrics Concurrence having the strongest correlation, the SERP Rank having a weak correlation (the negative correlation is expected as lower values indicate a better SERP Rank) and the Date Modified having a very low correlation. Comparing LC and LC$_{ns}$ we find that discarding spaces improves the correlation slightly therefore LC$_{ns}$ improves performance in both accuracy and efficiency. The results of this experiment show that analysing the content of the metadata in comparison to the other metadata available leads to

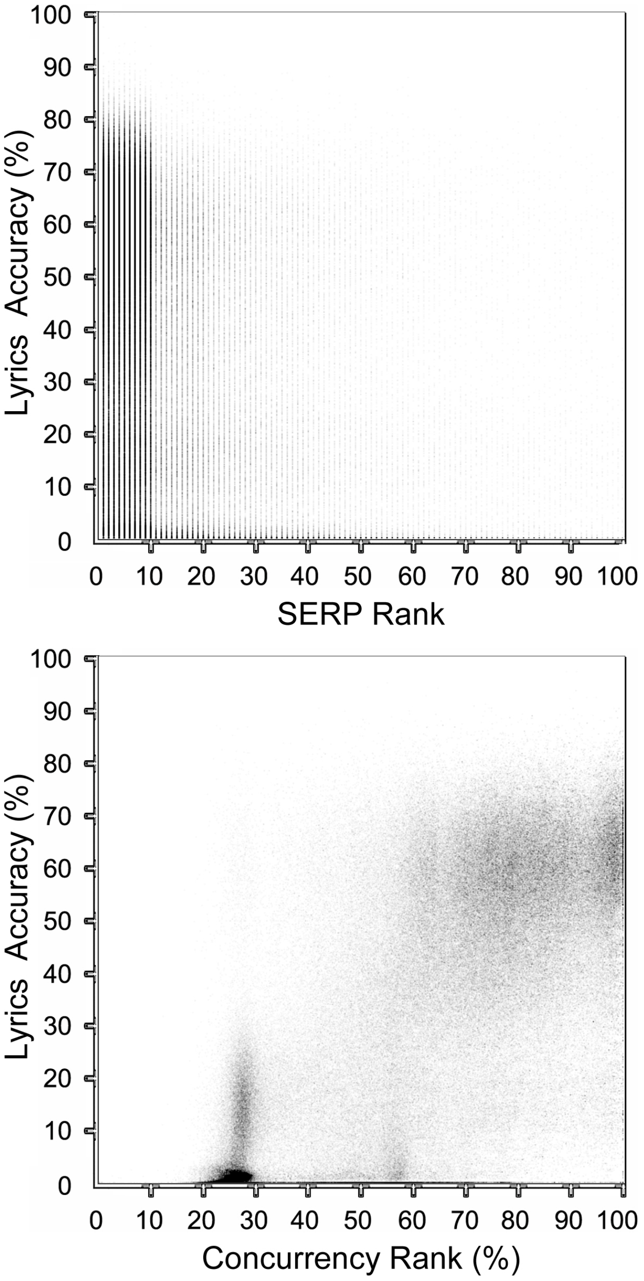


Figure 4. Scatter graphs showing the trends between LA and respectively the SERP Rank (above) and Lyrics Concurrence (below) on 358,535 lyrics.

a better ranking system than methods based on user statistics and link analysis or the date modified.

Ranking Method	PCC (r)	SCC (ρ)	Samples
	LA		
Lyrics Concurrence	0.654	0.607	358535
Lyrics Concurrence NS	0.657	0.609	358535
SERP Rank	-0.206	-0.190	358535
Date Modified	0.016	0.012	137875

Table 5. Number of samples and correlation values between various ranking methods and the Lyrics Accuracy (LA).

7.1 To What Extent do Non-Lyrics Affect Ranking Correlations?

As mentioned previously, the Lyrics data contains many files that are not lyrics at all (as is evident from the dark cluster of low accuracy results in Figure 4) and this may affect the correlations. We therefore repeat the ranking methods experiment excluding the files that have a Lyrics Accuracy of less than 10%, the results of which are shown in Table 6. The ranking methods all see a reduction in the correlation between rank and Lyrics Accuracy. However, this difference also suggests that the methods could be used to help distinguish lyrics from non-lyrics.

Ranking Method	PCC (r)	SCC (ρ)	Samples
	LA		
Lyrics Concurrence	0.477	0.477	289346
Lyrics Concurrence NS	0.484	0.484	289346
SERP Rank	-0.191	-0.191	289346
Date Modified	0.009	0.033	107661

Table 6. A modified version of Table 5 showing correlation values between various ranking methods and the Lyrics Accuracy (LA) without the lyrics with an LA of less than 10%.

7.2 Is Lyrics Concurrence Dependent on Sample Size?

To see if the number of lyrics available for a particular song effects the correlation of Lyrics Concurrence with lyrics Accuracy, we calculate the correlation between N (the number of lyrics for a particular song) and C (correlation between LA and LC) for each of the 61,755 songs. The result, 0.074, is not statistically significant for the sample size, suggesting that Lyrics Concurrence is a relevant indicator of accuracy providing the sample size is at least 3 as is the case in these tests.

7.3 Song Lyrics Detection

We also attempt to use the lyrics ranking methods as lyrics detection systems by taking the highest ranking lyrics for each of the 61,755 songs. Table 7 shows the average accuracy of the ranking methods. Of the ranking methods, the Lyrics Concurrence is the most successful feature for selecting the most accurate lyrics to use.

Detection Method	Lyrics Accuracy
Lyrics Concurrence	47.3%
Date Modified	43.6%
SERP Rank	42.5%
Randomly Selected	38.6%

Table 7. The average Lyrics Accuracy of the top ranked lyrics over 61,755 tracks (41,614 tracks in the case of Date Modified as 38.5% of the lyrics don't have an associated date). The final row shows the average as if the lyrics were randomly selected.

8. DISCUSSION

In this paper we have examined the need for greater ranking of online music metadata and proposed a solution to this problem. The Lyrics Concurrence is a method for ranking music lyrics based on the similarity of its lyrical content to other lyrics of the same song. The rationale of the Concurrence factor is that the correctness of metadata, is determined by agreement of expert human annotators. We have shown that Lyrics Concurrence is a reliable indicator of accuracy, providing a greater correlation with the accuracy of the lyrics than the date modified or SERP Rank. During the time of this experiment there were no ratings available for the lyrics, however, some lyrics websites have started to incorporate this feature. User ratings can act as an additional ranking method and future work could compare this method with those evaluated here, however, a similar study found user ratings to be a poor ranking method for guitar tablature [10].

It is hoped that the Concurrence ranking method can be utilised in search engines to ensure that accurate annotations are ranked more favourably, although the computational costs involved in comparing hundreds of lyrics with each other may limit the usage of such a technique to offline cases. Future ranking methods might focus on combining Concurrence with SERP Rank, User Rating, or linking lyrics with other sources of metadata such as chords, in order to improve the correlation of the ranking with the accuracy. Such an approach may allow a more complete annotation of a different type to fill out any missing or abbreviated segments by repeating the aligned section.

9. REFERENCES

[1] 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)*, pages 591–596, 2011.

[2] Hiromasa Fujihara, Masataka Goto, Jun Ogata, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Automatic Synchronization between Lyrics and Music CD Recordings Based on Viterbi Alignment of Segregated Vocal Signals. In *Proceedings of the 8th IEEE International Symposium on Multimedia (ISM)*, pages 257–264, Washington, DC, USA, 2006.

[3] Min-Yen Kan, Ye Wang, Denny Iskandar, Tin Lay Nwe, and Arun Shenoy. LyricAlly: Automatic Synchronization of Textual Lyrics to Acoustic Music Signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):338–349, 2008.

[4] Florian Kleedorfer, Peter Knees, and Tim Pohle. Oh Oh Oh Whoah! Towards Automatic Topic Detection In Song Lyrics. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 287–292, September 2008.

[5] 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 (ISMIR)*, pages 564–569, London, UK, September 2005.

[6] Jan Korst and Gijs Geleijnse. Efficient Lyrics Retrieval and Alignment. In *Proceedings of the 3rd Philips Symposium on Intelligent Algorithms (SOIA)*, Eindhoven, the Netherlands, December 2006.

[7] V. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics*, 10(8):707–710, February 1966.

[8] Beth Logan, Andrew Kositsky, and Pedro Moreno. Semantic Analysis of Song Lyrics. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, volume 2, pages 827–830, Baltimore,Maryland, USA, 2004.

[9] Julie Beth Lovins. Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics*, 11:22–31, 1968.

[10] Robert Macrae and Simon Dixon. Guitar Tab Mining, Analysis and Ranking. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, pages 453–458, 2011.

[11] Jose P. G. Mahedero, Álvaro MartÍnez, Pedro Cano, Markus Koppenberger, and Fabien Gouyon. Natural Language Processing of Lyrics. In *Proceedings of the 13th Annual ACM International Conference on Multimedia (ACM-MM)*, pages 475–478, New York, NY, USA, 2005. ACM.

[12] Matthias Mauch, Hiromasa Fujihara, and Masataka Goto. Lyrics-to-Audio Alignment and Phrase-Level Segmentation Using Incomplete Internet-Style Chord Annotations. In *Proceedings of the 7th Sound and Music Computing Conference (SMC)*, 2010.

[13] 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 (ISMIR)*, 2008.

[14] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford InfoLab, 1999.

[15] Ye Wang, Min-Yen Kan, Tin Lay Nwe, Arun Shenoy, and Jun Yin. LyricAlly: Automatic Synchronization of Acoustic Musical Signals and Textual Lyrics. In *Proceedings of the 12th Annual ACM International Conference on Multimedia (ACM-MM)*, pages 212–219, New York, NY, USA, 2004.

[16] Chi Wong, Wai Szeto, and Kin Wong. Automatic Lyrics Alignment for Cantonese Popular Music. *Multimedia Systems*, 12:307–323, 2007.

THE ROLE OF MUSIC IN THE LIVES OF HOMELESS YOUNG PEOPLE: A PRELIMINARY REPORT

Jill Palzkill Woelfer
The Information School
University of Washington
Seattle, WA 98195
woelfj@uw.edu

Jin Ha Lee
The Information School
University of Washington
Seattle, WA 98195
jinhalee@uw.edu

ABSTRACT

This paper is a preliminary report of findings in an on-going study of the role of music in the lives of homeless young people which is taking place in Vancouver, British Columbia and Seattle, WA. One hundred homeless young people in Vancouver took part in online surveys, 20 of these young people participated in interviews and 64 completed design activities. Surveys included demographic and music questions. Interviews consisted of questions about music listening and preferences. In the design activities, participants envisioned a music device and provided a drawing and a scenario. Since the study is on-going, findings are limited to descriptive analysis of survey data supplemented with interview data. These findings provide initial insights into music listening behaviors, social aspects of shared music interests, and preferred music genres, bands and artists, and moods.

1. INTRODUCTION

Homelessness is a pressing problem with lasting social and economic consequences. Experts estimate that in a given year, 3 million young people age 12-24 experience homelessness in the U.S and 50,000-60,000 experience homelessness in Canada [11,13]. The number of homeless young people and interest in their welfare has prompted research into their characteristics and circumstances. This extensive research with homeless young people has found that these young people are a heterogeneous group ranging from youth to young adulthood, with varying experiences of abuse and neglect [11,13]. Indeed, much is known about the psychological and social (psycho-social) aspects of homeless young people, but far less is known about their everyday lives, including interests in music, and associated experiences with technologies.

Subsequently, since 2007, the first author has investigated the experiences that homeless young people, aged up to 30, have with technologies, including music players [15,16,17]. One finding arising from this work is that homeless young people have a keen interest in music and use digital means to find and listen to music and share music with others. The current study builds on this prior work by taking a general and exploratory stance, asking: What role does music play in the lives of homeless young people? In response, this paper presents preliminary findings from an on-going study in Vancouver, British Columbia and Seattle, Washington, reporting on data collected in Vancouver in February and March 2012.

2. LITERATURE REVIEW

Consideration of the role that music might play in the lives of young people began in antiquity. Writing in B.C.E. 350, Aristotle proposed that music "might have some influence over the character and the soul" and should therefore "be introduced into the education of the young" [2]. In the 20th century, the music interests of young people living at home have been studied extensively in the psycho-social literature. For example, over 100 studies since the 1970s have focused on possible associations between preferences for particular genres of music or types of music-related media use and risk-taking behaviors, such as drug and alcohol use, high-risk sexual activity, and so on [1]. Despite the high rates of risk-taking behaviors among homeless young people [11,13] and the general lack of knowledge regarding homeless young people's interests in and behaviors related to music, after extensive searching only a single study in the psycho-social literature was found that had investigated music and homeless young people [7].

In a similar fashion, studies in the domain of music information retrieval have investigated the music listening and sharing behaviors of young people living at home. Williams [14] examined issues relating to popular music audiences by conducting unstructured small group discussions with teenagers in England. His subjects stated that music was important in their lives, but "interestingly, they framed its significance in terms of its practical use (of music) in their daily routines," rather than identification or self-construction [14]. Laplante and Downie published a series of studies examining music related behaviors of young adults in Montreal, Canada, specifically on music seeking in everyday life [9], relevance judgments [8], and outcomes of music seeking [10]. In one study, participants reported that informal channels such as friends, colleagues or relatives played a significant role in obtaining music information [9]. In another study, participant's relevance judgments were based on a combination of different criteria, some pertaining to the music itself, but also external factors such as use, disposition, or personal knowledge [8]. Finally, further analysis revealed that participant's satisfaction with music depended on both hedonic (i.e., experiencing pleasure) as well as utilitarian outcomes [10]. Carlisle [5] conducted in-depth online interviews with five young Australians, aged 18-22, and found that each young person wanted digital music for markedly different reasons and had high personal stakes in their musical perspective. Taken together, these studies provide important insights regarding the needs, uses, and music seeking behaviors of young adults in various regions. However, as in the psycho-social literature, searching did not reveal studies with homeless young people. Subsequently, the current study aims to increase knowledge by providing empirical data on the role of music in the lives of homeless young people.

3. STUDY DESIGN

When the current study is complete, all procedures will have been carried out with equal numbers of participants in Vancouver, BC and Seattle, WA. This brief, preliminary report only includes data from homeless young people in Vancouver (collected in February and March 2012) since data collection in Seattle is planned to begin in June 2012. The study uses convenience sampling and is taking place in collaboration with two youth service agencies, one in each city, that provide assistance and shelter for homeless young people. Data in the study is anonymous and names of participants are not collected.

In keeping with the exploratory nature of the goals of the research, the study is a broadly conceptualized, mixed methods design. The procedures with homeless young people include three components: (1) an online survey with up to 100 homeless young people in each city; (2) individual, semi-structured interviews with up to 20 homeless young people in each city; and (3) a self-directed, individual design activity with up to 100 homeless young people in each city. Following approved human subjects protocols, young people were recruited by staff at a homeless youth service agency while attending agency programs. The first author obtained verbal consent from all participants, then introduced and conducted or moderated the surveys, interviews and design activities. The procedures took place sequentially so that a young person first engaged in the survey and was then invited to take part in an interview - until 20 interviews were completed. Finally, all young people who took part in the surveys, whether or not they had been interviewed, were invited to complete the design activity. Homeless young people were compensated with gift cards, from $5-20 depending on the number of procedures. In sum, 100 homeless young people in Vancouver, BC completed surveys, 20 of these people also completed interviews and 64 participated in the design activity.

4. FINDINGS AND DISCUSSION

4.1 Overview

The data analyzed in this paper include self-reported survey responses from 100 homeless young people to ten key demographic and music questions (Table 1) with detail from four additional survey questions. Homeless young people completed surveys on one of three laptop computers. Time needed to complete the surveys ranged from 15-65 minutes (M=35 minutes). Importantly, in order to minimize stress due to the personal nature of the questions, responses to survey questions other than age, gender, race, and sexual orientation were voluntary, resulting in some no responses. Survey participants were 16-24 years old (Mdn=22) and predominately male (63%). Participants identified as heterosexual (68%), bisexual (21%), homosexual (6%), queer (4%), and unsure (1%). Most participants reported their race as White (38%), Aboriginal (27%), or Mixed Race (23%). Half of the participants (50%) had not completed high school (Table 1).

Additionally, regarding homelessness, participants reported that they first became homeless between 2 and 23 years of age (Mdn=16). Participants also reported that they had experienced a total between 0 and 132 months (11 years) of homelessness during their lifetimes. The "0 years, 0 months" responses may indicate that, despite attending programs at an agency that provides services exclusively for homeless young people, some of the participants did not consider themselves to be homeless or they had very recently become homeless. However, these 15 responses should be interpreted cautiously since "0 years, 0 months" was the default answer. Thus, these responses may also indicate skipped questions. If these responses are eliminated, the number of months of homelessness reported by the remaining 85 respondents was 1 to 132 months during their lifetimes (M=24 months).

Regarding music, 97 participants reported that they listened to music on a daily basis and for many reasons with the most frequent responses being related to emotional welfare. Notably, for these questions, there were three "I don't listen to music" responses from two participants. The first participant gave contradicting responses, indicating that she listened to music "2-4 hours a day" on one question and "I don't listen to music" on the other. This may indicate some change in her music listening behavior, although this was not confirmed. The second participant, who took part in an interview, was a guitarist and singer who performed on the street to make money. Explaining her responses, she stated, "I don't listen to music. I play music."

Approximately one-third of the young people indicated that they listened to the same music as their parents, and half indicated other family members. Over half of the young people indicated that they listened to the same music as friends from home or the streets. Indeed, 17 participants also indicated that they made decisions about establishing friendships based on music preferences. More than half of the participants listened most often to hip hop, rock, and rhythm and blues (R&B), although techno and metal were also listened to by nearly half of the participants.

4.2 Discussion

The data presented so far indicate that music does play a role in the lives of homeless young people. In order to elaborate this role, a discussion follows regarding: (1) music-listening behaviors; (2) social aspects of shared music interests; and (3) genres, bands/artists and moods. The discussion is supplemented with evidence drawn from 20 interviews with homeless young people. Since names of participants were not collected in order to preserve participant's privacy, pseudonyms are used to identify participants as needed.

Participants in the interviews were 14 young men and 6 young women, aged 18-24 (Mdn=21). Consistent with prior work [16], interview times ranged widely, from 9-70 minutes (M=45). However, despite homeless young people's general distrust of adults and strangers, with one exception the participants seemed at ease, speaking willingly and at length. Of the 20 interview participants, 13 people had personal music players, such as MP3 players and mobile phones, and 15 people reported that they had music collections ranging from "11-50" to "over 10,000" songs (Mdn=101-500 songs). The size of these song collections may seem surprising. However, in the interviews, some participants stated that since they did not have money to pay for music, they "stole music" and used a variety of "pirate" websites to download songs.

The interviews with homeless young people included questions regarding music listening and preferences, as well as questions related to the importance and influence of music. The interviews began with an activity where young people wrote responses on a 24-hour timeline. When writing on the timelines, participants were asked to indicate what music they listened to, where they were, who they were with, what they were doing, and so on.

After the timeline was filled in, young people rated the importance of music (from very low to very high) and the influence of music (from very negative to very positive) on 5-point Likert scales. The importance of music was rated from low to very high with an average between high and very high (4.125 out of 5). The influence of music was rated from very negative to very positive with an average between neutral and positive (3.75 out of 5). As part of this rating, young people told stories about a time when music was important and a time when music had a positive or negative influence. Finally, young people shared at least one favorite song using a speaker to play

DEMOGRAPHICS	n	WHY DO YOU LISTEN TO MUSIC?*	n
Age		Calm down or relieve tension	77
16-18	9	Help get through difficult times	74
19-21	35	Relieve boredom	72
22-24	56	Get rid of negative feelings/anger	68
Gender		Express feelings or emotions	55
Male	63	Be creative/use imagination	54
Female	36	Wake myself up	51
Transgender	1	Reduce loneliness	48
Ethnicity		Separate myself from society	47
White	38	Get better playing/writing music	34
Aboriginal	27	Create an image	33
Mixed Race	23	Be trendy or cool	16
Black	4	Please my friends	12
Asian	3	Please my parents	3
Arab	1	No response	3
Latino	1	I don't listen to music	2
Other	3	**ON AVERAGE, HOW MANY HOURS A DAY DO YOU LISTEN TO MUSIC?**	
Degree		1 hour or less	26
None	50	2-4 hours	34
High school	38	5-8 hours	19
Trade school	6	9 hours or more	18
2 yr college	3	No response	2
4 yr college	0	I don't listen to music	1
No response	3	**FRIENDS WITH SOMEONE BASED ON THE MUSIC HE OR SHE LIKES?**	
Current job?		Yes	17
Yes	26	No	80
No	73	No response	3
No response	1		

WHAT KINDS OF MUSIC DO YOU LISTEN TO?*	
Hip hop	70
Rock	65
R&B	56
Techno	49
Metal	47
Pop	42
Reggae	38
Punk	35
Jazz	30
Other (includes Dubstep, Country, Rap, Celtic, etc.)	44
WHO LISTENS TO THE SAME MUSIC AS YOU DO?*	
Friends or people you know from home	61
Friends or people you know from the streets	52
Brothers, sisters, cousins, or other family members	50
Boyfriend, girlfriend, or sex partner	44
Friends or people you met online	33
Parents (including foster family or step family)	31
Staff at youth agencies	13
Boss or employer at your job	10
No response	7

Table 1. Self-reported participant characteristics (n=100). *Multiple responses were allowed.

songs from music players, or playing videos on YouTube, or singing songs that they had composed or memorized.

4.2.1 Music-listening Behaviors

In the surveys and interviews, homeless young people reported that they listened to music on a daily basis, more often for practical purposes than for identification or self-construction echoing the findings in [14]. For example, on the one hand, over 70 of the 100 participants reported that they listened to music for practical purposes, such as to "calm down or relieve tension," "help get through difficult times," or "relieve boredom." On the other hand, fewer than 50 participants indicated that they listened to music in order to "separate myself from society," "create an image," "be trendy or cool," or "please my friends (parents)," issues related to identity and self-construction.

As further evidence of the practical aspects of listening to music, the timelines from the interviews indicated that all 20 respondents listened to music while engaging in activities such as waking up and going to sleep, hanging out with friends, and looking for work. For example, Sheila was 22 and listened to music from the time she woke up to the time she went to sleep. Sheila spoke of the importance and practical aspects of music while also highlighting its impact on her emotions, saying:

> *I've been through a lot of f**ked up shit in my life and it's nice…to hear people's opinions…how they dealt with things in music … to kind of relate 'I'm not the only one'…. I just think music is part of my life and I don't think there would ever be a point…where I would say, 'I don't want to listen to music,' 'cause I either want to cry to it or I want to be happy to it or I want to dance around to it, but there's always a song for no matter what emotion you're experiencing.*

In another example, Brian, who was 21, listened to music throughout the day. However, he said he did not listen to music while studying for college classes where he had recently begun to learn how to read and write. Brian rated the influence of music as both very positive and very negative. Speaking of the positive influence, he said that he liked to listen to 1990s rock music such as Bob Seger. Brian said that this music "brings you back to the positive times growing up," particularly playing games with his brother and friends. More recently, he found that:

> *When I'm doing my art I like to listen to MP3. So it kind of calms me so I'm into the artistic zone. Art is my hobby. I just recently discovered that after I quit drinking.*

On the other hand, he found that even some of his favorite songs by the Canadian musician, Matthew Good, could have a negative influence on his emotions:

> *I have post traumatic stress disorder,… some of the music that I listen to kind of triggers me… and makes me feel down. Cause some of the songs I do enjoy are really deep and really sad,… so I get kind of saddened a bit…, so I then actually change the song to try to get on a happier page.*

4.2.2 Social Aspects of Shared Music Interests

Findings in the current study indicate that friends or family are sources of music information [9], and that shared interests play some role in social relationships. In survey and interview responses, homeless young people reported that they shared music interests with friends from home as well as friends from the street. For some interview participants, shared interests with friends led to the desire to attend music concerts, which can be difficult for homeless young people due to their economic circumstances. However, Arthur, age 21, a Dubstep fan who played a favorite song by Flux Pavilion [6], spoke enthusiastically about plans he had to go to an upcoming concert. Arthur and his friend Drew, age 20, another Dubstep fan, also talked in their individual interviews about going together to raves, dance parties where electronic music is played.

Although for most respondents (80%), music was not a determining factor in establishing friendships, 17% responded positively on the survey when asked, "Do you decide whether to be friends with someone based on the music he or she likes?" Indeed, for some participants, not having shared music interests could be potentially isolating. For instance, Matthew, age 20, enjoyed death metal music, and chose a favorite video by Behemoth to play during his interview [4]. Matthew expressed surprise when the first author was not put off by the video and said he rarely listened to his favorite music with other people because they did not share his taste in music.

Participants also reported shared music interests with family. For example, Sheila, introduced above, spoke about choosing the music for her mother's funeral:

> *And I had a lot of good stuff like you know like the classic "Arms of an Angel" kind of thing, but I also had songs like me and her used to listen to that were in it [the funeral], so they [the songs] may not have been like funeral appropriate but they were what we'd listen to.*

In a second example, Amanda, age 24, said that her adoptive mother had introduced her to music and that she liked to sing because it made her mother happy. Amanda had been homeless for nine years, and earned money by singing and playing guitar on the street. A long-time heroin user, Amanda recounted how she had recently "got clean with my music" when she had been invited to sing one of her own songs at a concert. Amanda shared her performance via a video, and said she had been clean for 22 days at the time of the concert, adding:

> *She [Amanda's mother] was sitting right there [in the audience] and she was crying. Everyone was crying. It was a big deal that I was clean.*

4.2.3 Music Genre, Bands/Artists and Mood

Homeless young people reported their music preferences via two different survey questions. One question asked what kinds of music were listened to the most (Table 1). Hip hop emerged as the most preferred category followed by Rock and R&B. However, there is precedence in the literature that music preferences may be gender specific [3]. When broken down by gender, the top three categories preferred by young men were Hip hop, Rock, and Techno, while the top three categories for young women were R&B, Rock, and Hip hop. These differences are potentially noteworthy given that associations between Hip hop (i.e., Rap) music and risk-taking, and the emotional use of music, such as R&B, have been investigated [1].

However, while genre has been used traditionally in studies as a way of gauging young people's music preferences, genre can only go so far in typifying music preferences. Subsequently, in a second question participants named their three favorite bands/artists, resulting in 192 unique responses (out of a possible 300). Table 2 lists the top 17 responses, chosen by three or more participants, and includes mainstream artists, such as Eminem, as well as "underground" bands such as Insane Clown Posse.

Band or Artist	Style [12]	Total	M	F
Eminem	Hardcore Rap	11	7	4
2Pac	Gangsta Rap	9	6	3
Marianas Trench	Punk-Pop	5	2	3
Wiz Khalifa	East Coast Rap	4	3	1
Lady Gaga	Pop	4	2	2
Dr. Dre	Gangsta Rap	3	3	0
AC/DC	Hard Rock	3	2	1
The Notorious B.I.G.	Gangsta Rap	3	2	1
Insane Clown Posse	Rap-Metal	3	2	1
Iron Maiden	Heavy Metal	3	2	1
Korn	Heavy Metal	3	2	1
Skrillex	Dubstep	3	2	1
Lil Wayne	Southern Rap	3	1	2
Nicki Minaj	Hardcore Rap	3	1	2
Nickleback	Heavy Metal	3	1	2
Adele	Pop/Rock	3	0	3
Deadmau5	Club/Dance	3	0	3

Table 2. Favorite bands and musical artists, by male (M) and female (F).

Given that homeless young people reported that listening to music can have an impact on emotional well-being, moods associated with bands/artists were analyzed. Using allmusic.com, 176 unique mood labels were found for 155 of the 192 bands/artists [12]. Of the top 20 moods (Table 3), *aggressive* and *confrontational* appear to have strong negative valence, while *energetic*, *confident*, *rousing*, *brash*, *fun*, *playful*, *freewheeling*, *intense*, *party/celebratory* and *boisterous* appear to indicate high levels of energy or intensity. While no claims can be made about effects of these moods on homeless young people, it appears that the moods in music may be related to reasons for listening to music (Table 1). Recall that Sheila noted how listening to songs with themes related to difficult life experiences could be cathartic; reducing her feelings of isolation and that Brian found that certain songs could trigger his post-traumatic stress disorder symptoms.

Additionally, Marvin, age 24, explained a cathartic effect of his favorite music, saying:

Do you ever get mad?, you kind of get pissed off and you just go to your room and you listen to music. It's either that or you punch your little brother out or something.

Mood	Count	Mood	Count
Energetic	68	Intense	38
Aggressive	66	Party/Celebratory	38
Confident	63	Boisterous	36
Rousing	63	Dramatic	36
Brash	51	Stylish	36
Fun	42	Earnest	35
Playful	42	Reflective	34
Confrontational	41	Passionate	34
Freewheeling	41	Rebellious	33
Fiery	39	Visceral	33

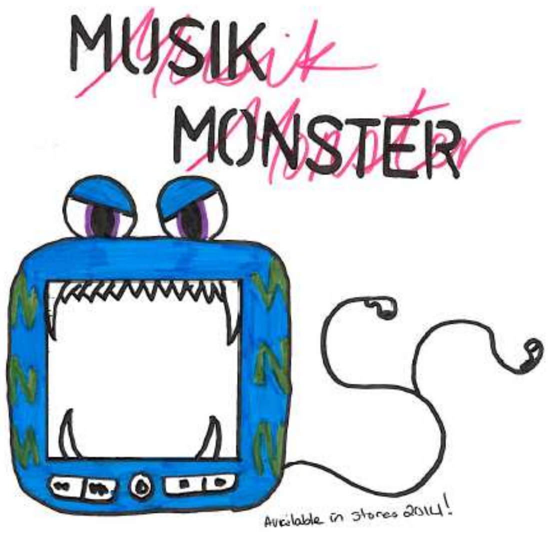
Table 3. Top 20 moods corresponding to 155 bands and artists.

Finally, music with high energy/intensity moods may play a relatively straightforward, practical role as homeless young people move through their daily lives. For instance, recall that on the surveys, participants indicated that they listened to music in order to "wake myself up" and "reduce boredom," and similar answers were given when interview participants completed timelines.

5. CONCLUSION AND FUTURE WORK

In prior and on-going work, findings have revealed that homeless young people have ordinary interactions with technology which are conditioned by the extraordinary circumstances of homelessness [15,16,17]. In this report, we have presented further evidence that indicates aspects of the ordinary and extraordinary. For homeless young people who participated in this study in Vancouver, British Columbia, music appears to play a role in daily life that may be fairly ordinary. Like most young people their age, homeless young people listen daily to a variety of music and music plays a part in their relationships with friends and family. Yet, for homeless young people in this study, experiences with music were also extraordinary. Sheila found comfort in songs that resonated with her difficult life experiences. Amanda played music to make money while living on the street and as part of overcoming her drug addiction. Brian listened to music to regulate aspects of post-traumatic stress disorder, a common result of problematic circumstances during childhood [13].

Importantly, as these results are preliminary, further work is needed to fully elaborate the role of music in the lives of homeless young people. Once the study is completed in Seattle, a comprehensive thematic analysis of the interview data, including the themes of music listening behaviors and social aspects of shared musical interests, with independent cross-coding will take place. Additionally, associations between preferred music, bands and artists, moods and risk-taking behaviors will be analyzed. Finally, results from the design activity where participants envisioned a music device that could help homeless young people (Figure 1) will be analyzed and independently cross-coded. Results from this design activity analysis will provide context and additional evidence of the role of music and associated technologies in the lives of homeless young people.



So I was sitting one day on granville + georgia st, chilling out after a long day of walking. My bags sitting at my side trying to get enough change for a bite to eat. When some lady dropped this thing that looked like an iPod. I ran to pick it up, gave it back to her + being the kickass lady she was she gave the player to me + said it would be better use to me. I asked her what it was + why she was being so nice. She told me she bought it brand new the day before + it wasn't to her likeing [sic]. She said it has lists of shelters + places to get food + their phones numbers in it. So she wanted to help someone out.

I was so greatful [sic]. I was able to escape from reality with beautiful music for a little while. Then it came time for me to find another tree to sleep under. All of a sudden I remembered about the shelter listings on the MP3. (the [sic] lady called it a Musik Monster) First place that came up was [the collaborating agency]. I called, did an intake + now am living happily in my own home. Looking for the next person to help with my Musik Monster.

Figure 1. Design activity drawing and scenario.

6. ACKNOWLEDGMENTS

We would like to thank the study participants and the collaborating service agency. Partial support for the first author was provided by Fulbright Canada.

7. REFERENCES

[1] American Academy of Pediatrics: "Policy Statement – Impact of Music, Music Lyrics, and Music Videos on Children and Youth," *Pediatrics*, Vol. 124, No. 5, pp. 1488-1494, 2009.

[2] Aristotle: Politics – Part V. Trans. by B. Jowett. http://classics.mit.edu/Aristotle/politics.8.eight.html

[3] J.J. Arnett: *Metalheads: Heavy metal music and adolescent alienation*, Westview Press, Boulder, CO, 1996.

[4] Behemoth: *Ov Fire and the Void*, 2009. http://www.youtube.com/watch?v=sKjNk-sAS 7U

[5] J. Carlisle: "Digital Music and Generation Y: Discourse Analysis of the Online Music Information Behaviour Talk of Five Young Australians," *Information Research*, Vol. 12, No. 4, 2007.

[6] Flux Pavilion: *Lines in Wax*, 2012. http://www.youtube. com/watch?v=Dx2nH2RJEAA

[7] M.D. Kipke, J.B. Unger, S. O'Connor, R.F. Palmer, and S.R. LaFrance: "Street Youth, their Peer Group Affiliation and Differences According to Residential Status, Subsistence patterns, and Use of Services," *Adolescence*, Vol. 32, No. 127, pp. 655-669, 1997.

[8] A. Laplante: "Users' Relevance Criteria in Music Retrieval in Everyday Life: An Exploratory Study," Proc. of ISMIR, pp. 601-606, 2010.

[9] A. Laplante and J.S. Downie: "Everyday Life Music Information-Seeking Behaviour of Young Adults," Proc. of ISMIR, pp. 381-382, 2006.

[10] A. Laplante and J.S. Downie: "The Utilitarian and Hedonic Outcomes of Music Information-seeking in Everyday Life," *Library & Information Science Research*, Vol. 33, No. 3, pp. 202-210, 2011.

[11] Public Health Agency of Canada: *Street Youth in Canada: Findings from Enhanced Surveillance of Canadian Street Youth, 1999-2003*, Ottawa, 2006.

[12] Rovi Corporation.: *Allmusic*, 2012. http://www. allmusic. com

[13] L.B. Whitbeck: *Mental health and emerging adulthood among homeless young people*, Psychology Press, New York, 2009.

[14] C. Williams: "Does it Really Matter? Young people and Popular Music," *Popular Music*, Vol. 20, No. 2, pp. 223-242, 2001.

[15] J.P. Woelfer and D.G. Hendry: "Homeless Young People and Technology: Ordinary Interactions, Extraordinary Circumstances." ACM *interactions*, Vol. 28, No. 2, pp. 70-73, 2011.

[16] J.P. Woelfer and D.G. Hendry: "Homeless Young People and Living with Personal Digital Artifacts," Proc. of CHI, pp. 1697-1706, 2011.

[17] J.P. Woelfer and D.G. Hendry: "Homeless Young People's Experiences with Information Systems: Life and Work in a Community Technology Center," Proc. of CHI, pp. 1291–1300, 2010.

A SURVEY ON MUSIC LISTENING AND MANAGEMENT BEHAVIOURS

Mohsen Kamalzadeh
Simon Fraser University
mkamalza@sfu.ca

Dominikus Baur
University of Calgary
dominikus.baur@gmail.com

Torsten Möller
Simon Fraser University
torsten@sfu.ca

ABSTRACT

We report the results of a survey on music listening and management behaviours. The survey was conducted online with 222 participants with mostly technical backgrounds drawn from a college age population. The median size of offline music collections was found to be roughly 2540 songs (sum of physical media and digital files). The major findings of our survey show that elements such as familiarity of songs, how distracting they are, how much they match the listener's mood, and the desire of changing the mood within one listening session, are all affected by the activity during which music is listened to. While people want to have options for manipulating the above elements to control their experience, they prefer a minimal amount of interaction in general. Current music players lack such flexibility in their controls. Finally, online recommender systems have not gained much popularity thus far.

1. MOTIVATION

Since the advent of mp3 files and the fast spread of high bandwidth Internet connectivity, there has been an extreme increase in the number of songs listeners can have immediate access to. In the past decade, the size of personal digital music libraries has seen a similar fast growth. Moreover, subscription based on-demand streaming services like Spotify have made millions of songs readily available to their users. Many studies exist on music listening and management behaviours [2, 4, 10, 11], but with the immense speed at which technology advances, new questions frequently arise on how listeners interact with the immense amount of music available to them.

As Downie et al. [6] point out, one of the main challenges that ISMIR currently faces is encouraging the participation of potential users of Music Information Retrieval (MIR) systems. In this study we investigate the issues such users have in their day-to-day interaction with music. We first divide music interaction into two main categories: (a) music listening, and (b) management of music collections.

Music listening comprises the process of deciding what to listen to in a music listening session and the kind of control exerted by the user on the played music. Management includes obtaining music, managing tags, creating and maintaining playlists, sorting, and so on. Music listening can be both a personal or a social experience. Here, we concentrate on personal music listening. Methods of playback range from low control methods like shuffling one's whole collection along with skipping songs, to higher control ones like having pre-compiled playlists for various occasions or even choosing songs one after another.

Although the amount of user studies on music information retrieval and browsing has been growing as of late, there is a lack of studies when it comes to understanding what factors influence a user's music listening choices in various contexts, what methods of playback are used and why, and what devices and services are more frequently used. Previous studies have focused mainly on users' information seeking behaviours [1,7,9], discovering new music [3], digital music library management [4, 11], use of physical or digital media [2], playlist generation behaviours [10], music listening contexts [2, 5, 8, 10], reasons for listening [5,8], and social aspects of music consumption [10].

In this study, we focus on the act of music listening by investigating our participants' listening behaviours, and trying to understand influential factors in their choice of playback method, and the amount of control and interaction they desire. We compare some of our results regarding playback methods and playlist creation to what Vignoli [11] and Stumpf and Muscroft [10] found. We also study our participants' use of music recommendation services like Grooveshark, iTunes Genius, and Last.fm. Finally, we discuss some implications for the design of future music listening tools.

2. RELATED WORK

There is a close relation between searching (or browsing) and managing libraries, in the sense that the most frequently used cues and properties in searching and browsing can be a good basis for organizing a personal library. This is to some extent confirmed by similar observations by Bainbridge et al. [1], Lee and Downey [9], and Vignoli [11]. In the first two studies, which focus on finding music or music information, the most used properties are reported to be "performer" and "song title". Vignoli [11] asks participants about the attributes they mostly use when retrieving songs from their personal libraries, and again artist name and song title come out on top.

The study by Vignoli [11] is one of the very few that discusses issues relating to the acts of music listening as well as collection management. Vignoli asks participants how often they use various playback methods and reports that the most favorite method is "I choose one or more albums", while "I search for a single song" ranks second. It is notable that these two are both highly controlled experiences, compared to choosing an artist or shuffling the whole collection. Also, users liked to create playlists as opposed to using existing ones, which is also indicative of the higher level of control desired.

Regarding how these playlists are created, a recent paper by Stumpf and Muscroft [10] reports that the concepts most frequently mentioned by participants in a think-aloud playlist creation task were tempo and mood. However, the study had only 7 participants, so a generalization is difficult. In this paper, among other things, we also discuss our own results regarding these attributes and playback methods for listening to music during various activities, and look for similarities with what Vignoli, Stumpf, and Muscroft observed.

3. METHODS

Our online survey included a total of 32 questions covering both collection management (13 questions), music listening (14 questions) and demographics (5 questions). The Likert scale is the most used question format throughout the survey. Medians and modes are used for reporting the results of Likert scales, as opposed to averages.

The survey population consisted mostly of Simon Fraser University's (Canada) Computing Science and Engineering faculty and students who were invited to take part in the survey with mailing lists. We also initiated snowball sampling by encouraging the respondents to spread the word to their friends and family. The questionnaire went through several revisions and was pilot tested with a total of 10 respondents before being sent out to participants. The results presented in this paper were gathered in two stages. The first stage, which targeted only Computing Science graduate students and faculty, had 79 participants. After the first stage, we analyzed the results for questions with low response rates and revised three of them for the second stage, which had 143 participants (Computing Science undergraduates and Engineering students). We present aggregate results for all of the 222 participants in case of identical questions, and stage 2 results for revised questions.

Our participants had an average age of 25.85 with a standard deviation of 9.02, and a median of 23. The majority of our participants (73%) were males, and 95% had a Computing Science or Engineering background.

4. RESULTS

In this section, we present an overview of the results for major questions, in both listening and management categories. Naturally, not all participants answered all the questions, so for each question, only the participants that have answered it are included in computing averages, medians, etc. Whenever we look at results of two or more questions together, we only include the subset of participants who have answered all of them.

4.1 Music Listening

Average hours of music listening per day (both active and passive): During active listening, one is listening to music for the sake of listening, not doing other activities. Passive listening happens when music is listened to during other activities to get in and out of moods, to cancel out ambient noise, to go through boring activities, and so on [5]. Average hours (per day) of active and passive listening were asked in form of ranges. For active listening, these ranges were: Less than 1 hour, 1-2, 2-4, 4-8, and more than 8 hours. In case of passive listening, since it generally happens more than active listening, the choices were changed to cover a larger range: less than 1, 1-2, 2-4, 4-6, 6-8, 8-10, 10-12, and more than 12 hours. For the 174 participants that answered these questions, the median and mode choice for number of active listening hours per day was "less than 1 hour". Both median and mode jump to "2-4 hours" in case of passive listening.

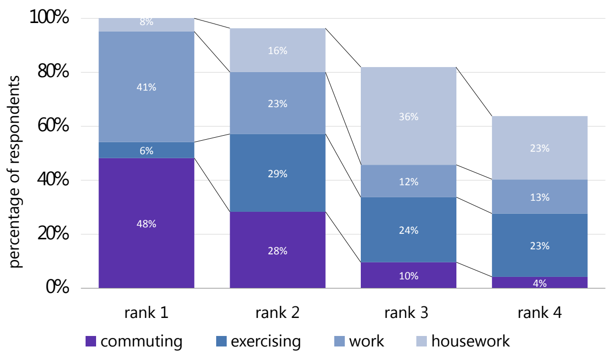


Figure 1. Activities ranked based on their portions of a participant's overall passive listening hours (based on 178 responses).

To understand during what activities passive listening happens, we asked participants to rank 4 activities: commuting, exercising, work, and housework. These are also the top activities reported by Lamont and Webb in [8], except for exercising. This is because we considered exercising as an activity which is reliant on playlists more than the other 3 and could thus broaden our scope when we later ask about playback methods during these activities. As seen in

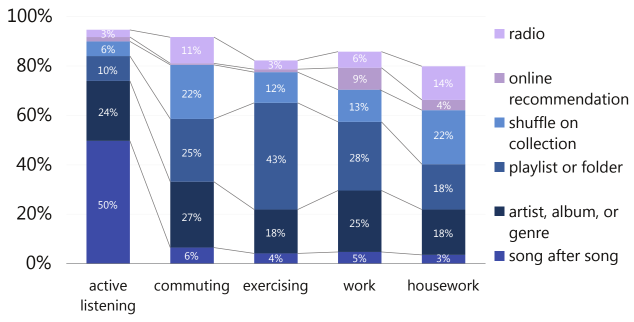
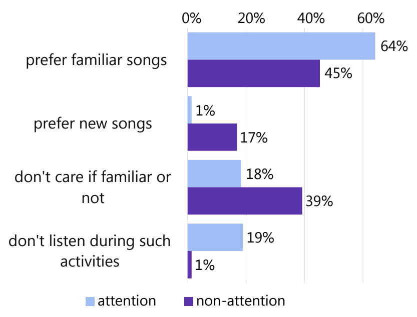
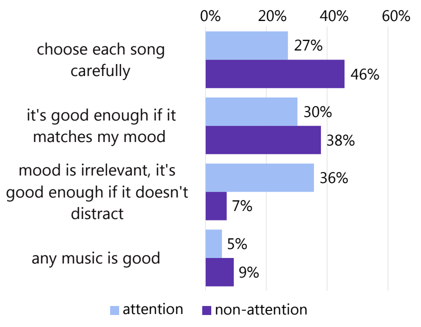


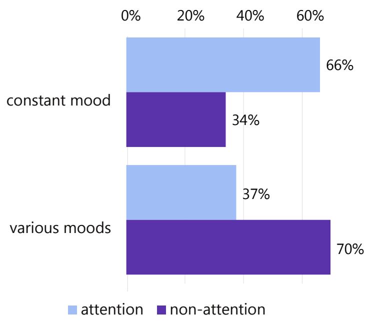
Figure 2. Preferred methods of playback for various simultaneous activities (passive listening) and active listening.



(a) Familiarity of songs



(b) Importance of individual songs / pickiness of the listener



(c) Constant vs. varying mood

Figure 3. Results for questions regarding familiarity, importance, and mood variance of songs.

Figure 1, commuting and work take similarly large chunks of the first rank, with exercise and housework following on ranks two and three. We also provided a comment section for the corresponding question to be able to pinpoint other important activities that we might have missed. Surfing the Internet, playing video games, and during/before sleep were the three activities most often mentioned (13, 10, and 6 times, respectively). Brown et al. [2] found out that the most popular places for listening to music were the car (82% of the time), the living room (61%), and work (38%). Our results show a shift towards work. The reason can be both our population's age and technical background and the fact that with the rapid growth of technology since 2001, nowadays much of people's work happens on their computer which also contains a large collection of music. One distinction between the different activities comes from the amount of attention they need and the amount of control on the music a listener would want. These factors influence the chosen playback methods.

Preferred playback methods for each activity: We study the methods of playback our participants preferred for the same list of activities as before, namely commuting, exercising, work, and housework, along with active listening. For each activity, the respondent was asked to choose one of 6 playback methods. In Figure 2 we see the percentages for each activity and method out of all 169 participants who answered this question. Respondents were told not to choose any method if an activity didn't apply to them, therefore the sum of the columns isn't always 100%.

Choosing song after song dominates the active listening portion and this is not surprising. To figure out which method is generally preferred for passive listening, we exclude active listening results, sum the total number of times each method was chosen and divide that by the number of all the choices made by all participants. We observe that the overall preferred method is "a prepared playlist or folder of songs" with a 29% share. "Picking an artist, album, or genre" and "a shuffle on your whole collection" are second and third with 22% and 19% shares. "Radio, including online stations", "song after song", and "online recommendation services" end up with quite small shares of 8%, 5%, and 4%, respectively. For the same reason as above, the sum of these percentages isn't necessarily 100. In comparison, Vignoli [11] observed that "I choose one or more albums" was the top choice, which is in line with what we see here: an overall preference for higher control. "I search for a single song" is second there, which could be because Vignoli does not classify methods based on activities, resulting in active listening skewing the results.

Importance, familiarity, and mood of songs, and interaction tolerance: As mentioned earlier, a distinction between activities during which music is listened to can be the amount of attention the activities need. We believe that work and commuting (if it is not driving) can lie on two opposing ends of this spectrum, with work needing very high attention from the listener and commuting needing much less. As both these activities contribute heavily in our participants' listening hours, it is crucial to have a better understanding of listening behaviours during each. We hypothesized that having to pay (or not) pay attention to the activity will affect the following 4 aspects:

(a) How familiar the songs are.
(b) How picky the listener usually is (we call this *importance* of songs)
(c) If a constant mood is preferred or if there need to be various moods (in one session of listening).
(d) What the maximum amount of desired interaction is.

Our questionnaire contained a question on each of the above for both activities that need attention (we will call these "attention activities") and those that do not (we will call these "non-attention activities").

Figure 3(a) shows that familiar songs are generally preferred for both attention and non-attention activities, and in case of attention activities, participants strongly preferred familiar songs with nearly 0% preferring new ones. Figure 3(b) shows that while it is important that the music during attention activities does not distract the listener, a large fraction of participants expressed a need for matching moods and choosing each song carefully, even during attention activities. This is expected for non-attention activities, but is somewhat surprising for attention ones, and indicates a general preference for high control on the music. This is in agreement with our results for playback methods discussed earlier. Figure 3(c) shows that although constant mood was the dominant choice for attention activities, still nearly 40% preferred various moods.

To target issue d (maximum amount of desired interaction), we asked participants what their maximum amount

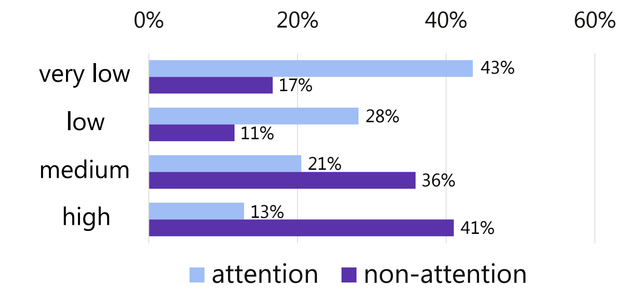


Figure 4. Results from questions on amount of interaction with music player (based on 82 responses).

of desired interaction would be if they wanted to change the mood. One metric for "amount" of interaction can be the amount of time it takes for the user to perform it. Choices included examples that gave our respondents an idea of this time. These were: (a) "very low interaction": e.g. skipping tracks; (b) "low": e.g. specifying your desired change in mood but not having to find any particular song; (c) "medium": e.g. switching to another playlist; (d) "high": e.g. finding specific songs one after another.

The question was more complex in the first stage. Due to high non-response, it was changed to the one described. The results discussed here are from the second stage.

For attention activities (see Figure 4), although a preference for lower interaction is expected, it is interesting to see that along with "very low interaction", "low interaction" was also acceptable by a large margin. During non-attention activities, participants preferred to have higher control on the music source with the medium and high choices dominating the scene.

Use of online music services: It is clear from the above results that online recommendation services are not popular at all even among our survey population which consists mostly of college-age people with technical backgrounds. Indeed, when asked about what music services they have ever used, 33.8% (75 out of 222) said they haven't ever used any of the provided choices (Last.fm, iTunes Genius, Grooveshark, Zune Smart DJ, Pandora, Spotify, iLike, and Musicovery) and didn't provide any other service in the "other" comment section. For the remaining 147 participants, Grooveshark, iTunes Genius, and Last.fm were the most prevalent choices with 46%, 45%, and 42%. 23% had used Pandora, and 7% Spotify. YouTube was the most popular "other" choice with 6 participants (3%). When asked about their favourite service, 61% of participants who answered the question said they didn't normally use these services. The rest of the responses reflect what we have above, with Grooveshark, iTunes Genius, and Last.fm being the top three. This result, however, seems to be more dependent on the popularity of these services than conscious choice as we had only a small number of participants that had tried all or almost all services. For an in-depth analysis more than our 222 participants are needed.

4.2 Management of Music Collections

Primary sources of music (CD, mp3 player, radio, etc.): Participants were asked to choose between 5 frequency ad-

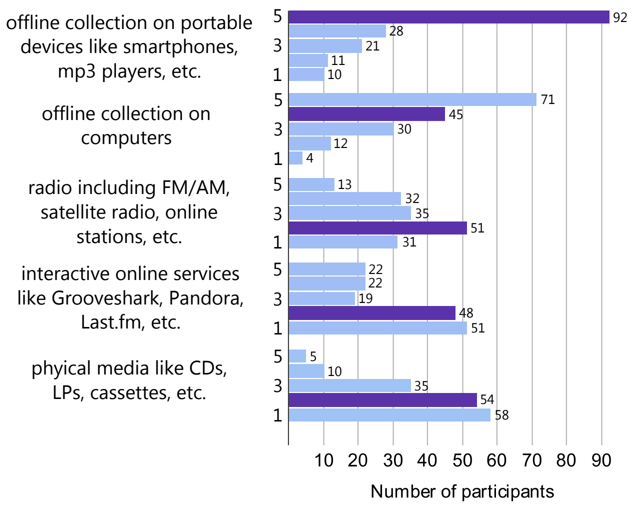


Figure 5. Popularity of music sources. Y axis: 1=Never; 2=Rarely; 3=Sometimes; 4=Often; 5=Very Often. X axis: number of participants who chose each option with the darker bars being the median choice.

verbs in a Likert scale. In Figure 5, the number of participants who chose each choice for each source is shown, with the darker bars being the median for each source. While offline collections on portable devices score the highest, we need to keep in mind that most of our participants have a technical background.

Collection statistics: Our participants had a median of 15 pieces of physical media and 2000 digital songs [1] . Most participants (65%) said they were likely to correct inaccurate tags that they find in their collections. The median respondent maintained between 2 to 4 playlists.

Handling of digital collections (manually with a folder structure, or using an application like iTunes): Almost half of the respondents (83 out of 155: 53%) preferred to manually manage their music folders rather than relying on an application. Applications ended up second with 26%, and 21% said they used both. It appears that management using applications has gained much more traction since Vignoli's study [11] in 2004, which reports that all the 7 participants used manual folders. Participants were also asked about what application they preferred for managing their libraries, with the choices offered being Windows Media Player, iTunes and "other". In Figure 6 we see the choices made by the 144 participants who answered the question, along with the difficulties expressed with each of these applications.

Important factors for managing music collections (album, artist, genre, etc.): Participants were asked to specify how important various factors were for them in managing their collections by choosing between "Very Important", "Important", "Somewhat Important", and "Not Important" for each factor. Artist, with a median of "Very Important" was the top choice here. This confirms the findings reported by Vignoli [11] and Bainbridge et al. [1]. Second were album and genre with a median of "Impor-

[1] For digital collections, participants had a choice of providing number of songs or gigabytes. In cases were gigabytes were provided, they were converted to number of songs, assuming 4 megabytes per song.

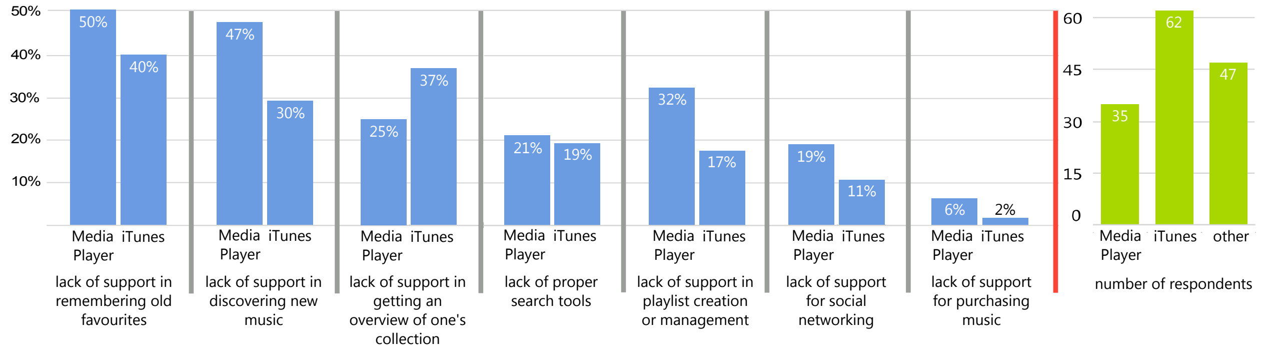


Figure 6. Applications used for library management and difficulties faced with them. Each column shows the perceived lack of support for a task among each application's users. The right-most column shows the number of respondents who chose each application.

tant" and a mode of "Very Important". These were also among the top factors in both the above studies, along with song title.

Important factors in creating playlists: We also asked participants who said they created and managed playlists, about important factors in doing so. Due to high non-response, this question was altered for stage 2 and only stage 2 results are reported in this paper. The format was a three-choice Likert scale for "importance" of factors. This and the question regarding factors important in managing collections were located far apart from each other in the questionnaire. We also altered the choices to not be similar to those offered in the management question, so as to prevent participants from recalling their management answers. The choices for each factor were one of "Not Important", "Somewhat Important", and "Very Important". Mood came out on top with both a median and mode of "Very Important". Genre, artist, and tempo all had a median of "Important" and a mode of "Very Important". In case of mood, our results confirm Stumpf and Muscroft's findings [10], but not for tempo. They reported that tempo was actually the most important factor for their participants, with mood and rhythmic quality being 2nd and 3rd. Here, tempo is only 4th.

To summarize all the results for important factors in management and playlist creation, we scale all of them to a range between 1 and 4 were 4 is the highest score possible for each factor. The results are shown in Table 1.

	playlist creation	management
mood	3.42 (1st)	2.39
genre	3.17 (2nd)	2.52 (3rd)
artist	2.95 (3rd)	3.54 (1st)
tempo	2.80	1.90
album	2.31	2.73 (2nd)
instruments	2.25	1.78

Table 1. Factors important in playlist creation and management of music collections. Scores are out of 4.

5. DISCUSSION

Understanding our participants' listening behaviours starts from knowing when they listen to music. Commuting and work were the most popular activities making up our participants' listening hours. A good music listening tool has to cater to at least the most prominent activities by supporting the playback methods that best fit them.

In Figure 2, we can see that more controlled methods like a playlist or choosing certain artists, albums, or genres are generally more popular than less controlled methods like shuffle, radio, and online recommendation services, and this is more pronounced for "work" and "exercising". This is expected, because exercising requires very specific tempo and rhythm, and work generally needs high attention, so with a shuffle on one's whole collection, the songs are unlikely to satisfy the needed degrees of familiarity, mood, and not being distracting.

But the question is: Are these conventional playback methods enough? According to Figure 3(c), nearly 40% of our participants expressed a desire to have **various moods even during attention activities**. To achieve this, listeners have to resort to switching playlists (assuming they even have prepared ones), applying various filters of artist, album, genre, etc. while listening, or even creating a playlist every time, not to mention choosing songs one after another. According to Figure 4, all of these require amounts of interaction more than what a person would normally want to have with the music source during attention activities. It is interesting to note that 71% of our participants were OK with very low or low interaction. Having in mind that very low interaction is essentially a shuffle on one's collection and that shuffle is not appropriate for many activities, we see a need for novel interaction methods in the "low interaction" range.

We set out to understand what the users would want to have control over, in a Utopian music player. We hypothesized that the familiarity of the songs, their mood and how distracting they are, and if they should have similar moods or not, are among the elements that are affected by the kind of activity during which music is listened to. Judging by Figure 3 and Figure 4 we claim that our hypothesis was confirmed with all the results showing notable differences between attention and non-attention activities.

One could say that online recommendation services like Last.fm, Pandora, and Grooveshark support low interaction while also introducing the listener to new music. Our results show that these services have not really gained traction with users. Several reasons can be speculated for that, like price, sub-par interfaces, availability (different countries), accessibility (computer only or mobile too?). For instance, accessibility can be the reason why online services are used mostly during "work", which is in case of our survey population, mostly done on the computer while online. But there is also the quality of recommendations. Right now, all the noteworthy online recommendation services operate on the basis of similarity to a seed song or a user's library of favourite songs. The maximum control the listener has is skipping songs or in some services, inserting songs into the playlist (e.g. Grooveshark). The listener can have no control over what aspects of the songs are taken into account for computing the "similarity". It is evident from our results that, contrary to the idea behind playlists which are pre-compiled lists for various occasions, there seems to be an inherent impulsiveness in the choice of music. That is, at any point during a session of listening, the listener might want to steer the experience to a new direction. The elements mentioned above are only some of the aspects that should be controllable in this "steering" act. Spotify apps like Moodagent or EchoNest's steerable playlist API are promising developments in this regard.

6. CONCLUSION

The results of studying our 222 participants' music management and listening behaviours were reported and analysed. We discussed how our participants manage their music collections, during what activities they listen to music, how many hours a day they listen to music, if and how they manage playlists, what methods of playback they prefer, what their primary sources of music are, and if they use online recommendation services.

The most important attributes of songs for collection management were artist, album, and genre, which is in agreement with the findings by Vignoli [11] and Bainbridge et al. [1]. We found that mood, genre, and artist were most important for creating playlists, which partly confirms what Stumpf and Muscroft [10] found with 7 participants. They reported that tempo was actually the top choice, with mood and rhythmic quality being 2nd and third, and genre 5th. The median size of personal music collections was found to be 2540 songs. Participants listened to these collections on portable devices and computers more than any other source. This was mostly during commuting and work. Only half of the respondents said they only used manual folder structures for managing their collections rather than applications such as iTunes. This is in contrast with what Vignoli [11] reports from 2004, where all the respondents only used manual folders. The very limited popularity of online music services was surprising to us, considering our population's mostly technical backgrounds and young ages.

Overall, for passive listening (listening to music during other activities), more controlled playback methods like prepared playlists and filters of album, artist, etc. were more popular than shuffling. We discussed these in relation to elements such as familiarity of songs, how distracting they are, how much they match the listener's mood, and if various moods are desired in a session of listening or not, and concluded that there's a need for novel interfaces with easy and efficient support for manipulating these elements dynamically and with a low amount of required interaction.

We would like to note that one issue with our current results is the heavy focus on participants with technical background. To have more reliable results, we are currently extending the survey to other population groups.

7. 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," *Proc. 4th ISMIR*, pp. 221–222, 2003.

[2] B. A. T. Brown, E. Geelhoed, and A. J. Sellen: "The Use of Conventional and New Music Media: Implications for Future Technologies," *Proc. 8th INTERACT*, pp. 67–75, 2001.

[3] S. J. Cunningham, D. Bainbridge, and D. McKay: "Finding New Music: A Diary Study of Everyday Encounters with Novel Songs," *Proc. 8th ISMIR*, pp. 83–88, 2007.

[4] S. J. Cunningham, M. Jones, and S. Jones: "Organizing Digital Music for Use: An Examination of Personal Music Collections," *Proc. 5th ISMIR*, pp. 447–454, 2004.

[5] T. DeNora: *Music in Everyday Life*, Cambridge University Press, 2000.

[6] J. S. Downie, D. Byrd, and T. Crawford: "Ten Years of ISMIR: Reections on Challenges and Opportunities," *Proc. 10th ISMIR*, pp. 13–18, 2009.

[7] J. S. Downie and S. J. Cunningham: "Toward a Theory of Music Information Retrieval Queries: System Design Implications," *Proc. 3rd ISMIR*, pp. 13–17, 2002.

[8] A. Lamont and R. Webb: "Short-and long-term musical preferences: what makes a favourite piece of music?," *Psychology of Music*, vol. 38, no. 2, pp. 222–241, 2009.

[9] J. H. Lee and J. S. Downie: "Survey of Music Information Needs, Uses, and Seeking Behaviours: Preliminary Findings," *Proc. 5th ISMIR*, pp. 441–446, 2004.

[10] S. Stumpf and S. Muscroft: "When Users Generate Music Playlists: When Words Leave Off, Music Begins?," *Proc. ICME 2011*, pp. 1–6, 2011.

[11] F. Vignoli: "Digital Music Interaction Concepts: A User Study," *Proc. 5th ISMIR*, pp. 415–420, 2004.

AUTOMATIC MUSIC TRANSCRIPTION: BREAKING THE GLASS CEILING

Emmanouil Benetos, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri †
Centre for Digital Music, Queen Mary University of London
{emmanouilb,simond,dimitrios,holger,anssik}@eecs.qmul.ac.uk

ABSTRACT

Automatic music transcription is considered by many to be the Holy Grail in the field of music signal analysis. However, the performance of transcription systems is still significantly below that of a human expert, and accuracies reported in recent years seem to have reached a limit, although the field is still very active. In this paper we analyse limitations of current methods and identify promising directions for future research. Current transcription methods use general purpose models which are unable to capture the rich diversity found in music signals. In order to overcome the limited performance of transcription systems, algorithms have to be tailored to specific use-cases. Semi-automatic approaches are another way of achieving a more reliable transcription. Also, the wealth of musical scores and corresponding audio data now available are a rich potential source of training data, via forced alignment of audio to scores, but large scale utilisation of such data has yet to be attempted. Other promising approaches include the integration of information across different methods and musical aspects.

1. INTRODUCTION

Automatic music transcription (AMT) is the process of converting an audio recording into some form of musical notation. AMT applications include automatic retrieval of musical information, interactive music systems, as well as musicological analysis [28]. Transcribing polyphonic music is a nontrivial task and while the problem of automatic pitch estimation for monophonic signals can be considered solved, the creation of an automated system able to transcribe polyphonic music without restrictions on the degree of polyphony or the instrument type still remains open. In this work we will be addressing the problem of polyphonic transcription; for an overview of melody transcription approaches the reader can refer to [39].

† Equally contributing authors. We acknowledge the support of the MIReS project, supported by the European Commission, FP7, ICT-2011.1.5 Networked Media and Search Systems, grant agreement No 287711. E. Benetos is funded by a Queen Mary University of London Westfield Trust Research Studentship. D. Giannoulis and H. Kirchhoff are funded by a Queen Mary University of London CDTA Studentship.

The core problem for creating an AMT system is the detection of multiple concurrent pitches. In past years the majority of multi-pitch detection methods employed a combination of audio feature extraction and heuristic techniques, which also produced the best results in the MIREX multi-F0 (frame-wise) and note tracking evaluations [5,33]. One commonly used technique of these methods is the iterative spectral subtraction approach of [27]. The best performing method in the MIREX multi-F0 and note tracking task is the work by Yeh [45], who proposed a joint pitch estimation algorithm based on a pitch candidate set score function, which is based on several audio features.

Another set of approaches formulates the frame-wise multiple-F0 estimation problem within a statistical framework. The problem can then be viewed as a maximum a posteriori (MAP) estimation problem:

$$\hat{\mathbf{c}} = \arg\max_{\mathbf{c}\in\mathcal{C}} \mathcal{L}(\mathbf{c}|\mathbf{x}) \tag{1}$$

where $\mathbf{c} = \{F_0^1, \ldots, F_0^N\}$ is a set of fundamental frequencies, $\mathcal{C}$ is the set of all possible F0 mixtures, and $\mathbf{x}$ is the observed audio signal within a single analysis frame. If no prior information is specified, the problem can be expressed as a maximum likelihood (ML) estimation problem using Bayes' rule, e.g. [11, 14]. A related method was proposed in [37], using a generative model with a non-homogeneous Poisson process.

Finally, the majority of recent transcription papers utilise and expand *spectrogram factorisation* techniques (e.g. [7, 10]). Non-negative matrix factorisation (NMF) is a technique first introduced as a tool for music transcription in [43]. In its simplest form, the NMF model decomposes an input spectrogram $\mathbf{X} \in \mathbb{R}_+^{K\times N}$ with K frequency bins and N frames as:

$$\mathbf{X} = \mathbf{W}\mathbf{H} \tag{2}$$

where $\mathbf{W} \in \mathbb{R}_+^{K\times R}$ contains the spectral bases for each of the R pitches and $\mathbf{H} \in \mathbb{R}_+^{R\times N}$ is the pitch activity matrix across time. An alternative formulation of NMF called probabilistic latent component analysis (PLCA) has also been employed for transcription (e.g. [22]). In PLCA the matrices in the model are considered to be probability distributions, thus allowing for a model that can be easily extended and formalised. Additional transcription methods have been proposed in the literature, employing sparse coding techniques (e.g. [1]), genetic algorithms (e.g. [40]), and machine learning algorithms (e.g. [38]), which due to space limitations cannot be detailed here.

For note tracking, hidden Markov models (HMMs) are frequently used at a postprocessing stage (e.g. [38]). Other

Participants	2009	2010	2011
Yeh and Roebel	0.69	0.69	0.68
Dressler	-	-	0.63
Benetos and Dixon	-	0.47	0.57
Duan, Han, and Pardo	0.57	0.55	-

Table 1. Best results using the accuracy metric for the MIREX Multi-F0 estimation task, from 2009-2011. Details about the employed metric can be found in [33].

techniques include temporal smoothing (e.g. using a median filter) and minimum duration pruning [10].

In the remainder of this paper we analyse limitations of current approaches and identify promising directions for overcoming the obstacles in current performance.

2. CHALLENGES

Despite significant progress in AMT research, there exists no end-user application that can accurately and reliably transcribe music containing the range of instrument combinations and genres available in recorded music. The performance of even the most recent systems is still clearly below that of a human expert, who requires multiple takes, makes extensive use of prior knowledge and complex inference, and produces imperfect results. Furthermore, current test sets are limited in their complexity and coverage. Table 1 gives the results for the frame-based multiple-F0 estimation task of the MIREX evaluation [33]. Results for the note tracking task are much inferior, in the range of 0.2–0.35 average F-measure with onset-offset detection and 0.4–0.55 average F-measure with onset detection only. As we propose in Section 3, informing transcription via user-assistance or by providing a draft score in some applications are ways to increase systems' performance and overcome the observed plateau.

Currently proposed systems also fall short in flexibility to deal with diverse target data. Music genres like classical, heavy metal, hip-hop, ambient electronic and traditional Chinese music have little in common. Furthermore styles of notation vary with genre. For example Pop/Rock notation might represent melody, chords and (perhaps) bass line, whereas a classical score would usually contain all the notes to be played, and electroacoustic music has no standard means of notation. The task of tailoring AMT systems to specific styles has yet to be addressed. In Section 4 we propose systems focusing on instrument- or genre-specific transcription.

Algorithms are developed independently to carry out individual tasks such as multiple-F0 detection, beat tracking and instrument recognition. Although this is necessary, considering the complexity of each task, the challenge remains in combining the outputs of the algorithms, or better, combining the algorithms themselves to perform joint estimation of all parameters, in order to avoid the cascading of errors when the algorithms are combined sequentially. In Section 5, we propose the fusion of information across multiple musical aspects and the combination of methods targeting the same feature.

Another challenge concerns the availability of data for training and evaluation. Although there is no shortage of transcriptions and scores in standard music notation, human effort is required to digitise and time-align them to the recordings. Except for the case of solo piano, data sets currently employed for evaluation are small: a small subset from the RWC database [20] which contains only 12 tracks is commonly used (although the RWC database contains many more recordings) and the MIREX multi-F0 recording lasts only 54 seconds. Such small datasets cannot be considered representative; the danger of overfitting and thus overestimating system performance is high. It has been observed for several tasks that dataset developers tend to attain the best MIREX results [33]. In Section 6, we discuss ways to generate more training data.

At present, there is no established single unifying framework for music transcription as HMMs are for speech recognition. Likewise, there is no standard method for front end processing of the signal, with various approaches including STFT, constant Q transform [8] and auditory models, each leading to different mid-level representations. The challenge in this case is to characterise the impact of such design decisions on the AMT results. In Section 7, we consider the implications and steps required to progress from existing systems to complete transcription.

In addition to the above, the research community shares code and data on an ad hoc basis, with poor management and restrictive licensing limiting the level of re-use of research outputs. Many PhD students, for example, start from scratch spending valuable time "reinventing wheels" before proceeding to address current research issues. The lack of standard methodology is a contributing factor, with the multiplicity of approaches to AMT making it difficult to develop a useful shared code-base. The Reproducible Research movement [9], with its emphasis on open software and data, provides examples of best practice which are worthy of consideration by our community.

Finally, present research in AMT introduces certain challenges in itself that might constrain the evolution of the field. Advances in AMT research have mainly come from engineers and computer scientists, particularly those specialising in machine learning. Currently there is minimal contribution from computational musicologists, music psychologists or acousticians. Here the challenge is to integrate knowledge from these fields, either from the literature or by engaging these experts as collaborators in AMT research.

AMT research is quite active and vibrant at present, and we do not presume to predict what the state of the art will be in the next years and decades. In the remainder of the paper we propose promising techniques that can be utilised and further investigated in order to address the aforementioned limitations in transcription performance. In Fig. 1 we provide a general diagram of transcription, incorporating techniques discussed in the following sections.

3. INFORMED TRANSCRIPTION

3.1 Semi-automatic Approaches

Semi-automatic or *user-assisted transcription* refers to approaches where the user provides a certain amount of prior information to facilitate the transcription process [26]. Although such systems are not applicable to the analysis of large music databases, they can be of use for musicians, musicologists, and—if a suitable synthesis method exists—for intelligent audio manipulation.

AMT systems usually have to solve a number of tasks, the nature of which depends on the type of music analysed and the level of detail required for the score representation. While some of these tasks might be quite easy for a human listener, it is often difficult to find an algorithmic formulation. The advantage of semi-automatic approaches is the fact that certain tasks that are inherently difficult to solve algorithmically can be assisted by the user of the system. Semi-automatic transcription systems might also pave the way for more robust fully-automatic ones, because the possibility of replacing the human part by an equally-performing computational solution always exists.

In principle any acoustic or score-related information that can facilitate the transcription process can act as prior information for the system. However, to be of use in a practical application, it is important that it does not require too much time and effort, and that the required information can be reliably extractable by the user, who might not be an expert musician.

Depending on the expertise of the targeted users, information that is easy to provide could include key, tempo and time signature of the piece, structural information, information about the instrument types in the recording, or even asking the user to label a number of notes for each instrument. Although many proposed transcription systems -often silently- make assumptions about certain parameters, such as the number or types of instruments in the recording, not many published systems explicitly incorporate prior information from a human user. In the context of source separation, Ozerov et al. [36] proposed a framework that enables the incorporation of prior knowledge about the number and types of sources, and the mixing model. The authors showed that by using prior information, a better separation can be achieved than with completely blind systems. A system for user-assisted music transcription was proposed in [26], where the user provides information about the instrument identities or labels a number of notes for each instrument. This knowledge enabled the authors to sidestep the error-prone task of source identification or timbre modelling, and to evaluate the proposed non-negative framework in isolation.

3.2 Score-informed Approaches

Contrary to speech, only a small fraction of music is fully spontaneous, as musical performances are typically based on an underlying composition or song. Although transcription is usually associated with the analysis of an unknown piece, there are certain applications for which a score is available, and in these cases the AMT system can exploit this additional knowledge [42]. For example in automatic instrument tutoring [6, 44], a system evaluates the performance of a student based on a reference score and provides feedback. Thus, the correctly played passages need to be identified, along with any mistakes made by the student, such as missed or extra played notes. Another example application is the analysis of expressive performance, where the tempo, dynamics, and timing deviations relative

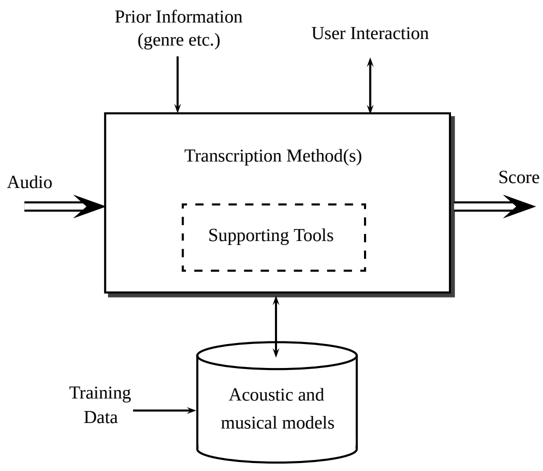


Figure 1. General overview of transcription. Supporting tools refer to techniques which can facilitate transcription, e.g. key estimation, instrument recognition.

to the score are the focus of the analysis. There are often small differences between the reference score and the performance, and in most cases, the score will not contain the absolute timing of notes and thus will need to be time-aligned with the recording as a first step.

One way to utilise the automatically-aligned score is for initialising the pitch activity matrix $\mathbf{H}$ in a spectrogram factorisation-based model (see Eq. (2)), and keeping these fixed while the spectral templates $\mathbf{W}$ are learnt, as in [16]. After the templates are learnt, the gain matrix can also be updated in order to cater for note differences between the score and the recording.

4. INSTRUMENT- AND GENRE-SPECIFIC TRANSCRIPTION

Current approaches for AMT usually employ instrument models that are not restricted to specific instrument types, but applicable and adaptable to a wide range of musical instruments. In fact, most transcription algorithms that are based on heuristic rules and those that employ human sound perception models even deliberately disregard specific timbral characteristics in order to enable an instrument-independent detection of notes. Even many so-called piano transcription methods are not so much tailored to piano music as *tested* on such music; they do not implement a piano-specific instrument model. Similarly, the aim of many transcription methods is to be applicable to a broad range of musical genres.

The fact that only a small number of publications on instrument- and genre-specific transcription exist, is particularly surprising when we compare AMT to the more mature discipline of automatic speech recognition. Continuous speech recognition systems are practically always language-specific and typically also domain-specific, and many modern speech recognisers include speaker adaptation [24].

Transcription systems usually try to model a wide range of musical instruments using a single set of computational methods, thereby assuming that those methods can be applied equally well to different kinds of musical instruments.

However, depending on the sound production mechanism of the instruments, their characteristics can differ considerably and might not be captured equally well by the same computational model or might at least require instrument-specific parameters and constraints if a common model is used. Furthermore, acoustic instruments incorporate a wide range of playing styles, which can differ notably in sound quality. On the other hand we can revert to the extensive literature on the physical modelling of musical instruments. A promising direction is to incorporate these models in the transcription process or at least use them as prior information that can then be adapted to the recording under analysis. Some examples of instrument-specific transcription are for violin [29], bells [30], tabla [19] and guitar [3].

The application of instrument-specific models, however, requires the target instrumentation either to be known or inferred from the recording. Instrument identification in a polyphonic context, as opposed to monophonic, is rendered difficult by the way the different sources blend with each other, with a high degree of overlap in the time-frequency domain. The task is closely related to sound source separation and as a result, many systems operate by first separating the signals of different instruments from the mixture or by generating time-frequency masks that indicate spectral regions that belong only to a particular instrument which can then be classified more accurately [13]. There are also systems that try to extract features directly from the mixture or by focusing on time-frequency regions with isolated note partials [4]. A review of instrument identification methods can be found in [34, sect. IV].

The advantage of restricting a transcription system to a certain musical genre lies in the fact that special (expert) knowledge about that genre can be incorporated. Musicological knowledge about structure (e.g. sonata form), harmony progressions (e.g. 12-bar blues) or specific instruments (e.g. Irish folk music) can enhance the transcription accuracy. Genre-specific AMT systems have been designed for genres such as Australian aborginal music [35]. In order to build a general-purpose AMT system, several genre-specific transcription systems could be combined and selected based on a preliminary genre classification stage.

5. INFORMATION INTEGRATION

5.1 Fusing information across the aspects of music

Many systems for note tracking combine multiple-F0 estimation with onset and offset detection, but disregard concurrent research on other aspects of music, such as instrumentation, rhythm, or tonality. These aspects are highly interdependent and they could be analysed jointly, combining information across time and across features to improve transcription performance.

A human transcriber interprets the performed notes in the context of a metrical structure consisting of a semi-regular, hierarchical system of accents. Extensive research has been performed into tempo induction, beat tracking and rhythm parsing [21], but transcription rarely takes advantage of this knowledge. An exception is the use of beat-synchronous features in chord transcription [31], where the audio is segmented according to the location of beats, and features are averaged over these beat-length intervals. The advantage of a more robust feature (less overlap between succeeding chords) is balanced by a loss in temporal resolution (harmonic change is assumed not to occur within a beat). For note transcription, it is unrealistic to assume that notes do not change within beats, but a promising approach would be to use a similar technique at a lower (i.e. sub-beat) metrical level, corresponding to the fastest note sequences. The resulting features would be more robust than frame-level features, and advantage could be taken of known (or learnt) rhythmic patterns and effects of metrical position.

Key is another high-level musical cue that, if known or estimated from the signal, provides useful prior information for the extraction of notes and chords. Key can be modelled as imposing a probability distribution over notes and chords for different metrical positions and durations. Therefore, by specifically modelling key, transcription accuracy can be improved, e.g. by giving more weight to notes which belong to the current key. Genre and style are also influential factors for modelling the distribution of pitch classes in a key. Several key estimation approaches have been proposed, but these are rarely exploited for AMT, with the exception of [41], which gave the best results for the MIREX 2008 note tracking task.

Likewise, local harmony (the current chord) can be used to inform note transcription. The converse problem, determining the chord given a set of detected notes, is also a transcription task. A chord transcription system which uses a probabilistic framework to jointly model the key, metre, chord and bass notes is presented in [31].

Finally, information can also be integrated over time. Most AMT systems to date have modeled only short-term dependencies, often using Markov models to describe expected melodic, harmonic and rhythmic sequences. As a notable exception, [32] utilized structural repetitions for chord transcription. Also the musical key establishes a longer-term (tonal) context for pitch analysis.

5.2 Combining methods targeting the same feature

Information could also be integrated by combining multiple estimators or detectors for a single feature, for instance combining two multi-pitch estimators, especially if these are based on different acoustic cues or different processing principles. This could help overcome weak points in the performance of the individual estimators, offer insight on the weaknesses of each and raise the overall system accuracy. In a different context, several pitched instrument onset detectors, which individually have high precision and low recall, have been successfully combined in order to obtain an improved detection accuracy [23]. For classification, adaptive boosting (AdaBoost) provides a powerful framework for fusing different classifiers in order to improve the performance [17].

5.3 Joint transcription and source separation

Source separation could be of benefit to transcription-related tasks such as instrument identification, where both tasks are interdependent, and accomplishing one of them could significantly ease the other. In this spirit, joint source separation and musical instrument identification methods have been proposed using signal model-based probabilistic inference in the score-informed case [25]. Also, ideas and algorithms from the field of source separation can be utilised for AMT, especially regarding the exploitation of spatial information, if this is available [12, 36].

However, for most AMT tasks there is only one or two mixture signals available, and the number of sources is larger than the number of mixtures. In this case, the separation task is underdetermined, and can only be solved by requiring certain assumptions to hold for the sources. These could include sparsity, non-negativity and independence or they could involve structured spectral models like NMF models [22], spectral Gaussian scaled mixture models (Spectral-GSMMs) [2] or the source-filter model for sound production. Further constraints such as temporal continuity or harmonicity can be applied on spectral models. Techniques that employ spectral source modelling or an NMF-based framework that explicitly models the mixing process of the sources have been shown to perform well because they exploit the statistical diversity of the source spectrograms [2].

Finally, source separation can be fully utilised in a semi-supervised system like [12], where the user initially selects the desired audio source through the estimated F0 track of that source and subsequently the system refines the selected F0 tracks, and estimates and separates the relevant source.

6. CREATING TRAINING DATA

A large subset of AMT approaches perform experiments only on piano data, e.g. [10, 14, 38]. One reason is because it is relatively easy to create recordings with aligned ground-truth using e.g. a Disklavier. However, this emphasis on piano music sometimes leads to models that are tailored for pitched percussive instruments and could also be a cause for overfitting. Thus, ground-truth for multiple-instrument recordings is crucial for the further development of sophisticated transcription systems.

If musical scores become widely available in digital form (for example via crowd-sourced transcriptions), they provide valuable side-information for signal analysis, and in the extreme cases reduce the transcription task to the alignment of an existing score to the input audio, although it should be noted that different renditions of a song often vary considerably in their instrumentation and arrangement. One such example is the set of syncRWC annotations [1].

Most of the current AMT methods involve a training stage, where the parameters of the method are optimised using manually annotated data. The availability of recorded music with the exact underlying score opens up huge and largely unutilised opportunities for training complex models. In the case of genre- and instrument-specific transcription, separate parameter sets can be trained for different target material.

7. TOWARDS A COMPLETE TRANSCRIPTION

Most of the aforementioned transcription approaches tackle the problems of multiple-F0 estimation and note onset and offset detection. However, in order to fully solve the AMT problem and have a system that provides an output that is equivalent to sheet music, additional issues need to be addressed, such as metre induction, rhythm parsing, key finding, note spelling, dynamics, fingering, expression, articulation and typesetting. Although there are approaches that address many of these individual problems, there exists no 'complete' AMT system to date.

Regarding typesetting, current tools produce readable scores from MIDI data only (e.g. Lilypond [2]), however, cues from the music signal could also assist in incorporating additional information into the final score (e.g. expressive features for note phrasing). As far as dynamics are concerned, in [15] a method was proposed for estimating note intensities in a score-informed scenario. However, estimating note dynamics in an unsupervised way has not been tackled. Another issue would be the fact that most existing ground-truth does not include note intensities, which is difficult to annotate manually, except for datasets created using reproducing pianos (e.g. [38]), which automatically contain intensity information such as MIDI note velocities.

Recent work [3] addresses the problem of automatically extracting the fingering configurations for guitar recordings in an AMT framework. For computing fingering, information from the transcribed signal as well as instrument-specific knowledge is needed. Thus, a robust instrument identification system would need to be incorporated for computing fingerings in multi-instrument recordings.

For extracting expressive features, some work has been done in the past, mostly in the score-informed case. In [18] a framework for extracting expressive features both from a score-informed and an uninformed perspective is proposed. For the latter, an AMT system is used prior to the extraction of expressive features. It should be mentioned though that the extracted features (e.g. auditory loudness, attack) do not necessarily correspond to expressive notation. Thus, additional work needs to be done in order to provide a mapping between mid-level features and actual expressive markings in a transcribed music score.

8. CONCLUSIONS

Automatic music transcription is a rapidly developing research area where several different approaches are still being actively investigated. However from the perspective of evaluation results, the performance seems to converge towards a level that is not satisfactory for all applications.

One viable way of breaking the glass ceiling is to insert more information into the problem. For example, genre- or instrument-specific transcription allows the utilisation of high-level models that are more precise and powerful than their more general counterparts. A promising research direction is to combine several processing principles, or to extract various types of musical information, such as the key, metrical structure, and instrument identities, and feed that into a model that provides context for the note detection process. To enable work in this area, sharing code and data between researchers becomes increasingly important.

Note detection accuracy is not the only determining factor that enables meaningful end-user applications. Often

[1] `http://staff.aist.go.jp/m.goto/RWC-MDB/AIST-Annotation/SyncRWC/`

[2] `http://lilypond.org/`

it is possible to circumvent the limitations of the underlying technology in creative ways. For example in semi-automatic transcription, the problem is redefined as achieving the required transcription accuracy with minimal user effort. It is important to have end-user applications that drive the development of AMT technology and provide it with relevant feedback.

9. REFERENCES

[1] S. A. Abdallah and M. D. Plumbley. Polyphonic transcription by non-negative sparse coding of power spectra. In *ISMIR*, pages 318–325, 2004.

[2] S. Arberet, A. Ozerov, F. Bimbot, and R. Gribonval. A tractable framework for estimating and combining spectral source models for audio source separation. *Signal Processing*, 92(8):1886–1901, 2012.

[3] A.M. Barbancho, A. Klapuri, L.J. Tardon, and I. Barbancho. Automatic transcription of guitar chords and fingering from audio. *IEEE TASLP*, 20(3):915–921, 2012.

[4] J.G.A. Barbedo and G. Tzanetakis. Musical instrument classification using individual partials. *IEEE TASLP*, 19(1):111–122, 2011.

[5] M. Bay, A. F. Ehmann, and J. S. Downie. Evaluation of multiple-F0 estimation and tracking systems. In *ISMIR*, pages 315–320, 2009.

[6] E. Benetos, A. Klapuri, and S. Dixon. Score-informed transcription for automatic piano tutoring. In *EUSIPCO*, 2012.

[7] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in Bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE TASLP*, 18(3):538–549, 2010.

[8] J.C. Brown. Calculation of a constant Q spectral transform. *JASA*, 89(1):425–434, 1991.

[9] J. B. Buckheit and D. L. Donoho. WaveLab and reproducible research. Technical Report 474, Dept of Statistics, Stanford Univ., 1995.

[10] A. Dessein, A. Cont, and G. Lemaitre. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. In *ISMIR*, pages 489–494, 2010.

[11] Z. Duan, B. Pardo, and C. Zhang. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE TASLP*, 18(8):2121–2133, 2010.

[12] J.L. Durrieu and J.P. Thiran. Musical audio source separation based on user-selected F0 track. In *LVA/ICA*, pages 438–445, 2012.

[13] J. Eggink and G.J. Brown. A missing feature approach to instrument identification in polyphonic music. In *ICASSP*, volume 5, pages 553–556, 2003.

[14] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE TASLP*, 18(6):1643–1654, 2010.

[15] S. Ewert and M. Müller. Estimating note intensities in music recordings. In *ICASSP*, pages 385–388, 2011.

[16] S. Ewert and M. Müller. Using score-informed constraints for NMF-based source separation. In *ICASSP*, pages 129–132, 2012.

[17] Y. Freund, R. Schapire, and N. Abe. A short introduction to boosting. *JSAI*, 14(771-780):1612, 1999.

[18] R. Gang, G. Bocko, J. Lundberg, S. Roessner, D. Headlam, and M.F. Bocko. A real-time signal processing framework of musical expressive feature extraction using MATLAB. In *ISMIR*, pages 115–120, 2011.

[19] O. Gillet and G. Richard. Automatic labelling of tabla signals. In *ISMIR*, 2003.

[20] M. Goto. Development of the RWC music database. In *18th Int. Congress Acoustics*, pages 553–556, 2004.

[21] F. Gouyon and S. Dixon. A review of automatic rhythm description systems. *CMJ*, 29(1):34–54, 2005.

[22] G. Grindlay and D. Ellis. Transcribing multi-instrument polyphonic music with hierarchical eigeninstruments. *IEEE JSTSP*, 5(6):1159–1169, 2011.

[23] A. Holzapfel, Y. Stylianou, A.C. Gedik, and B. Bozkurt. Three dimensions of pitched instrument onset detection. *IEEE TASLP*, 18(6):1517–1527, 2010.

[24] X. Huang, A. Acero, and H.-W. Hon, editors. *Spoken Language Processing: A guide to theory, algorithm and system development*. Prentice Hall, 2001.

[25] K. Itoyama, M. Goto, K. Komatani, T. Ogata, and H.G. Okuno. Simultaneous processing of sound source separation and musical instrument identification using Bayesian spectral modeling. In *ICASSP*, pages 3816–3819, 2011.

[26] H. Kirchhoff, S. Dixon, and A. Klapuri. Shift-variant non-negative matrix deconvolution for music transcription. In *ICASSP*, 2012.

[27] A. Klapuri. Multiple fundamental frequency estimation based on harmonicity and spectral smoothness. *IEEE TASLP*, 11(6):804–816, 2003.

[28] A. Klapuri and M. Davy, editors. *Signal Processing Methods for Music Transcription*. Springer, 2006.

[29] A. Loscos, Y. Wang, and W.J.J. Boo. Low level descriptors for automatic violin transcription. In *ISMIR*, pages 164–167, 2006.

[30] M. Marolt. Automatic transcription of bell chiming recordings. *IEEE TASLP*, 20(3):844–853, 2012.

[31] M. Mauch and S. Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE TASLP*, 18(6):1280–1289, 2010.

[32] M. Mauch, K. Noland, and S. Dixon. Using musical structure to enhance automatic chord transcription. In *ISMIR*, pages 231–236, 2009.

[33] Music Information Retrieval Evaluation eXchange (MIREX). http://music-ir.org/mirexwiki/, 2011.

[34] M. Müller, D. Ellis, A. Klapuri, and G. Richard. Signal processing for music analysis. *IEEE JSTSP*, 5(6):1088–1110, 2011.

[35] A. Nesbit, L. Hollenberg, and A. Senyard. Towards automatic transcription of Australian aboriginal music. In *ISMIR*, pages 326–330, 2004.

[36] A. Ozerov, E. Vincent, and F. Bimbot. A general flexible framework for the handling of prior information in audio source separation. *IEEE TASLP*, 20(4):1118–1133, 2012.

[37] P.H. Peeling and S.J. Godsill. Multiple pitch estimation using non-homogeneous Poisson processes. *IEEE JSTSP*, 5(6):1133–1143, 2011.

[38] G. Poliner and D. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP JASP*, 8:154–162, 2007.

[39] G. Poliner, D. Ellis, A. Ehmann, E. Gomez, S. Streich, and B. Ong. Melody transcription from music audio: Approaches and evaluation. *IEEE TASLP*, 15(4):1247–1256, 2007.

[40] G. Reis, N. Fonseca, F. F. de Vega, and A. Ferreira. Hybrid genetic algorithm based on gene fragment competition for polyphonic music transcription. In *Conf. Applications of Evolutionary Computing*, pages 305–314. 2008.

[41] M.P. Ryynänen and A. Klapuri. Polyphonic music transcription using note event modeling. In *WASPAA*, pages 319–322, 2005.

[42] E.D. Scheirer. Using musical knowledge to extract expressive performance information from audio recordings. In H. Okuno and D. Rosenthal, editors, *Readings in Computational Auditory Scene Analysis*. Lawrence Erlbaum, 1997.

[43] P. Smaragdis and J. C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *WASPAA*, pages 177–180, 2003.

[44] Y. Wang and B. Zhang. Application-specific music transcription for tutoring. *IEEE MultiMedia*, 15(3):70–74, 2008.

[45] C. Yeh. *Multiple fundamental frequency estimation of polyphonic recordings*. PhD thesis, Université Paris VI - Pierre et Marie Curie, France, 2008.

PUTTING THE USER IN THE CENTER OF MUSIC INFORMATION RETRIEVAL

Markus Schedl
Department of Computational Perception,
Johannes Kepler University, Linz, Austria

Arthur Flexer
Austrian Research Institute for
Artificial Intelligence, Vienna, Austria

ABSTRACT

Personalized and context-aware music retrieval and recommendation algorithms ideally provide music that perfectly fits the individual listener in each imaginable situation and for each of her information or entertainment need. Although first steps towards such systems have recently been presented at ISMIR and similar venues, this vision is still far away from being a reality. In this paper, we investigate and discuss literature on the topic of user-centric music retrieval and reflect on why the breakthrough in this field has not been achieved yet. Given the different expertises of the authors, we shed light on why this topic is a particularly challenging one, taking a psychological and a computer science view. Whereas the psychological point of view is mainly concerned with proper experimental design, the computer science aspect centers on modeling and machine learning problems. We further present our ideas on aspects vital to consider when elaborating user-aware music retrieval systems, and we also describe promising evaluation methodologies, since accurately evaluating personalized systems is a notably challenging task.

1. WHY CARE ABOUT THE USER?

In our discussion of the importance and the challenges of development and evaluation in Music Information Retrieval (MIR) we distinguish between systems-based and user-centric MIR. We define systems-based MIR as all research concerned with experiments existing solely in a computer, e.g. evaluation of algorithms on digital databases. In contrast, user-centered MIR always involves human subjects and their interaction with MIR systems.

Systems-based MIR has traditionally focused on computational models to describe universal aspects of human music perception, for instance, via elaborating musical feature extractors or similarity measures. Doing so, the existence of an objective "ground truth" is assumed, against which corresponding music retrieval algorithms (e.g., playlist generators or music recommendation systems) are evaluated. To give a common example, music retrieval approaches have been evaluated via genre classification experiments for years. Although it was shown already in 2003 that musical genre is an ill-defined concept [1], genre information still serves as a proxy to assess music similarity and retrieval approaches in systems-based MIR.

On the way towards user-centered MIR, the coarse and ambiguous concept of genre should either be treated in a personalized way or replaced by the concept of similarity. When humans are asked to judge the similarity between two pieces of music, however, certain other challenges need to be faced. Common evaluation strategies typically do not take into account the musical expertise and taste of the users. A clear definition of "similarity" is often missing too. It might hence easily occur that two users apply a very different, individual notion of similarity when assessing the output of music retrieval systems. While a first person may experience two songs as rather dissimilar due to very different lyrics, a second one may feel a much higher resemblance of the very same songs because of a similar instrumentation. Similarly, a fan of Heavy Metal music might perceive a Viking Metal track as dissimilar to a Death Metal piece, while for the majority of people the two will sound alike.

The above examples illustrate that there are many aspects that influence what a human perceives as similar in a musical context. These aspects can be grouped into three different categories according to [29]: *music content*, *music context*, and *user context*. Examples for each category are given in Figure 1. It is exactly this multifaceted and individual way of music perception that has largely been neglected so far when elaborating and evaluating music retrieval approaches, but should be given more attention, in particular considering the trend towards personalized and context-aware systems.

A *personalized system* is one that incorporates information about the user into its data processing part (e.g., a particular user taste for a movie genre). A *context-aware system*, in contrast, takes into account dynamic aspects of the user context when processing the data (e.g., location and time where/when a user issues a query). Although the border between personalization and context-awareness may appear fuzzy from this definition, in summary, personalization usually refers to the incorporation of more static, general user preferences, whereas context-awareness refers to the fact that frequently changing aspects of the user's environmental, psychological, and physiological context are considered.

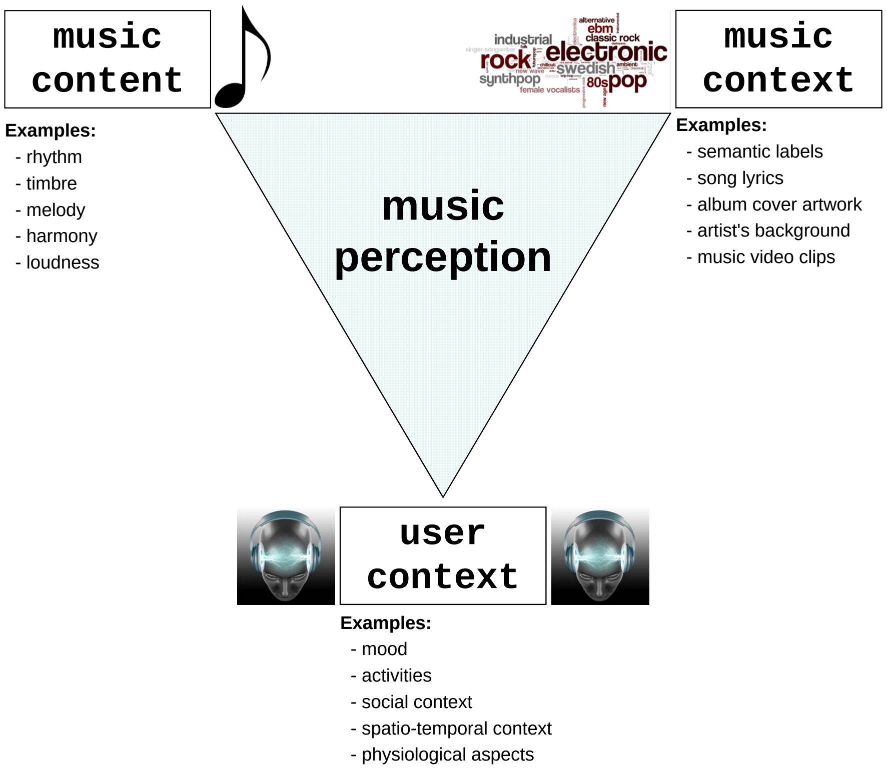


Figure 1. Factors that influence human music perception.

The remainder of this paper is organized as follows. Section 2 reviews approaches that, in one way or the other, take the user into account when building music retrieval systems. Evaluation strategies for investigating user-centric MIR are discussed in Section 3. In Section 4, we eventually summarize important factors when creating and evaluating user-aware music retrieval systems.

2. HOW TO MODEL THE USER?

Existing user-aware systems typically model the user in a very simplistic way. For instance, it is common in *collaborative filtering* approaches [22, 28] to build user profiles only from information about a user u expressing an interest in item i. As an indicator of interest may serve, for example, a click on a particular item, a purchasing transaction, or in MIR the act of listening to a certain music piece. Such indications, in their simplest form, are stored in a binary matrix where element $r(u, i)$ denotes the presence or absence of a connection between user u and item i. In common recommendation systems, a more fine-grained scale for modeling the user interest in an item is typically employed – users frequently rate items according to a Likert-type scale, e.g., by assigning one to five stars to it. Matrix factorization techniques are subsequently applied to recommend novel items [19].

Taking a closer look at literature about context-aware retrieval and recommendation in the music domain, we can see that approaches differ considerably in terms of how the user context is defined, gathered, and incorporated. The majority of approaches rely solely on one or a few aspects (temporal features in [7], listening history and weather conditions in [21], for instance), whereas comprehensive user models are rare in MIR. One of the few exceptions is Cunningham et al.'s study [8] that investigates if and how various factors relate to music taste (e.g., human movement, emotional status, and external factors such as temperature and lightning conditions). Based on the findings, the authors present a fuzzy logic model to create playlists.

There further exists some work that assumes a mobile music consumption scenario. The corresponding systems frequently aim at matching music with the current pace of a walker or jogger, e.g., [3, 24]. Such systems typically try to match the user's heartbeat with the music played [23]. However, almost all proposed systems require additional hardware for context logging, e.g., [8, 9, 11].

In [15] a system that matches tags describing a particular place with tags describing music is presented. Employing text-based similarity measures between the multimodal sets of tags, Kaminskas and Ricci propose their system for location-based music recommendation. Baltrunas et al. [2] suggest a context-aware music recommender system for music consumption while driving. Although the authors take into account eight different contextual factors (e.g., driving style, mood, road type, weather, traffic conditions), their application scenario is quite restricted and their system relies on explicit human feedback, which is burdensome.

Zhang et al. present *CompositeMap* [34], a model that takes into account similarity aspects derived from music content as well as social factors. The authors propose a multimodal music similarity measure and show its applicability to the task of music retrieval. They also allow a simple kind of personalization of this model by letting the user weight the individual music dimensions on which similarity is estimated. However, they do neither take the user context into consideration, nor do they try to learn a user's preferences.

In [26] Pohle et al. present preliminary steps towards a simple personalized music retrieval system. Based on a clustering of community-based tags extracted from *last.fm*, a small number of musical concepts are derived using *Non-Negative Matrix Factorization* (NMF) [20,32]. Each music artist or band is then described by a "concept vector". A user interface allows for adjusting the weights of the individual concepts, based on which artists that match the resulting distribution of the concepts best are recommended to the user. Zhang et al. propose in [34] a very similar kind of personalization strategy via user-adjusted weights.

Knees and Widmer present in [17] an approach that incorporates *relevance feedback* [27] into a text-based music search engine [16] to adapt the retrieval process to user preferences. The search engine proposed by Knees et al. builds a model from music content features (MFCCs) and music context features (term vector representations of artist-related Web pages). To this end, a weight is computed for each (term, music item)-pair, based on the term vectors. These weights are then smoothed, taking into account the closest neighbors according to the content-based similarity measure (Kullback-Leibler divergence on Gaussian Mixture Models of the MFCCs). To retrieve music via natural language queries, each textual query issued to the system is expanded via a *Google* search, resulting again in a term weight vector. This query vector is subsequently compared to the smoothed weight vectors describing the music pieces, and those with smallest distance to the query vector are returned.

Nürnberger and Detyniecki present in [25] a variant of the *Self-Organizing Map* (SOM) [18] that is based on a model that adapts to *user feedback*. To this end, the user can move data items on the SOM. This information is fed back into the SOM's codebook, and the mapping is adapted accordingly.

In [33] Xue et al. present a *collaborative personalized search model* that alleviates the problems of *data sparseness* and *cold-start for new users* by combining information on different levels (individuals, interest groups, and global). Although not explicitly targeted at music retrieval, the idea of integrating data about the user, his peer group, and global data to build a social retrieval model might be worth considering for MIR purposes.

The problem with the vast majority of approaches presented so far is that evaluation is still carried out without sufficient user involvement. For instance, [7, 25, 26] seemingly do not perform any kind of evaluation involving real users, or at least do not report it. Some approaches are evaluated on user-generated data, but do not request feedback from real users during the evaluation experiments. For example, [16] makes use of collaborative tags stored in a database to evaluate the proposed music search engine. Similarly, [21] relies on data sets of listening histories and weather conditions, and [33] uses a corpus of Web search data. Even if real users are questioned during evaluation, their individual properties (such as taste, expertise, or familiarity with the music items under investigation) are regularly neglected in evaluation experiments. In these cases, evaluation is typically performed to answer a very narrow question in a restricted setting. To give an example, the work on automatically selecting music while doing sports, e.g. [3, 23, 24], is evaluated on the very question of whether pace or heartbeat of the user does synchronize with the tempo of the music. Likewise Kaminskas and Ricci's work on matching music with places of interest [15], even though it is evaluated by involving real users, comprises only the single question whether the music suggested by their algorithm is suited for particular places of interest. Different dimensions of the relation between images and music are not addressed. Although this is perfectly fine for the intended use cases, such highly specific evaluation settings are not able to provide answers to more general questions of music retrieval and recommendation, foremost because these settings fail at offering explanations for the (un)suitability of the musical items under investigation.
An evaluation approach that tries to alleviate this shortcoming is presented in [4], where subjective listening tests to assess music recommendation algorithms are conducted using a multifaceted questionnaire. Besides investigating the enjoyment a user feels when listening to the recommended track ("liking"), the authors also ask for the user's "listening intention", whether or not the user knows artist and song ("familiarity"), and whether he or she would like to request more similar music ("give-me-more"). A similar evaluation scheme is suggested in [12]. However, Firan et al. only investigate liking and novelty.
In summary, almost all approaches reported are still more systems-based than user-centric.

3. HOW TO EVALUATE USER-CENTERED MIR?

In what follows we will argue that whereas evaluation of systems-based MIR has quite matured, evaluation of user-centered MIR is still in its infancy. Let us start by reviewing what the nature of experiments is in the context of MIR. The basic structure of MIR experiments is the same as in any other experimental situation: the question is whether there are effects of the variation of the independent variables (also called factors) on the dependent variables. In the case of systems-based MIR, independent variables are e.g. type and certain parameter characteristics of the algorithms used or type and characteristics of the data set in question. Typical dependent variables are various performance measures like accuracy, precision, root mean squared error or training time. A standard computer experiment is genre classification where the independent variable is the type of classification algorithm, say algorithm A and B, and the dependent variable is the achieved accuracy. Statistical testing is used to ensure that the observed effects on the dependent variables are caused by the varied independent variables and not by mere chance, i.e. to ascertain that the observed differences are too large to attribute them to random influences only. Besides using the proper statistical instruments to establish statistical significance of results it is equally important to make sure to control all important factors in the experimental design. Any factor that is able to influence the dependent variables has to be part of the experimental design. E.g. if algorithm A, compared to algorithm B, works better for electronic dance music than for rock music then any experimental design not containing dance music will obscure differences between A and B. The important thing to note is that for systems-based MIR which uses only computer experiments it is comparably easy to control all important factors which could have an influence on the dependent variables. This is because the number of factors is both manageable and controllable since the experiments are being conducted on computers and not in the real world.

Already early on in the history of MIR research, gaps concerning the evaluation of MIR systems have been identified. Futrelle and Downie [14], in their review of the first three years of the ISMIR conference published in 2003, identify two major problems: (i) no commonly accepted means of comparing retrieval techniques, (ii) few if any attempts to study potential users of MIR systems. The first problem concerns evaluation of computer experiments and the second problem the barely existing inclusion of users in MIR studies. Flexer [13], in his review of the 2004 ISMIR conference [5], argues for the necessity of statistical evaluation of MIR experiments. He presents minimum requirements concerning statistical evaluation by applying fundamental notions of statistical hypotheses testing to MIR research. His discussion is concerned with systems-based MIR, the example used throughout the paper is that of automatic genre classification based on audio content analysis. The MIR community is criticized for the lack of statistical evaluation it uses, e.g. only two papers in the ISMIR 2004 proceedings [5] employed a statistical test to prove significance of their results. These ongoing discussions about evaluation of MIR experiments have led to a first evaluation benchmark taking place at the ISMIR conference 2004 [6] and further on to the establishment of the annual evaluation campaign for MIR algorithms (Music Information Retrieval Evaluation eXchange, MIREX) [10]. In 2011, MIREX consisted of 16 tasks ranging from audio classification, cover song identification, audio key detection to structural segmentation and audio tempo estimation. All but two tasks are concerned with systems-based MIR and a purely computer-based evaluation of algorithms. The two exceptions using human evaluations in a more real-world setting are *Audio Music Similarity and Retrieval* and *Symbolic Melodic Similarity*. Starting with the MIREX 2006 evaluation [10] statistical tests are being used to analyze results.

The situation concerning evaluation of user-centric MIR research is far less well developed. In a recent comprehensive review [31] of user studies in the MIR literature by Weigl and Guastavino, papers from the first decade of ISMIR conferences and related MIR publications were analyzed. A central result is that MIR research has a mostly systems-centric focus. Only twenty papers fell under the broad category of "user studies" which is an alarmingly small number given that 719 articles have been published in the ISMIR conference series alone. To make things worse, these user studies are "predominantly qualitative in nature" and of "largely exploratory nature" [31]. The explored topics range from e.g. user requirements and information needs, insights into social and demographic factors to user-generated meta-information and ground truth. This all points to the conclusion that evaluation of user-centered MIR is at its beginning and that especially a more rigorous quantitative treatment is still missing.

In discussing the challenges of quantitative evaluation of user-centered MIR we like to turn to an illustrative example: the recent 2011 *Audio Music Similarity and Retrieval* task [1] within the annual MIREX [10] evaluation campaign. Each of 18 competing algorithms was given 7000 songs (30 second audio clips) for which they computed similarity rankings. The data consisted of 10 equally sized genre classes ranging from classic music to rock to hip-hop. From the 7000 songs, "100 songs were randomly selected from the 10 genre groups (10 per genre) as queries and the first 5 most highly ranked songs out of the 7000 were extracted for each query (after filtering out the query itself, returned results from the same artist were also omitted). Then, for each query, the returned results (candidates) from all participating algorithms were grouped and were evaluated by human graders" [1]. For each individual query/candidate pair, a single human grader provided both a FINE score (from 0 (failure) to 100 (perfection)) and a BROAD score (not similar NS, somewhat similar SS, very similar VS) indicating how similar the songs are in their opinion. The independent variable here is the type of algorithm used to compute the similarity rankings. The dependent variables are the subjects' broad and fine appraisal of the perceived similarity. But since this is a real-world experiment involving human subjects there is a whole range of factors that have not been assessed. E.g. there are social and demographic factors that might clearly influence the user's judgment of music similarity: their age, gender, cultural background and especially their musical history, experience and knowledge. But also factors concerning their momentary situation during the actual listening experiment might have an influence: time of day, mood, physical condition. Not to forget more straightforward variables like type of speakers or headphones used for the test. As al-

[1] The 2011 results and details can be found at: `http://www.music-ir.org/mirex/wiki/2011:Audio_Music_Similarity_and_Retrieval_Results`

ready mentioned in section 1, even the choice of dependent variable is debatable. After all, what does "similar" really mean in the context of music? Timbre, mood, harmony, melody, tempo, etc might all be valid answers for different people. This points to a certain lack of rigor concerning the instruction of subjects during the experiment. This enumeration of potential problems is not intended to badmouth this MIREX task which still is a valuable contribution and an applaudable exception to the rule of computer-only evaluation. But it is meant as a warning and to highlight the explosion of independent variables and factors that might add to the variance of observed results and might obscure significant differences. In principle, all such factors have to be recorded and made independent variables in the overall experimental design.

If MIR is to succeed in maturing from purely systems-based to user-centered research we will have to leave the nice and clean world of our computers and face the often bewilderingly complex real world of real human users and all the challenges this entails for proper design and evaluation of experiments. To make this happen it will be necessary that our community with a predominantly engineering background opens up to the so-called "soft sciences" of e.g. psychology and sociology which have developed instruments and methods to deal with the complexity of human subjects.

4. DISCUSSION AND CONCLUSIONS

Incorporating real users in both the development and assessment of music retrieval systems is of course an expensive and arduous task. However, recent trends in music distribution, in particular the emergence of music streaming services that make available millions of tracks to their users, call for intelligent personalized and context-aware systems to deal with this abundance. Concerning the development of such systems, we believe that the following two reasons have prevented major breakthroughs so far: (i) a general lack of research on user-centered systems, (ii) a lack of awareness concerning the complexity of evaluation of user-centered systems. In designing such systems, the user should already be taken into account at an early stage during the development process. We need to better understand what the user's individual requirements are and address these requirements in our implementations. Otherwise it is unlikely that even the spiffiest personalized systems will succeed (without frustrating the user). We hence identify the following four key requirements for elaborating user-centric music retrieval systems:

Personalization aspects have to be taken into account. In this context, it is important to note the highly subjective, cognitive component in the understanding of music and judgement of its personal appeal. Therefore, designing user-aware music applications requires intelligent machine learning techniques, in particular, preference learning approaches that relate the user context to concise, situation-dependent music preferences.

User models that encompass different social scopes are needed. They may aggregate an individual model, an interest group model, a cultural model, and a global model. Furthermore, the user should be modeled as comprehensively as possible, in a fine-grained and multifaceted manner. With today's sensor-packed smartphones and other intelligent devices it has become easy to perform extensive context logging. Of course, privacy issues must also be taken seriously.

Multifaceted similarity measures that combine different feature categories (music content, music context, and user context) are required. The corresponding representation models should then not only allow to derive similarity between music via content-related aspects, such as beat strength or instruments playing, or via music context-related properties, such as the geographic origin of the performer or a song's lyrics, but also to describe users and user groups in order to compute a listener-based similarity score.

Evaluation of user-centric music retrieval approaches has to include all independent variables that are able to influence the dependent variables into the experimental design. In particular, such factors may well relate to individual properties of the human assessors. Furthermore, it is advisable to make use of recent approaches that minimize the amount of labor required by the human assessors, while at the same time maintaining the significance of the experiments. This can be achieved, for instance, by employing the concept of "Minimal Test Collections" in the evaluation of music retrieval systems [30].

By paying attention to these advices, we are sure that the exciting field of user-centric music information retrieval will continue to grow and eventually provide us with algorithms and systems that offer personalized and context-aware access to music in an unintrusive way.

5. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Fund (FWF): P22856-N23 and P24095 as well as by the European Commission, FP7 (Seventh Framework Programme), ICT-2011.1.5 Networked Media and Search Systems, grant agreement no. 287711.

6. REFERENCES

[1] J.-J. Aucouturier and F. Pachet. Representing Musical Genre: A State of the Art. *Journal of New Music Research*, 32(1):83–93, 2003.

[2] L. Baltrunas, M. Kaminskas, B. Ludwig, O. Moling, F. Ricci, K.-H. Lüke, and R. Schwaiger. InCarMusic: Context-Aware Music Recommendations in a Car. In *Proc. EC-Web*, 2011.

[3] J. T. Biehl, P. D. Adamczyk, and B. P. Bailey. DJogger: A Mobile Dynamic Music Device. In *CHI 2006: Extended Abstracts*, 2006.

[4] D. Bogdanov and P. Herrera. How Much Metadata Do We Need in Music Recommendation? A Subjective Evaluation Using Preference Sets. In *Proc. ISMIR*, 2011.

[5] C.L. Buyoli and R. Loureiro. *Fifth International Conference on Music Information Retrieval*. Universitat Pompeu Fabra, 2004.

[6] P. Cano, E. Gómez, F. Gouyon, P. Herrera, M. Koppenberger, B. Ong, X. Serra, S. Streich, and N. Wack. ISMIR 2004 Audio Description Contest. 2006.

[7] T. Cebrián, M. Planagumà, P. Villegas, and X. Amatriain. Music Recommendations with Temporal Context Awareness. In *Proc. RecSys*, 2010.

[8] S. Cunningham, S. Caulder, and V. Grout. Saturday Night or Fever? Context-Aware Music Playlists. In *Proc. Audio Mostly*, 2008.

[9] S. Dornbush, J. English, T. Oates, Z. Segall, and A. Joshi. XPod: A Human Activity Aware Learning Mobile Music Player. In *Proc. Workshop on Ambient Intelligence, IJCAI*, 2007.

[10] J. S. Downie. The Music Information Retrieval Evaluation eXchange (MIREX). *D-Lib Magazine*, Dec 2006.

[11] G. T. Elliott and B. Tomlinson. PersonalSoundtrack: Context-aware Playlists that Adapt to User Pace. In *CHI 2006: Extended Abstracts*, 2006.

[12] Claudiu S. Firan, Wolfgang Nejdl, and Raluca Paiu. The Benefit of Using Tag-Based Profiles. In *Proceedings of the 5th Latin American Web Congress (LA-WEB)*, pages 32–41, Santiago de Chile, Chile, October–November 2007.

[13] A. Flexer. Statistical Evaluation of Music Information Retrieval Experiments. *Journal of New Music Research*, 35(2):113–120, June 2006.

[14] J. Futrelle and J. S. Downie. Interdisciplinary Research Issues in Music Information Retrieval: ISMIR 2000-2002. *Journal of New Music Research*, 32(2):121–131, 2003.

[15] M. Kaminskas and F. Ricci. Location-Adapted Music Recommendation Using Tags. In *Proc. UMAP*, 2011.

[16] P. Knees, T. Pohle, M. Schedl, and G. Widmer. A Music Search Engine Built upon Audio-based and Web-based Similarity Measures. In *Proc. SIGIR*, 2007.

[17] P. Knees and G. Widmer. Searching for Music Using Natural Language Queries and Relevance Feedback. In *Proc. AMR*, 2007.

[18] T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Germany, 3rd edition, 2001.

[19] Y. Koren, R. Bell, and C. Volinsky. Matrix Factorization Techniques for Recommender Systems. *Computer*, 42, Aug 2009.

[20] D. D. Lee and H. S. Seung. Learning the Parts of Objects by Non-negative Matrix Factorization. *Nature*, 401(6755):788–791, 1999.

[21] J. S. Lee and J. C. Lee. Context Awareness by Case-Based Reasoning in a Music Recommendation System. In *Proc. UCS*, 2007.

[22] G. Linden, B. Smith, and J. York. Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, 4(1), 2003.

[23] H. Liu, J. Hu, and M. Rauterberg. Music Playlist Recommendation Based on User Heartbeat and Music Preference. In *Proc. ICCTD*, 2009.

[24] B. Moens, L. van Noorden, and M. Leman. D-Jogger: Syncing Music with Walking. In *Proc. SMC*, 2010.

[25] A. Nürnberger and M. Detyniecki. Weighted Self-Organizing Maps: Incorporating User Feedback. In *Proc. ICANN/ICONIP*, 2003.

[26] T. Pohle, P. Knees, M. Schedl, and G. Widmer. Building an Interactive Next-Generation Artist Recommender Based on Automatically Derived High-Level Concepts. In *Proc. CBMI*, 2007.

[27] J. J. Rocchio. Relevance Feedback in Information Retrieval. In Gerard Salton, editor, *The SMART Retrieval System - Experiments in Automatic Document Processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[28] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based Collaborative Filtering Recommendation Algorithms. In *Proc. WWW*, 2001.

[29] M. Schedl and P. Knees. Personalization in Multimodal Music Retrieval. In *Proc. AMR*, 2011.

[30] J. Urbano and M. Schedl. Towards Minimal Test Collections for Evaluation of Audio Music Similarity and Retrieval. In *Proc. AdMIRe*, 2012.

[31] D. Weigl and C. Guastavino. User Studies in the Music Information Retrieval Literature. In *Proc. ISMIR*, 2011.

[32] W. Xu, X. Liu, and Y. Gong. Document Clustering Based on Non-negative Matrix Factorization. In *Proc. SIGIR*, 2003.

[33] G.-R. Xue, J. Han, Y. Yu, and Q. Yang. User Language Model for Collaborative Personalized Search. *ACM Transactions on Information Systems*, 27(2), Feb 2009.

[34] B. Zhang, J. Shen, Q. Xiang, and Y. Wang. CompositeMap: A Novel Framework for Music Similarity Measure. In *Proc. SIGIR*, 2009.

THE IMPACT (OR NON-IMPACT) OF USER STUDIES IN MUSIC INFORMATION RETRIEVAL

Jin Ha Lee
The Information School
University of Washington
jinhalee@uw.edu

Sally Jo Cunningham
Department of Computer Science
University of Waikato
sallyjo@waikato.ac.nz

ABSTRACT

Most Music Information Retrieval (MIR) researchers will agree that understanding users' needs and behaviors is critical for developing a good MIR system. The number of user studies in the MIR domain has been gradually increasing since the early 2000s reflecting the need for empirical studies of users. However, despite the growing number of user studies and the wide recognition of their importance, it is unclear how large their impact has been in the field; on how systems are developed, evaluation tasks are created, and how we understand critical concepts such as music similarity or music mood. In this paper, we present our analysis on the growth, publication and citation patterns, and design of 155 user studies. This is followed by a discussion of a number of issues/challenges in conducting MIR user studies and distributing the research results. We conclude by making recommendations to increase the visibility and impact of user studies in the field.

1. INTRODUCTION

Understanding users is a fundamental step in developing successful Music Information Retrieval (MIR) systems and services. Most MIR researchers will agree with this idea, and furthermore, it is not uncommon to hear various speakers at MIR related conferences specifically arguing for the importance of user studies, academically as well as commercially. Despite the growing number of user studies and the wide recognition of their importance in the MIR domain, it is unclear as to what impact these studies have really made. Have these studies in fact changed how MIR systems are developed or evaluation tasks are designed? Have they really changed how we understand critical concepts such as music similarity or music mood? For MIR researchers specializing in user studies to move forward in this domain, it is necessary to understand our past: what have we been doing and what kind of impact have we made or not? In order to lay the foundation for this discussion, we collected 155 user studies related to music, reviewed the content, and analyzed the publication and citation patterns, and research design of these studies.

2. STUDY DESIGN

2.1 Definition of "User Studies"

Our first challenge was to define and set the boundaries for "user studies." From our analysis of relevant literature, we found two major categories of user studies: "studies of users" (e.g., music information needs) and "studies involving users" (e.g., usability testing). Weigl and Guastavino [7], in their recent review article of user studies in MIR literature, defined user studies as "documents report(ing) 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 (p. 335)." In this study, we adopt a broader definition of "user studies" as studies reporting on 1) empirical investigation of needs, behaviors, perceptions, and opinions of humans, 2) experiments and usability testing involving humans, 3) analysis of user-generated data, or 4) review of the studies above. This is because a broader definition will allow for a comparison of these different types of user studies and enable us to see patterns of concentration with regards to particular types of user studies related to MIR.

2.2 Data Collection

We conducted an extensive literature search in multiple domains related to music (e.g., MIR, Library and Information Science (LIS), Human Computer Interaction (HCI), Computer Science (CS), Engineering, Psychology, Musicology) to identify these studies. We conducted searches in multiple databases including WorldCat, EBSCO, Web of Knowledge, IEEE Xplore, ACM DL, InfoPsych, and Google Scholar. We used the different combinations of the following search terms: music, user, human, people, need, use, behavior, testing, involvement, learning, interaction, design, accessibility, usability, user-centered, etc. After retrieving the relevant studies, we also followed the citations in order to broaden our search. In total, we found 155 studies related to music users.

3. PUBLICATION PATTERNS OF USER STUDIES

3.1 Growth of the Publications

First, we analyzed several aspects related to the publication patterns of the user studies. We examined the publications dates of the user studies in order to learn more about the growth pattern. Figure 1 shows the distribution of the number of user studies published by year. We can observe the steady increase in the number of publications over the years. There were a small number of user studies pre-dating 2000, but the substantial growth started in early 2000s when the need for empirical user studies was pointed out in works such as [1], [2], and [3]. There was also a noticeable increase in 2009 and we expect that this growth pattern will continue for the coming years, at least in the near future. Although this growth pattern is encouraging, when compared with the number of studies focusing on the system aspect of MIR, the overall number of user studies is still relatively small [7].

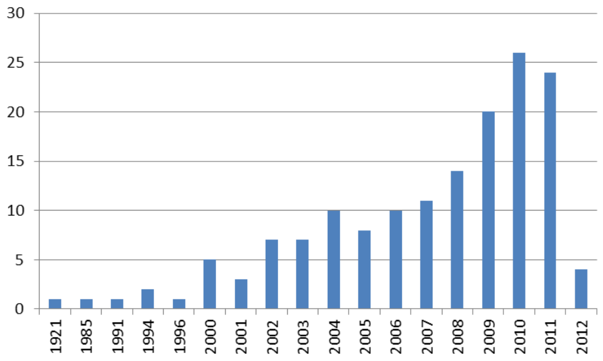


Figure 1. Distribution of the number of user studies by the year of publication

3.2 Types of Publications

We also examined the publication venues of these studies. Of the 155 studies, there were 91 conference publications, 56 journal articles, 6 workshop papers, 1 book chapter, and 1 white paper. There were a total of 83 different venues where music user studies appeared. The primary source of user studies was the ISMIR conference proceedings with 41 user studies, and all the other journals and conference proceedings included 5 or fewer user studies. 65 of the 155 user studies (42%) were the only music user study published in the particular venue. This pattern of concentration in a small number of core publications can be explained by Bradford's law which characterizes the pattern of diminishing returns in searching for references in scholarly publications [1]. The concentration of MIR user studies in the ISMIR proceedings is perhaps stronger than Bradford's predicted 1:n:n^2 ratio of journals (where each proportion contains approximately the same number of articles). The top sources in the order of number of relevant papers are: ISMIR (41); ACM Conference on Human Factors in Computing Systems (5); ACM International Conference on Multimedia (4), ACM/IEEE-CS Joint Conference on Digital Libraries (4), International Conference on Information Visualization (4); International Conference on Mobile and Ubiquitous Multimedia (4), Journal of New Music Research (3), Music Perception (3), Psychology of Music (3), IEEE International Conference on Multimedia and Expo (3), etc.

The skewed distribution of publications poses a challenge for researchers of user studies as well as readers who are interested in finding these studies. We confirmed that it is in fact impossible to find all these studies using a single database or search engine. Also many researchers tend to conduct their literature search in their own domain, which will exclude many relevant works published in other domains (e.g., psychology scholars not citing MIR literature in CS domain). Although the ISMIR proceedings are freely available on the Web, a large number of other publications are fee-based. Unless the researchers' or readers' institutions have subscriptions to these different publications, it will be difficult and expensive to access these works. This also raises a question about distributing our knowledge to the general public who are simply interested in music and also people who are in music industry. Much of the MIR research aims to not only contribute to improving the general knowledge of music and how people interact with music, but also to create better systems and services related to music. If there is a barrier for general public and people outside of academia to access these works, then without a doubt, the impact we can make in the field will also be diminished.

3.3 Co-authorship Analysis

We performed a co-authorship analysis to further understand the patterns of publication. Figure 2 shows the co-authorship graph generated by using NodeXL, a tool for visualization and exploration of networks [6]. The graph's vertices were grouped based on the Clauset-Newman-Moore cluster algorithm and the graph was laid out using the Harel-Koren Fast Multiscale layout algorithm. The nodes represent the authors and the line connecting the nodes represents the co-authorship between the two authors. The size of the node is scaled based on the number of publications by a particular author, and the width of the line connecting two nodes is scaled based on the number of times the pair of authors have co-authored a user study.

A few strong networks emerged. The most notable network is grouped around Sally Jo Cunningham, J. Stephen Downie, Jin Ha Lee, David Bainbridge and 20 other scholars. The two networks formed around Jukka Holm and Arto Lehtiniemi, and Charlie Inskip, Andrew MacFarlane, and Pauline Rafferty are also very prominent. These strong networks seem to be forming based on the particular lab/university and regions: University of Illinois at Urbana-Champaign and University of Waikato for the first group, Finland for the second group, and UK for the third group. Another notable network formed around Adrian C. North and David Hargreaves in UK represents many user studies published in psychology. Another aspect to note is that the network is very disconnected, with a large number of small components, each consisting of a small number of authors. Part of the reason for this pattern could be because MIR is still a relatively new field, and there have not been many opportunities for cross-institutional ties to be formed. Or, it may reflect the widespread appeal of music as a subject for research (which is corroborated by the number and diversity of publication venues surveyed for this study). Further analysis will be necessary to determine the reasons for seeing this kind of co-authorship patterns.

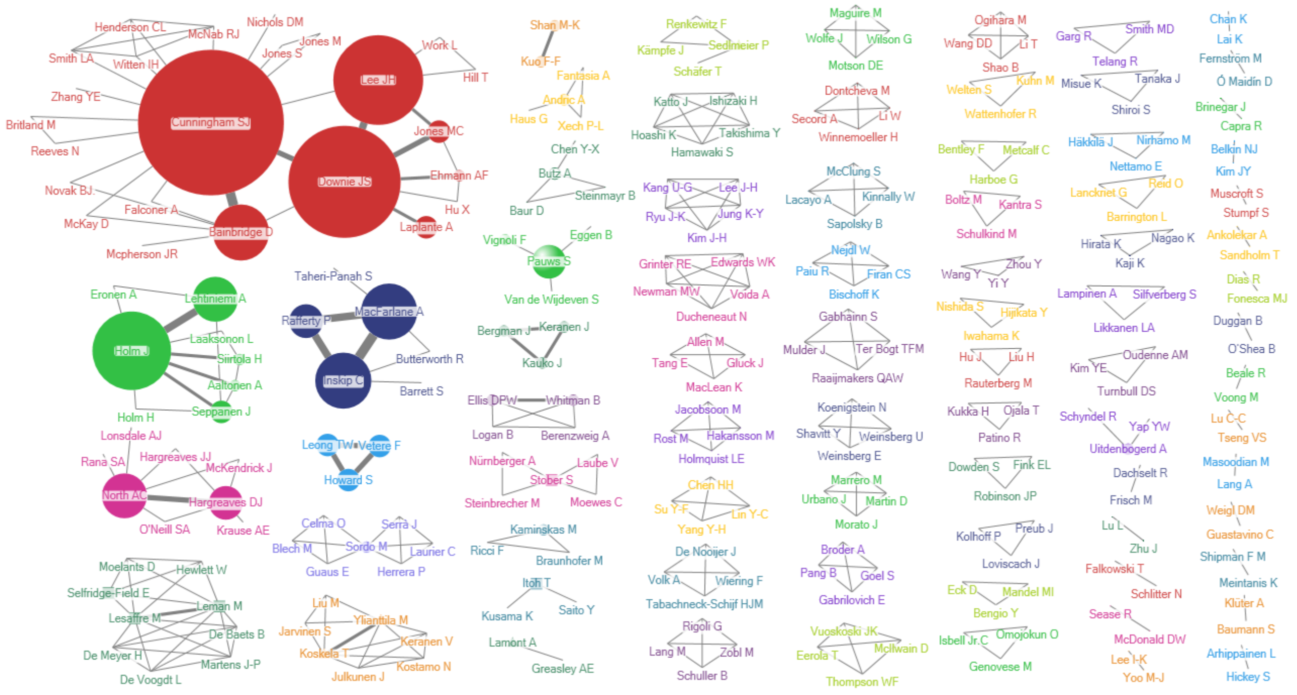


Figure 2. Co-authorship network among the authors

4. CITATION PATTERNS OF USER STUDIES

As part of the effort in understanding the impact of these studies, we investigated how often they were cited as of April 24, 2012 using the citation data from Google Scholar (GS). The reason for using GS is because the major publications in the field such as ISMIR conference proceedings are not indexed in other major databases such as EBSCO, Web of Science, etc. Also since we are interested in the scholarly as well as commercial impact of the user studies, being able to search for patents in addition to scholarly work on GS was deemed useful. We found a total of 3097 citations of 154 user studies in research publications and patents (one study did not show up). Figure 3 shows the distribution of the citation counts of the user studies in other materials. The X-axis represents the number of citations and the Y-axis represents the number of user studies that had the specified range of citation counts. The average number was 20.1 with the standard deviation of 44.5, the median of 5.5, and the maximum of 348.

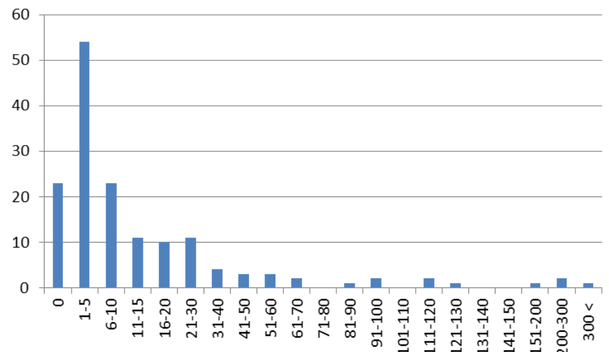


Figure 3. Distribution of the number of references of the user studies in other articles and patents

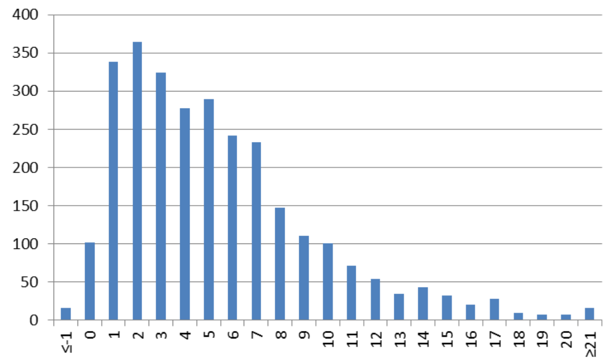


Figure 4. Distribution of the number of years for user studies to get cited

We were also interested in how long it takes for user studies to get cited. Figure 4 shows the distribution of the number of years it took for the user studies to get cited. This is based on the publication dates of the 2864 out of the 3097 citing articles and patents we were able to retrieve on GS. The X-axis represents the number of years passed after the publication of user studies and the Y-axis represents the number of citing articles/patents. The negative numbers (-1,-2) represent the cases where the author was self-citing a study that was yet to be published, or citing a study that was made available online before the print publication. The mean number of years was 5.48 with the standard deviation of 0.10, median of 5, and maximum of 90. About 40% (1144 out of 2864) were cited in 3 years or less after the user study was published and about 60% (1710) in 5 years or less. The citation pattern gradually decreases, and only about 15% (422) were cited after 10 years or more, and about 4% (119) after 15 years or more. The citation pattern suggests that the "perceived" relevance of the results quickly diminishes over time. Since the majority of the user studies were published after 2000, for a more complete picture, this analysis will have to be replicated in 10 or 20 years.

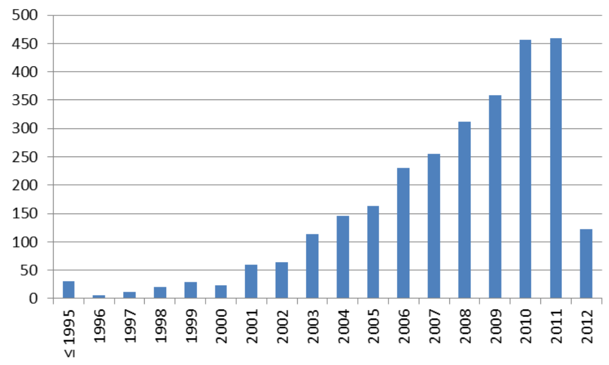


Figure 5. Distribution of the citing articles/patents by the publication year of citing articles/patents

Figure 5 shows the distribution of the citing articles/ patents by their publication dates. Overall the numbers of citing articles/patents are showing a pattern of steady increase. Figure 3, 4, and 5, altogether seem to suggest that the user studies are in fact making growing impact to the field, although the impact of the studies tend to quickly diminish over time based on citation patterns.

Author/Year	Title	Ref
McNab et al./96	Towards the digital music library: tune retrieval from acoustic input	348
Berenzweig et al./04	A large-scale evaluation of acoustic and subjective music-similarity measures	230
North et al./00	The importance of music to adolescents	224
Levitin, D. J./94	Absolute memory for musical pitch: evidence from the production of learned melodies	184
Voida et al./05	Listening in: practices surrounding iTunes music sharing	121
Ellis & Whitman/02	The quest for ground truth in musical artist similarity	111
North et al./04	Uses of music in everyday life	111
Boltz et al./ 91	Effects of background music on the remembering of filmed event	100
Pauws & Eggen/03	Realization and user evaluation of an automatic playlist generator	100
Lee & Downie/04	Survey of music information needs, uses, and seeking behaviours: preliminary findings	82

Table 1. The top 10 most cited user studies

Table 1 presents the top 10 most cited user studies in the field. There is a mix of user experiments, evaluation of particular systems, studies of information behaviors and user-generated data, etc. The most heavily cited user study was by McNab et al. In this study, 10 users were asked to sing 10 songs from memory which were taped for analysis of key, pitch, contour, etc. The article was published in Proceedings of the First ACM International Conference on Digital Libraries and was cited widely in various papers on content-based music retrieval systems and measures. We believe that the heavy citation of this paper and also Levitin was at least partly due to the fact that they were early papers which dealt with content-based MIR, a topic which has dominated MIR research for the past decade. Studies by Berenzweig et al. and Ellis & Whitman explore measures for generating ground truth based on user data which is strongly relevant to the evaluation of algorithms, another big accomplishment of the past decade (i.e., MIREX). Studies of more general user needs and behaviors (North et al., Lee & Downie) may have had a broader impact to multiple areas related to music. The popularity of particular music application (Voida), association of music and other multimedia (Boltz et al.), and particular organizational measures (Pauws & Eggen) also seem to affect the heavy citation patterns.

5. RESEARCH DESIGN OF USER STUDIES

Lastly, we examined the studies more deeply in order to learn more about the research design of these user studies. We analyzed the content of the studies to discover the types and frequency of the various methods used (Fig. 6).

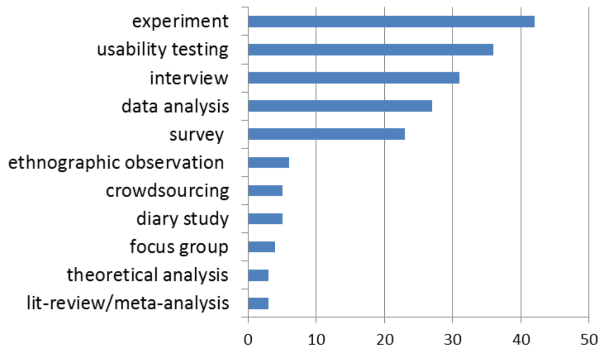


Figure 6. Research methods used in user studies

Experiment and usability testing were most commonly used (42%). The predominance of these methods may suggest that we are heavily focusing on evaluating what is out there rather than focusing on deeper problems or questions, a similar issue noted in other areas such as HCI [5]. These studies are primarily evaluating performance (e.g., error rate/time to perform task with a new system); identifying usability issues (i.e., interface design problems); or investigating acceptability of new system/ interface. The full user-centered design process should include stages supporting coming to an understanding of the users, development of system prototype(s), and evaluation of the prototypes with users. However, relatively few papers presenting a new system include both an initial user requirements elicitation study and a follow-up performance/usability/acceptability study.

We also investigated the scale of these user studies by tabulating how many human subjects were involved in these studies. 124 user studies involved human subjects, and 26 analyzed human-generated data such as queries, tags, etc. 7 studies did not directly involve human subjects or human-generated data, as they were papers based on literature review, meta-analysis, or theoretical reasoning. Figure 7 shows the distribution of the number of human subjects included in the studies of real users. Many studies are of fairly small scale: 57 of the 124 studies (46%) involve 20 or fewer human subjects, and 102 studies (82%) involve 100 or fewer subjects.

Note that the active involvement of participants is limited for lab experiments and usability tests, which typically run at most a couple of hours. Ethnographic observations are constrained by the time available to the researcher to conduct observations. Interviews, surveys, and focus groups are attractive in that they may invite introspection and comment on music-related behavior over the long term, but at the cost of relying primarily on retrospection rather than direct, measurable experience. Only data analysis and the diary study naturally offer the opportunity to examine authentic music information behavior over the long term, though 'long term' studies are mainly of one to four weeks. In evaluations of specific systems, the common finding is that the users like the new system and find the new interface entertaining or novel-— but it is generally unclear how or whether participant behavior may change after the novelty effect wears off.

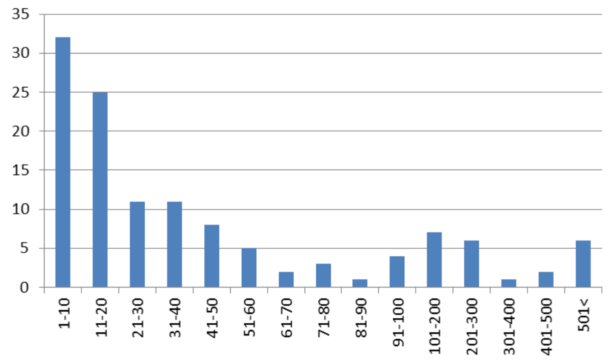


Figure 7. Number of subjects in user studies

6. DISCUSSION

Based on the results of our analyses as well as our own experiences in conducting music user studies, we provide a list of challenges/issues facing researchers who conduct music user studies which require further discussion. We believe that these issues are stemming from the uniqueness of the subject and the research domain.

6.1 Fast-changing Field

We believe that the speed with which the MIR field has evolved has had a strong affect on both the scale of user studies as well as the longevity of the research findings of these studies. The rapid development of tools and technologies for music storage, distribution, and experience in the past few decades has been remarkable. Some of the most popular music related services today such as Spotify or YouTube launched less than 6 years ago. This implies that how our users envision and expect from music services are most likely changing rapidly as well. Most of the young adults today probably never had to deal with physical media and grew up with various music streaming services. The results from studies that investigated how people find and purchase music on such physical media will have limited applicability today.

We conjecture that the fast-changing field is at least one of the reasons for the prevalence of small-scale studies. Large-scale studies take longer, in terms of recruiting human subjects, as well as collecting and processing data, in particular if researchers want to incorporate a qualitative component. Longitudinal studies are by definition time-consuming. Due to the rapidly changing environment, researchers are constantly under pressure to conduct and publish studies swiftly. This can be especially true for those who are trying to test a particular system or methods for providing access to music, as there is a good chance that by the time the research gets published in a journal, the results are already outdated. This may also explain a large proportion of user studies being published in conference or workshop proceedings.

6.2 Issue of Generalization

A large proportion of MIR user studies are small to moderate scale studies investigating a limited number of users. How does the scale of the study affect the generalizability of its results? Can we in fact make any reasonable inferences from studies of this scale that are generalizable to a larger user population? In addition, at least in certain parts of the world, it is not possible to obtain a comprehensive list of email addresses for the purpose of survey due to privacy concerns. This means that we often have to resort to convenience sampling, and study participants are in fact most frequently drawn from students or co-workers of the researchers which again can negatively affect the generalizability of our findings.

A point worth noting here is that researchers of music users are trying to grapple with this nebulous idea of users. Who really are our users? Where do we draw the boundaries? Music is so pervasive in our lives that it is difficult to know who is and is not affected by music. Moreover music is often enjoyed and sought out across different regions and cultures. Many of the MIR systems and services are now being used by global user base. Thus researchers of music users, in some sense, are expected to derive findings that can potentially have global implications on a wide range of users across space and time. Then how do we define and randomly sample this population in a practical sense? Even if we draw an artificial boundary and try to sample a smaller population, the subjects who participate in our studies will most likely be people who are interested in music to some degree. In this sense, the results are *always* likely to be biased.

Due to these issues, we believe that rather than aiming for generalizing the research findings, it might be helpful to take an alternative approach to understanding the purpose of these studies that each of these studies is discovering some piece of information about the users that is correct, but not comprehensive. When multiple pieces are put together, common themes emerge which we can generalize over multiple groups of users, as well as unique themes that can only apply to a particular user group.

6.3 Lack of Systematic Synthesis of Research Results

Although a large proportion (26%) of user studies were published in the proceedings of ISMIR conference, other studies were published in journals and conferences in multiple domains including LIS, HCI, Musicology, Psychology, etc. We had to repeat our search in multiple databases in order to retrieve all these studies scattered in multiple domains. Despite of our best efforts, we would not be surprised if there were studies we were not able to find. We suspect that this is probably one of the reasons hindering the synergic impact of these studies. Without being able to easily find all the previous user studies that have dealt with similar research questions and user populations, we will essentially reinvent the wheel every time. In order to resolve this issue, there is a need for additional review articles such as [7] and also an archive of all the citation information of user studies related to music. As the first step, we made our list of user studies with full citation available on the web[1]. However, a static webpage is far from an ideal way for collecting and sharing this type of information. We believe a more sustainable solution is needed, managed by multiple stakeholders.

6.4 The Disconnect Between System/Evaluation Task Designers and User Studies Researchers

In MIREX, the evaluation task is typically proposed by researcher(s) who are involved in developing algorithms related to the task. In the MIR domain, however, researchers who conduct user studies are not always algorithm developers themselves; this is especially true for researchers engaged in studies of music users focusing on information needs or behaviors. This disconnect may be one of the reasons why we have not seen a significant change in the way evaluation tasks have been run over the past seven years since MIREX started in 2005. Some of the suggestions made in the user studies might be logistically impossible to implement, or the evaluators might not even agree with those suggestions. Without a more thorough investigation asking the system developers and the organizers of evaluation tasks, it will be premature to determine what the exact reasons are.

7. CONCLUSION AND FUTURE WORK

In this paper, we reflected on how music user studies have been conducted and published, and what impact these studies have had on the field. Findings from our analysis suggest that there may be multiple layers of barriers for the user studies to make a strong impact: lack of findability due to the scattered patterns of publication, weak connections among scholars, dominance of small scaled studies that are difficult to generalize, etc. The purpose of this work is to provide an opportunity for starting a discussion at the ISMIR where many stakeholders involved in MIR research can together explore potential solutions to the issues raised in this paper. Thus, we want to conclude our paper with a set of questions that need further discussion:

For researchers conducting user studies:

- How can we provide systematic and intelligent access to the work we produce? Is there a sustainable method? Maybe a collaboratively managed resource?
- Is it necessary to change our research questions, methods, study populations, or venues in increase impact and affect change in the field?

For system and evaluation task designers/developers:

- How do you find out about new research on users and keep yourself updated? Are there particular kinds of publications do you seek often?
- What kinds of user studies do you find most and least useful? What do you see as the grand challenge in the area of MIR user studies?

In our future studies, we plan to survey and interview designers/developers of music related services and systems as well as organizers of MIREX evaluation tasks in order to more deeply understand the impact of these user studies. Specifically, we are interested in how the information on users are disseminated and diffused in the MIR domain, and how that knowledge may or may not affect the ways music services/systems are designed and modified. A deeper understanding on what kind of user information is actually sought by system designers/developers will be significant for researchers of MIR user studies.

8. ACKNOWLEDGMENTS

We thank Gary Gao and Tiffany Huang at University of Washington for their valuable contributions to this project.

9. REFERENCES

[1] S. Bradford: "Sources of information on specific subjects," *Journal of Information Science,* Vol. 10, No. 4, pp. 173-180, 1985.

[2] S. J. Cunningham: "User studies: a first step in designing an MIR testbed," In *The MIR/MDL Evaluation Project White Paper Collection* (3rd ed.). Champaign, Illinois: GSLIS, pp. 19-21, 2003.

[3] J. S. Downie: "Music information retrieval," *Annual Review of Information Science and Technology*, Vol. 37, ed. B. Cronin, Medford, NJ: Information Today, pp. 295-340, 2003.

[4] J. Futrelle and J. S. Downie: "Interdisciplinary communities and research issues in music information retrieval," *Proceedings of the 3rd ISMIR Conference*, pp. 215-221, 2002.

[5] S. Greenberg and B. Buxton: "Usability evaluation considered harmful (some of the time)," *Proceedings of the CHI'08*, pp. 111-120, 2008.

[6] D. L. Hansen, B. Schneiderman, and M. A. Smith: *Analyzing social media networks with NodeXL: Insights from a connected world*, Burlington, MA: Morgan Kaufmann, 2011.

[7] D. M. Weigl and C. Guastavino: "User studies in the music information retrieval literature," *Proceedings of the 12th ISMIR Conference*, pp. 335-340, 2011.

[1] http://www.jinhalee. com/miruserstudies

MEL CEPSTRUM & ANN OVA: THE DIFFICULT DIALOG BETWEEN MIR AND MUSIC COGNITION

Jean-Julien Aucouturier & Emmanuel Bigand
LEAD/CNRS UMR 5022, University of Burgundy, Dijon, France.
aucouturier@gmail.com; bigand@u-bourgogne.fr

Mel is a MIR researcher (the audio type) who's always been convinced that his field of research had something to contribute to the study of music cognition. His feeling, however, hasn't been much shared by the reviewers of the many psychology journals he tried submitting his views to. Their critics, rejecting his data as irrelevant, have frustrated him - the more he tried to rebut, the more defensive both sides of the debate became. He was close to give up his hopes of interdisciplinary dialog when, in one final and desperate rejection letter, he sensed an unusual touch of interest in the editor's response. She, a cognitive psychologist named Ann, was clearly open to discussion. This was the opportunity that Mel had always hoped for: clarifying what psychologists really think of audio MIR, correcting misconceptions that he himself made about cognition, and maybe, developing a vision of how both fields could work together. The following is the imaginary dialog that ensued. Meet Dr Mel Cepstrum, the MIR researcher, and Prof. Ann Ova, the psychologist.

1. ON AUDIO FEATURES

Ann Ova: Let me start with a tentative definition of what we, music cognition researchers, are interested in. To me, cognition is like digestion: a chain of transformations affecting a stimulus (e.g. a piece of music reaching the ears), transforming it, breaking it into blocks and eventually metabolizing it to produce a behavior (an emotional reaction, recognition, learning, etc.). As researchers, we are seeking to understand this mechanism of "stimulus digestion": what in the signal triggers it, how it is activated, what brain/mind functions are required.

Mel Cepstrum: When I hear this, I form the impression that your collective goal is not very different from ours in Music Information Retrieval. First, we study the same behaviors: the recognition of music into melodies, artists, styles, genres, or the prediction of emotional reactions. Second, we too are looking for mechanisms, which we prefer to call algorithms, and we conceptualize them using similar steps: sensory transformations first (we'd call this the signal processing front-end or feature extraction), then linking to memory and learning (we'd say databases and statistical models). It is therefore surprising to me that a lot of work in music cognition tends to rely on audio characteristics that can be extracted "by ear", thus ignoring much of our work in the past 10 to 15 years on musical signal processing. For instance, of the nearly 1,000 pages of the Handbook of Music and Emotion [14], not a single one is devoted to computerized signal analysis, but examples abound of research asking participants to subjectively evaluate a musical extract's tempo, complexity, height etc. on scales from 1 to 5, so these characteristics can be correlated with what you call "behavior". While I understand this may have been the only approach available to, say, Robert Francès in 1958 [10], surely you do realize that all of this (pitch extraction, beat tracking, etc.) can now be automated with computer algorithms? What's the superiority of doing it by hand?

A.O. This is true, much of what we study is analyzed by hand, or rather "by ear", by participants. I believe the advantage of doing so is that we only consider as possible acoustic correlates of a given behavior constructs that can be cognitively assessed by the participants themselves. We want to use what they *really* hear, not what a computer thinks they hear, and the best way to do this is to simply ask them.

M.C. But you'll agree that there are unique advantages to automatic analysis: it's fast and cheap, you can process a large number of stimuli in just minutes, while it would take a large number of participants to do the same by ear.

A.O. I understand this is an important criteria in your field - certainly one does not want to index iTunes by hand, but this is not an important concern for us. If a particular experimental design is expensive in terms of experimenter and participant time, but it is the design of choice, so be it.

M.C. Right - but isn't automatic signal analysis also more objective? It can extract physical properties from the signal, e.g. the *root-mean-square* that qualifies its physical energy or the *zero-crossing-rate* which describes the noisiness of the waveform - without mediating these by cognitive judgments. It can also realistically simulate the audio processing chain of the peripheral auditory system. For instance, *Mel-Frequency Cepstrum Coefficients*, a mathematical construct derived from the signal's Fourier transform, are designed to reproduce the non-linear tonotopic scale of the cochlea and the dynamical response of the hair cells of the basilar membrane.

A.O. This is only partly correct, you see. If you look at MFCCs closely (take Logan [23], say), you see that parts of the algorithm were designed to improve their computational value for machine learning, and not at all to improve their cognitive relevance. That final discrete cosine transform, for instance, is used to reduce correlations between coefficients, which would make their statistical modeling more complex. Now, one could argue I guess that the brain uses a similar computational trick - authors like Lewicki [19] are thinking along these lines, I suppose - but you'll agree that the responsibility rests on us, researchers, to prove that correct. Until then, MFCCs are maths. Useful maths for you perhaps, but irrelevant to our concerns.

M.C. Wait, that's a bit harsh. How about that study by Terasawa, Slaney and their colleagues at Stanford [31]: they resynthesized sounds from MFCCs and showed that human timbre dissimilarity ratings between sounds correlated exactly with the MFCCs. Doesn't that prove something?

A.O. Good one. This is indeed an important study, perhaps the first to tackle this problem diligently. But what does this prove, you ask? That an algorithmic construction, the MFCC, closely predicts a cognitive judgement. Should we conclude the brain implements a discrete cosine transform? Probably not. Just like fitting reactors on an airplane and seeing it take off should not lead us to conclude anything about how birds fly. Don't you think?

M.C. You're killing me. Are you seriously rejecting 10 years' worth of results as mere coincidences? Our findings that, say, taking the derivative of MFCCs improve genre classification by 10%, or that periodicities in the range 1-10 seconds (the *rhythm fluctuation patterns* of Pampalk [27]) are enough to account for timbre similarity, shouldn't that, at least, give you some sort of *intuition* about how these behaviors are cognitively produced?

A.O. Sorry if I sounded dismissive. In theory, you're right, and actually, we have been paying attention to your work (initiative like the MIRToolbox [17] have helped). But in practice, it's been really difficult to use your work, or to derive useful intuitions from it. Let me show you what I mean here, precisely. I'm looking at this data produced in my lab, a series of emotional valence and arousal judgements produced by participants listening to very short musical extracts (shorter than 500ms). At this level, it is unlikely that emotional reactions result from a cognitive analysis of say, melody or harmony, because the extracts are too short to even include a single note. The question for us is therefore to understand what low-level features of the raw sound are responsible for the emotion. It's the classical Gjerdingen & Perrott scenario [12], isn't it? This, if I understand correctly, is the ideal use-case for MIR features: a quasi-stationary signal, mostly important by its timbre quality. Well, let's have a look.

Table 1 reproduces the results of a multivariate regression we computed between the stimuli's valence and arousal and the whole batch of features offered by the MIRToolbox [17]. Let's see what intuition I, the cognition researcher, should derive from this. We see stimulus valence is very well explained by, let me get this right, the *entropy of the period of the magnitude of the maximum peak detected every 50ms in the signal's chromagram* (a chromagram, as you know, gives at each successive time position the energy observed at the frequency corresponding to each note - c, c#, d, etc., of each octave). Similarly, stimulus arousal seems to result from the *standard deviation of the 6th MFCC* and the *mean of the*

Table 1: Top MIR features in a regression of valence and arousal emotional judgements

Regression for valence	
Feature	β
tonal_chromagram_peak_PeakMagPeriodEntropy	-0.75
tonal_mode_Mean	0.13
spectral_mfcc_PeriodAmp_8 (600 Hz +/- 66)	0.12
spectral_ddmfcc_PeriodEntropy_1 (133.3)	-0.11

Regression for arousal	
Feature	β
spectral_mfcc_Std_6 (466.6)	-0.34
spectral_mfcc_Mean_3 (266.6)	0.28
tonal_keyclarity_Std	-0.28
spectral_mfcc_Std_7 (533.3)	0.24

3rd, and - mind you - not the opposite.

M.C. Hmm. This seems a bit too complicated maybe?

A.O. See what I mean? That surely fits well to the data, but I'm sure you realize it does not actually *explain* anything. Even if we took it literally, this would be a formidable mix-bag of an explanation. We have here an emotional reaction, valence, of which neuroscience tells us it is at least partly pre-attentive and subcortical, and which we explain here with constructs requiring memory and statistical learning ("entropy"), rhythmic entrainment ("period"), temporal integration ("maximum peak"), harmonic analysis ("chromagram") and arguably a participant's musical training in a western culture (because the chromagram relies on the 12-tone western pitch system).

M.C. Well, you got a point. But isn't this exactly the same problem when psychologists rely on features evaluated subjectively by their participants? When they study cultural differences between western and Indian classical music, Balkwill & Thompson [2] argue emotions are related to their stimuli's musical complexity, which they measure by asking participants, I quote, "*to evaluate how much was going on melodically in the except - was there a lot of repetition as in "Mary had a little lamb?*". Now, isn't that carrying a lot of assumptions too? The construct of "*being like Mary had a little lamb?*" is probably only derivable at a cortical level, using a lot of cognitive functions such as memory, melodic representations, etc. and certainly presupposes the participants know of that song in the first place. Are these assumptions realistic knowing what we know of emotions?

A.O. Well, you're probably right. And I guess one could even add that MIR has the advantage of not hiding these assumptions under their apparent lexical simplicity. But still, you have to admit that the logics behind your typical MIR signal feature is difficult for us to follow. If we want to use it to prove anything, it is crucially important for us to know *what* we're dealing with: a physical measure? a cognitive model? Let's have a look. In the MIR bestiary, we find, first, features deriving from traditional psychoacoustics: for instance, the *spectral centroid* which is the traditional correlate for the first perceptual dimension identified by MDS studies of timbre [13] or the *log attack time*, which correlates with the second most-important dimension; then, your field offers quite a lot of mathematical variants of these same characteristics, which seem to be justified only by the fact that they are conceptually close (for instance, *spectral skewness*, the 3rd spectral moment, which seems to be included because of the special status of the first moment, the above-mentioned *centroid*) or even that they are easy enough to compute (*spectral entropy*, obtained by multiplying the Fourier spectrum with its logarithm); other features seem to start their career as intermediary steps in the processing chain of another feature, gain special status and then a name of their own (for instance, the "*fluctuation pattern*" you mentioned earlier [27], which was originally an intermediary step in a tempo extraction algorithm); or even, as by-products of other algorithms, like some measures of *pulse clarity* [16] which are in fact the error estimation in the output of a beat-tracking algorithm. And the list goes on, growing every year: the sole MIRToolbox library offers more than 300 features, very few of which having a clear epistemological status. Now, I do not doubt they serve your purpose well, but I hope you see it is unclear whether they can serve ours.

2. ON PRECISION AND GROUNDTRUTH

M.C. I do. And I have to admit it never occurred to me that our drive to optimize our features for precision (deriving such features, selecting variants that work, recombining them, etc.) had taken us so far from the cognitive reality. Still, isn't it paradoxical that this same process is taking us closer and closer to the actual phenomenon, in terms of percentage of precision? I mean, work like Liu & Zhang [21] simulate with more than 95% precision the human judgements of "depression" and "contentement" made when listening to more than 250 extracts of music, by combining features describing timbre (e.g. spectral centroid), rhythm (e.g. average autocorrelation peak) and intensity (e.g. rms). Even if their algorithm has no pretense of being a cognitive model, the fact that it agrees with humans 95% of the time on hundreds of stimuli can only make us think it captures a large share of the physical and sensory features used by human cognition - right?

A.O. Let me ask a question. Isn't this definition of precision, relative to a so-called ground truth, a bit illusionary? Does everybody, in every culture, have the same exact definition of what is, say, "rock music", or of 2 songs that "sound the same"?

M.C. I see where you're going with this. We, MIR researchers, have always been uneasy about this point, to be honest. We're stuck between 2 research traditions: one, machine learning, which is interested in the capacity of algorithms to learn from a set of examples, whatever these examples are. For this tradition, whether the ground truth is meaningful or not is irrelevant. It is just taken as a temporary gold standard, relative to which different algorithms can be compared. Whether "rock" is indeed "rock" or "jazz" does not matter - actually, we want algorithms that have the flexibility to also learn that "jazz" is "rock" if we like them to. However, we also have the second research goal of being useful to electronic music distribution systems. Now, in this world, defining a unique ground truth is suddenly very relevant, but you soon realize it is also close to impossible: we have plenty of examples where what some call "rock", others will call "pop" or "jazz" and so on. I guess that's what you would call individual variations. Most of our recent research tries to address this paradox: for instance, how tags learned on one dataset generalize to other datasets [24], how to personalize music recommendations [5] or even letting uses define their own personal categories in interaction with the system [26]. But one cannot just rule out the idea of precision. After all, you psychologists also have to rely on the same concept: take the psychoacoustics of musical timbre. What these studies do is, similarly, consider average similarity ratings over many users (not that many, incidentally, compared with the thousands of samples we are routinely dealing with in MIR), and select features that explain the best percentage of the data's variance - finding, for instance, that the spectral centroid of an extract correlates at 93% with the first principal component of the human-constructed timbre space. But why should we accept spectral centroid as an important "psychologically-validated" characteristics of timbre, and simultaneously reject, say, Liu & Zhang's *average autocorrelation peak* [21] (or Alluri & Toiviainen's 6th *band spectral flux [1]*) when it allows to classify emotions at 95%? Sometimes, I wonder if you have a bit of a "not-invented-here" bias...

A.O. You may be right. Perhaps we have been disregarding advances in signal processing just because they look complicated and we can't be bothered to follow what you've been doing. Signal features produced by recent MIR research could and probably should be integrated to modern psychoacoustics, especially for those problems that could not yet be solved, such as dissonance [28], and you'll have to teach us on that. However, your using the example of psychoacoustics is interesting. I don't know if you realize that the psychoacoustics methodology is designed to investigate percepts, i.e. the immediate psychological gestalts corresponding to those few physical characteristics that define an auditory object, regardless of the listener and its culture. A musical sound *has* pitch, loudness and timbre. These are percepts. The same sound, however, does not *have* genre or emotion - these are constructed cognitively; their value could change (e.g. if you start calling the sound "pop" instead of "rock") without changing the physical definition of a sound. Now, to be honest, the frontier between what's a percept and what's a cognitive construction has been challenged in recent years, with the realization that action and perception are intertwined, but still most cognition researchers would agree a fundamental difference remains between the two. I'm worried you're applying the psychoacoustics metaphor to behaviors (genres, emotions, etc.) for which it does not apply.

M.C This is fascinating. I realize just now that, all these years, I have been using the terms psychoacoustics and music cognition is a nearly interchangeable way. The more I think about it, the more I realize that indeed MIR takes a psychoacoustics approach to, as you say, genres and emotions, treating these as if they were a set of physical properties of the sound. What's surprising is that it works so well. In fact, we're not capturing "rock" or "sad" music, we're capturing *things that sound like* "rock", or things that sound like a "sad song". Because music is a structured human activity, there are a lot of regularities there: most "sad" music indeed sounds the same (dark timbre, low pitch, what have you). But these features do not *make* the music sad -

A.O. - or at least, you're not testing whether they do -

M.C. right. We can potentially find music that *is* sad without exhibiting any of these features.

A.O. Take, say, that Dixieland upbeat tune they play at funerals in New Orleans.

M.C Exactly. For these songs, our models will fail completely. But because such songs are rare (or at least they're rare in our test databases), say there are maybe 5% of them, we can still reach 95% performance without actually modeling anything specific about how, say, genre is cognitively constructed.

3. ON PHYSICAL AND COGNITIVE MODELS

A.O. It's a possibility, indeed. But you make it sound worse than it is, I think. It's not that your approach is better or worse than ours, but it's important that we understand the difference, and how we can be complementary. You're interested in the result, and how much algorithms and humans agree on it. In music cognition, I think we're less interested in the result than we are in the process. If we were to design computer algorithms to do maths, say, we're not interested in building machines that can multiply numbers as well and as fast as humans, but rather in doing them in such a way that multiplying 8*7 is more difficult than 3*4, as it is for humans.

M.C This is indeed a true difference between our disciplines. We're happy when we see our algorithms duly classify as "rock" certain songs that are clearly on the border of that definition (Queen's Bohemian Rapsody, say) ...

A.O. ... whereas we would rather understand what makes a song more prototypically "rock" than another, or how much "rock" does one have to listen to form a stable representation of what that genre is.

M.C But your problem in that case is how to measure prototypicality, because if you ask the same participants to judge it subjectively then your argument becomes completely tautological...

A.O. You're right

M.C. ... whereas MIR gives you a tool to do just this: a measure, let's say a physical measure, of the "rockness" of a song. How much it sounds like rock.

A.O. This, what you just said, is really interesting. The key word here is "physical". I believe that music cognition would gain a lot indeed if it had a more complete and powerful arsenal of tools to control stimuli physically. Tools that do not have the pretense of infringing into cognitive thinking, just purely, state-of-the-art physical modeling. If we start seeing MIR in this way, a lot of research avenues open I think.

M.C. In sum, in order to be useful to cognition, we should stop trying to do any ourselves.

4. ON FOLK PSYCHOLOGY

A.O. I can sense the irony, you know. This said, if I can make a small request, and I'm saying this in part jokingly but not solely, it would help indeed if you guys could at least stop using the word "semantics".

M.C. Wait... What?

A.O. "Semantics" - as in "a semantic model" of genre classification, "mixing acoustics with semantic" information, "semantic gap". Just, what do you mean by this?

M.C. Well, I suppose we take it as the "high-level" meaning of music, like saying "rock" is semantically related to youth and rebellion, electric guitars, all that linguistic and social knowledge around music. All which is where perception stops and, err..., cognition kicks in? Activating the semantic networks of musical concepts, err...

A.O. See: that. We hate it where you do that. Folk psychology. Like there is a box in our head somewhere with a knowledge base, and some kind of process that activates this or that depending on the input. You lose us instantly with that kind of thinking. If you browse the psychology literature, you will not find a single cognitive model which uses a "semantic" layer. That single word, let alone your using it assuming that it will appeal to us, does probably more harm to the dialogue between our disciplines than the mathematical complexity of your work. The "entropy of the period of the magnitude", I can deal with; "semantics", I sincerely have no idea. It literally drives me away.

M.C. Interesting - that certainly explains some reviewer reactions when I tried to communicate MIR results in psychological journals! Now, on the question of musical genre, you have to admit, conversely, that research in cognition does not have much to say about the links between social, lexical, sensory categories - all that we wrongly call "semantics". Neuroscience research has shown for instance that Wagner operas could prime recognition on such words as heroism, courage, etc. [15]. This has probably profound implications for everyday music perception. How come music cognition research is not studying this?

A.O. You're right. Most of us would consider that musical genre, as an object of study, is too complex, i.e. we know in advance that studying it won't help us isolate experimentally any particular process that could constitute it. For instance, if one wants to understand the sensory process by which a rock song is recognized as rock, it is simpler, more elementary if you will, to study the same process in the case of the recognition of environmental sounds. This latter case is less plagued by cultural learning, ambiguity, subjectivity that musical genre.

M.C I see. Unfortunately, we in MIR don't have that luxury. If iTunes users want rock music, we cannot easily justify to study hammer noises instead.

A.O. Naturally. Once again, your discipline is interested in the result, and we are interested in the process.

5. INSPIRING EXAMPLES

M.C. But I'd like to backtrack a bit to your argument that MIR could contribute to cognition as a tool for physical modeling. This sounded promising.

A.O. Yes, I believe there is room to invent a methodology to use MIR tools to build a scientific proof in cognition. Precisely, MIR can be used, I think, as a physical measure of the information available for human cognition in the musical signal for a given task. And this measure can be used to control our stimuli and separate what's in the signal from what's constructed out of it by cognition.

M.C. I think there is work that already goes in the direction. In the speech domain, de Boer & Kuhl [8] for instance have shown that speech recognition algorithms (hidden Markov models) have better word recognition performance when they are trained and tested on infant-directed speech (IDS, or "motherese") than adult speech, which they claim validates the argument that the developmental value of IDS is to bootstrap language learning.

A.O. It's a lovely result, and indeed a very good example of how to integrate a physical, holistic recognition algorithm into a cognitive argument. What's important here is that the algorithm is not presented as a cognitive model: nobody here is pretending that the human brain implements a hidden Markov model. It only gives a proof of feasibility: from a purely physical point of view, the information exists in the IDS signal to allow for an easier treatment than adult speech. It would be very difficult to replace the machine by a human measure in this argument - computer modeling was really the clever thing to use.

M.C. There are a few other examples. For instance, Kaplan and colleagues [25] show that machine learning can classify dog barks into contexts like being afraid, playful, etc. This was taken to indicate, for the first time, that dog vocalizations contain communicative features. Like you said, from a purely physical, objective point of view, the point is to show that the information exists in the signal to allow for a potential communicative use.

A.O. I think we have discovered a design pattern here: one could probably imagine a similar application with music.

M.C. Let's see. Could we show for instance with a machine's good recognition performance that there exist enough harmonic information in, say, Indian classical music to explain the good performance (e.g. [2]) of western listeners when they are asked to classify emotions in raags (even though they are not familiar with this musical tradition)?

A.O. I confirm: this is exactly the type of question that's interesting for us, music cognition researchers, and indeed, I wouldn't know how to prove this without MIR. Provided the algorithm uses a representation of "harmony" which is both plausible biologically and agnostic culturally, of course. Some low-level measure of consonance/dissonance rather than a chromagram, perhaps?

M.C. Studying this would be pretty interesting from our point of view too. These characteristics found "enough to explain" a behavior would allow us to improve the performance of our algorithms in cross-cultural contexts, which are becoming a key issue in MIR. A recent international project, CompMusic [29], has even brought forth the question whether our most trusted algorithms have a western bias, because they were "evolved" with corpuses composed mostly of western music. By the way, you see, we're perhaps less naive than you first thought...

6. A SECOND LOOK AT BIOLOGICAL PLAUSIBILITY

A.O. I agree a lot of questions overlap between our 2 disciplines, perhaps more than I had first assumed. On the other hand, you'll also have to recognize that we are less hermetic to computational modeling than you think. I already mentioned the MIRToolbox, which is gaining interest among music psychologists. But most of our recent work include some element of computational modeling, often inspired by neuroscience. For instance, recent studies in harmonic priming [3] rely on fairly advanced computational models of auditory short-term memory [18]. Recently, human performance in tempo tracking was even explained in Journal of Experimental Psychology by a non-linear oscillation model [22].

M.C. This is true. Curiously, we MIR researchers are aware of these algorithms (auditory models by Leman [18], Large [22] or Cariani [4]), but for some reason they are not widely used. The criteria here is, again, precision. In our experience, when we first try them, "cognitively plausible" algorithms tend to work less well than brute-force engineer solutions, so they quickly drift into oblivion before we spend much time with them. One example was Lidy and Rauber's study [20] applying a wide range of "psychoacoustical" optimizations for genre recognition and finding very little improvement, if any. So we conclude, a bit hastingly maybe, that they're below our standards. But we've already discussed the difference between optimizing precision and modeling an underlying cognitive process.

A.O. Indeed. But even if you consider precision alone, the claim that cognitively plausible models will always perform poorly is not necessarily true, I think. In the image processing community, models which follow biological constraints radically are now giving comparable performance to their non-biologically-plausible alternatives (e.g. Serre, Poggio and colleagues [30]), and even faster learning rates. Now, you could argue that more is known in the psychophysiology of the visual cortex than for the auditory cortex, and it is therefore logical that machine vision should be ahead, but it is less and less the case, I reckon. We now have a good understanding of the response patterns of neurons throughout the auditory pathway (see e.g. the idea of spectro-temporal receptive fields [11]), and computational models even exist to model them [6].

M.C. That's right. I have seen one application of these models to instrument timbre classification, but this was by researchers outside the MIR community [9], and I don't think we have really picked it up. I guess we should look into these more seriously.

A.O. I think we should indeed. The potential is not only better precision, but better interdisciplinary dialogue. Again, let's turn to machine vision for an example. The visual cognition community has now started to take inspiration from models like Serre's [30] to explain experimental results. For instance, I'm looking at a recent paper by Crouzet, Thorpe and colleagues [7], finding that humans are capable of ultra-fast face categorization. The authors write: "*Our ability to initiate directed saccades toward faces as early as 100–110 ms after stimulus onset clearly leaves little time for anything other than a feed-forward pass. [Conveniently,] there is recent evidence (Serre et al. 2007) that such a purely feed-forward hierarchical processing mechanisms may be sufficient to account for at least some forms of rapid categorization*". In terms of interdisciplinary collaboration, I look at this with envy.

M.C. This is inspiring indeed! Let's work together so that, in a few years' time, we can write similar arguments in a similar article, linking a MIR model with some aspect of music cognition to derive a common scientific conclusion.

7. REFERENCES

[1] Alluri, V. & Toiviainen, P. (2010). Exploring perceptual and acoustic correlates of polyphonic timbre. Music Perception, 27(3), 223-241.

[2] Balkwill, L.& Thompson, W. F. (1999). A cross-cultural investigation of the perception of emotion in music: Psychophysical and cultural cues. Music Perception, 17, 43-64

[3] Bigand, E. & Poulin-Charronnat, B. (2006). Are we "experienced listeners"? A review of the musical capacities that do not depend on formal musical training. Cognition, 100(1), 100-130

[4] Cariani, P. (2001). Temporal Codes, Timing Nets, and Music Perception. Journal of New Music Research, 30(2), 107-136.

[5] Celma, O. & Lamere, P. (2011). If You Like Radiohead, You Might Like This Article, AI Magazine, 32(3), 57-66

[6] Chi, T., Ru, P. & Shamma, S. (2005) Multi-resolution spectrotemporal analysis of complex sounds. Journal of Acoustical Society of America, 118(2), 887-906

[7] Crouzet, S., Kirchner, H. & Thorpe, S. (2010). Fast saccades toward faces: Face detection in just 100 ms, Journal of Vision, 10(4):16.1-17

[8] De Boer, B. & Kuhl, P. (2003) Investigating the role of infant-directed speech with a computer model. Acoustics Research Letters On-line 4(4), 129–134

[9] Elhilali, M, Shamma, SA, Thorpe, SJ & Pressnitzer, D (2007) Models of timbre using spectro-temporal receptive fields: investigation of coding strategies, in proc. 19th International Congress on Acoustics, Madrid, Spain.

[10] Francès, R. (1958) La perception de la musique, Paris: Vrin.

[11] Ghazanfar, A. & Nicolelis, M. (2001). The Structure & Function of Dynamic Cortical & Thalamic Receptive Fields, Cerebral Cortex, 11(3):183-93.

[12] Gjerdingen, R. *&* Perrott, D. (2008) Scanning the Dial: The Rapid Recognition of Music Genres, Journal of *New Music Research,* 37*(2),* 93–100

[13] Grey, J. M. (1977). Multidimensional perceptual scaling of musical timbres. Journal of the Acoustical Society of America, 61, 1270–1277.

[14] Juslin, P. & Sloboda, J. (2010) Handbook of Music and Emotion. Oxford University Press, USA

[15] Koelsch, S., Kasper, E, Sammler, D., Schulze, K., Gunter, T. & Friederici, A. (2004) Music, Language and Meaning: Brain Signatures of Semantic Processing, Nature Neuroscience 7, 302 - 307.

[16] Lartillot, O., Eerola, T., Toiviainen, P. & Fornari, J. (2008) , Multi-feature modeling of pulse clarity: Design, validation, and optimization, in proc 9th Int. Conference on Music Information Retrieval, Philadelphia PA, USA

[17] Lartillot, O. & Toiviainen, P. (2007) A Matlab Toolbox for Musical Feature Extraction From Audio, in proc. 10th Int. Conference on Digital Audio Effects, Bordeaux, France.

[18] Leman, M. (2000). An Auditory Model of the Role of Short-term Memory in Probe-tone Ratings. Music Perception, 17(4), 481-510.

[19] Lewicki, M.S. (2002). Efficient coding of natural sounds, Nature Neuroscience, 5 (4), 356-363.

[20] Lidy, T. & Rauber, A. (2005). Evaluation of Feature Extractors and Psycho-Acoustic Transformations for Music Genre Classification. in proc. 6th Int. Conference on Music Information Retrieval, London, UK.

[21] Liu, D. & Zhang, H.-J. (2006) Automatic mood detection and tracking of music audio signal, IEEE Transactions on Speech and Audio processing, 14(1), 5-18

[22] Loehr, J., Large, E. & Palmer, C. (2011). Temporal coordination in music performance: Adaptation to tempo change. Journal of Experimental Psychology: Human Perception and Performance, 37 (4), 1292–1309

[23] Logan, B. (2000) Mel Frequency Cepstral Coefficients for Music Modeling. in Proc. 1st Int. Conf. on Music Information Retrieval, Plymouth, MA, USA.

[24] Marques G., Domingues M., Langlois T. & Gouyon F. (2011). Three Current Issues in Music Autotagging. in proc. 12th Int. Conf. on Music Information Retrieval, Miami, FL, USA.

[25] Molnár, C., Kaplan, F., Roy, P., Pachet, F., Pongrácz, P., Dóka, A. & Miklósi, Á. (2008) Classification of dog barks: a machine learning approach. Animal Cognition, 11(3):389-400.

[26] Pachet, F (2008). The future of content is in ourselves. Computers in Entertainment,6(3), 2008.

[27] Pampalk, E., Flexer, A. & Widmer, G. (2005). Improvements of Audio-Based Music Similarity and Genre Classification, in proc. 6th Int. Conf. on Music Information Retrieval, London, UK.

[28] Peretz, I. (2008). The need to consider underlying mechanisms: A response from dissonance. Behavioral and Brain Sciences, 31, 590-591

[29] Serra, X. (2011). A Multicultural Approach in Music Information Research, in proc. 12th Int. Conf. on Music Information Retrieval - see also *http://compmusic.upf.edu/node/71*

[30] Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M. & Poggio, T. (2007). Object recognition with cortex-like mechanisms. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(3), 411-426

[31] Terasawa, H., Slaney, M. & Berger, J. (2005) . The Thirteen Colors of Timbre, in proc. IEEE Workshop on Appl. of Signal Proc. to Audio and Acoustics, New Paltz, NY, USA.

MOVING BEYOND FEATURE DESIGN: DEEP ARCHITECTURES AND AUTOMATIC FEATURE LEARNING IN MUSIC INFORMATICS

Eric J. Humphrey, Juan Pablo Bello
Music and Audio Research Lab, NYU
{ejhumphrey, jpbello}@nyu.edu

Yann LeCun
Courant School of Computer Science, NYU
yann@cs.nyu.edu

ABSTRACT

The short history of content-based music informatics research is dominated by hand-crafted feature design, and our community has grown admittedly complacent with a few de facto standards. Despite commendable progress in many areas, it is increasingly apparent that our efforts are yielding diminishing returns. This deceleration is largely due to the tandem of heuristic feature design and shallow processing architectures. We systematically discard hopefully irrelevant information while simultaneously calling upon creativity, intuition, or sheer luck to craft useful representations, gradually evolving complex, carefully tuned systems to address specific tasks. While other disciplines have seen the benefits of deep learning, it has only recently started to be explored in our field. By reviewing deep architectures and feature learning, we hope to raise awareness in our community about alternative approaches to solving MIR challenges, new and old alike.

1. INTRODUCTION

Since the earliest days of music informatics research (MIR), content-based analysis, and more specifically audio-based analysis, has received a significant amount of attention from our community. A number of surveys (e.g. [8, 22, 29]) amply document what is a decades-long research effort at the intersection of music, machine learning and signal processing, with wide applicability to a range of tasks including the automatic identification of melodies, chords, instrumentation, tempo, long-term structure, genre, artist, mood, renditions and other similarity-based relationships, to name but a few examples. Yet, despite a heterogeneity of objectives, traditional approaches to these problems are rather homogeneous, adopting a two-stage architecture of feature extraction and semantic interpretation, e.g. classification, regression, clustering, similarity ranking, etc.

Feature representations are predominantly hand-crafted, drawing upon significant domain-knowledge from music theory or psychoacoustics and demanding the engineering acumen necessary to translate those insights into algorithmic methods. As a result, good feature extraction is hard to come by and even more difficult to optimize, often taking several years of research, development and validation. Due in part to this reality, the trend in MIR is to focus on the use of ever-more powerful strategies for semantic interpretation, often relying on model selection to optimize results. Unsurprisingly, the MIR community is slowly converging towards a reduced set of feature representations, such as Mel-Frequency Cepstral Coefficients (MFCC) or chroma, now de-facto standards. This trend will only become more pronounced given the growing popularity of large, pre-computed feature datasets [1] .

We contend the tacit acceptance of common feature extraction strategies is short-sighted for several reasons: first, the most powerful semantic interpretation method is only as good as a data representation allows it to be; second, mounting evidence suggests that appropriate feature representations significantly reduce the need for complex semantic interpretation methods [2, 9]; third, steady incremental improvements in MIR tasks obtained through persistence and ingenuity indicate that the the costly practice of manual feature optimization is not yet over; and fourth, task-specific features are ill-posed to address problems for which they were not designed (such as mood estimation or melody extraction), thus limiting their applicability to these and other research areas that may emerge.

In this paper we advocate a combination of deep signal processing architectures and automatic feature learning as a powerful, holistic alternative to hand-crafted feature design in audio-based MIR. We show how deeper architectures are merely extensions of standard approaches, and that robust music representations can be achieved by breaking larger systems into a hierarchy of simpler parts (Section 3). Furthermore, we also show that, in light of initial difficulties training flexible machines, automatic learning methods now exist that actually make these approaches feasible, and early applications in MIR have shown much promise (Section 4). This formulation provides several important advantages over manual feature design: first, it allows for joint, fully-automated optimization of the feature extraction and semantic interpretation stages, blurring boundaries between the two; second, it results in general-purpose architectures that can be applied to a variety of specific MIR problems; and lastly, automatically learned features can offer objective insight into the relevant musical attributes for a given task. Finally, in Section 5, we

[1] Million Song Dataset: http://labrosa.ee.columbia.edu/millionsong/

conclude with a set of potential challenges and opportunities for the future.

2. CLASSIC APPROACHES TO CLASSIC PROBLEMS

2.1 Two-Stage Models

In the field of artificial intelligence, computational perception can be functionally reduced to a two-tiered approach of data representation and semantic interpretation. A signal is first transformed into a data representation where its defining characteristics are made invariant across multiple realizations, and semantic meaning can subsequently be inferred and used to assign labels or concepts to it. Often the goal in music informatics is to answer specific questions about the content itself, such as "is this a C major triad?" or "how similar are these two songs?"

More so than assigning meaning, the underlying issue is ultimately one of organization and variance. The better organized a representation is to answer some question, the simpler it is to assign or infer semantic meaning. A representation is said to be *noisy* when variance in the data is misleading or uninformative, and *robust* when it predictably encodes these invariant attributes. When a representation explicitly reflects a desired semantic organization, assigning meaning to the data becomes trivial. Conversely, more complicated information extraction methods are necessary to compensate for any noise.

In practice, this two-stage approach proceeds by feature extraction – transforming an observed signal to a hopefully robust representation – and either classification or regression to model decision-making. Looking back to our recent history, there is a clear trend in MIR of applying increasingly more powerful machine learning algorithms to the same feature representations to solve a given task. In the ISMIR proceedings alone, there are twenty documents that focus primarily on audio-based automatic chord recognition. All except one build upon chroma features, and over half use Hidden Markov Models to stabilize classification; the sole outlier uses a Tonnetz representation, which are tonal centroid features derived from chroma. Though early work explored the use of simple binary templates and maximum likelihood classifiers, more recently Conditional Random Fields, Bayesian Networks, and Support Vector Machines have been introduced to squeeze every last percentage point from the same features.

If a feature representation were truly robust, the complexity of a classifier – and therefore the amount of variance it could absorb – would have little impact on performance. Previous work in automatic chord recognition demonstrates the significance of robust feature representations, showing that the appropriate filtering of chroma features leads to a substantial increase in system performance for the simplest classifiers, and an overall reduction of performance variation across all classifiers [9]. Additionally, researchers have for some time addressed the possibility that we are converging to glass ceilings in content-based areas like acoustic similarity [2]. Other hurdles, like the issue of hubs and orphans, have been shown to be not merely a peculiarity of the task but rather an inevitability of the feature representation [20]. As we consider the future of MIR, it is necessary to recognize that diminishing returns in performance are far more likely the result of sub-optimal features than the classifier applied to them.

2.2 From Intuition to Feature Design

Music informatics is traditionally dominated by the handcrafted design of feature representations. Noting that design itself is a well-studied discipline, a discussion of feature design is served well by the wisdom of "getting the right design and the design right" [6]. Reducing this aphorism to its core, there are two separate facets to be considered: finding the right conceptual representation for a given task, and developing the right system to produce it.

Consider a few signal-level tasks in MIR, such as onset detection, chord recognition or instrument classification, noting how each offers a guiding intuition. Note onsets are typically correlated with transient behavior. Chords are defined as the combination of a few discrete pitches. Classic studies in perception relate timbre to aspects of spectral contour [12]. Importantly, intuition-based design hinges on the assumption that someone can know what information is necessary to solve a given problem.

Having found conceptual direction, it is also necessary to craft the right implementation. This has resulted in substantial discourse and iterative tuning to determine better performing configurations of the same basic algorithms. Much effort has been invested in determining which filters and functions make better onset detectors [3]. Chroma – arguably the only music-specific feature developed by our community – has undergone a steady evolution since its inception, gradually incorporating more levels of processing to improve robustness [28]. Efforts to characterize timbre, for which a meaningful definition remains elusive, largely proceed by computing numerous features or, more commonly, the first several MFCCs [11].

In reality, feature design presents not one but two challenges – concept and implementation – and neither have proven easy to solve. First off, our features are ultimately constrained to those representations we can conceive or comprehend. Beyond relatively obvious tasks like onset detection and chord recognition, we can only begin to imagine what abstractions might be necessary to perform rather abstract tasks like artist identification. Furthermore, recognizing that feature extraction is still an open research topic, the considerable inertia of certain representations is cause for concern: 19 of 26 signal-based genre classification systems in the ISMIR proceedings are based on MFCCs, for example, many using publicly-available implementations. While sharing data and software is a commendable trend, now is a critical point in time to question our acceptance of these representations as we move toward the widespread use of pre-computed feature collections, e.g. the Million Song Dataset. Finally, above all else, the practice of handcrafted feature design is simply not sustainable. Manually optimizing feature extraction methods proceeds at a glacial pace and incurs the high costs of time, effort and funding. Somewhat ironically, the MIR community has collectively recognized the benefits of automatically fitting our classifiers, but feature optimization – the very data those methods depend on – remains largely heuristic.

Alternatively, data-driven approaches in *deep learning* have recently shown promise toward alleviating each and every one of these issues. Proven numerical methods can adapt a system infinitely faster than is attainable by our current research methodology, and the appropriate conceptual representations are realized as a by-product of optimizing an objective function. In the following section, we will illustrate how robust feature representations can be achieved through deep, hierarchical structures.

3. DEEP ARCHITECTURES

3.1 Shallow Architectures

Time-frequency analysis is the cornerstone of audio signal processing, and modern architectures are mainly comprised of the same processing elements: linear filtering, matrix transformations, decimation in time, pooling across frequency, and non-linear operators, such as the complex modulus or logarithmic compression. Importantly, the combination of time-domain filtering and decimation is often functionally equivalent to a matrix transformation – the Discrete Fourier Transform (DFT) can be easily interpreted as either, for example – and for the sake of discussion, we refer to these operations collectively as *projections*.

Now, broadly speaking, the number of projections contained within an information processing architecture determines its *depth*. It is critical to recognize, however, that the extraction of meaningful information from audio proceeds by transforming a time-varying function – a signal – into an instantaneous representation – features; at some specificity, all signals represent static concepts, e.g., a single piano note versus the chorus of a song. Therefore, the depth at which a full signal is summarized by a stationary feature vector is characteristic of a signal processing architecture, and is said to be particularly *shallow* if an entire system marginalizes the temporal dimension with only a single projection.

This is a subtle, but crucial, distinction to make; *feature* projections, lacking a time dimension, are a subset of *signal* projections. As we will see, shallow signal processing architectures may still incorporate deep feature projections, but the element of time warrants special attention. A signal projection that produces a finite set of stationary features attempts to capture *all* relevant information over the observation, and any down-stream representations are constrained by whatever was actually encoded in the process. Importantly, the range of observable signals becomes infinite with increasing duration, and it is progressively more taxing for signal projections – and therefore shallow architectures – to accurately describe this data without a substantial loss of information.

To illustrate the point further, consider the two signal processing architectures that produce Tonnetz and MFCC features. As shown in Figure 1, the processing chains are nearly identical; note that the penultimate representation when computing Tonnetz features is chroma. Both begin with a signal projection that maps a time-domain signal to an instantaneous estimation of frequency components, and conclude with a feature projection that reorganizes the estimated frequencies in task-specific ways. The overwhelming majority of music signal processing architectures operate in this paradigm of shallow signal transformations. Subject to the Fourier uncertainty principle, these systems exhibit time-frequency trade-offs and are constrained in practice to the analysis of short observations.

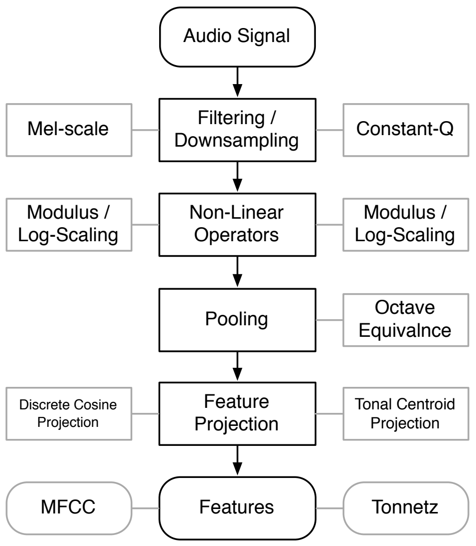


Figure 1: Tonnetz and MFFCs from Shallow Architectures

The vast majority of musical experiences do not live in short signals however, and it is therefore necessary to characterize information over longer durations. Previous efforts recognize this deficiency and address it through one of a few simple methods: a *bag of frames (BoF)* models features as a probability distribution, *shingling* concatenates feature sequences into a vector, or *delta-coefficients* represent low-order derivatives calculated over local features. These naive approaches are ill-posed to characterize the temporal dynamics of high-level musical concepts like mood or genre, and arguably contribute to the "semantic gap" in music informatics. It will become clear in the following discussion why this is the case, and how deeper architectures can alleviate this issue.

3.2 Motivating Deeper Architectures

This previous discussion begs a rather obvious question: why are shallow architectures poorly suited for music signal processing? If we consider how music is constructed, it is best explained by a *compositional containment hierarchy*. The space of musical objects is not flat, but rather pitch and intensity combine to form chords, melodies and rhythms, which in turn build motives, phrases, sections and entire pieces. Each level uses simpler elements to produce an emergent whole greater than the sum of its parts, e.g., a melody is more than just a sequence of pitches.

In a similar fashion, deeper signal processing structures can be realized by stacking multiple shallow architectures, and are actually just extensions of modern approaches. For a signal projection to marginalize time with a minimal loss of information, the observation must be locally stationary, and clearly this cannot hold for long signals. Sequences of instantaneous features, however, are again time-varying data and, when appropriately sampled, *are* themselves locally stationary signals. There are two remarkable conclusions to draw from this. First, everything we know about one-dimensional signal processing holds true for a time-feature signal and can be generalized thusly. And furthermore, simply cascading multiple shallow architectures relaxes previous constraints on observation length by producing locally stationary signals at various time-scales.

This hierarchical signal processing paradigm is at the heart of deeper architectures. There are many benefits detailed at length in [4], but two are of principal importance here: one, multi-layer processing allows for the emergence of higher-level attributes for two related reasons: deep structures can break down a large problem into a series of easier sub-problems, and each requires far fewer elements to solve than the larger problem directly; and two, each layer can absorb some specific variance in the signal that is difficult or impossible to achieve directly. Chord recognition captures this intuition quite well. One could define every combination of absolute pitches in a flat namespace and attempt to identify each separately, or they could be composed of simpler attributes like intervals. Slight variations, like imperfect intonation, can be reconciled by a composition of intervals, whereas a flat chord-space would need to address this explicitly.

Both of these benefits are observed in the successful application of convolutional neural networks (CNN) to handwritten digit classification [25]. Most prior neural network research in computer vision proceeded by applying multilayer perceptrons (MLP) directly to a pixel values of an image, which struggles to cope with spatial variation. Adopting a CNN architecture introduces a hierarchical decomposition of small, locally-correlated areas, acting as signal projections in space rather than time. Emergent properties of images are encoded in the visual geometry of edges, corners, and so on, and the architecture is able to develop an invariance to spatial translations and scaling.

Within audio signal processing, wavelet filterbanks, as cascaded signal projections, have been shown to capture long-term information for audio classification [1]. These second-order features yielded better classification results than first-order MFCCs over the same duration, even allowing for convincing signal reconstruction of the original signals. This outcome is evidence to the fact that deeper signal processing architectures can lead to richer representations over longer durations. Observing that multi-layer architectures are simply extensions of common approaches, it is fascinating to discover there is at least one instance in MIR where a deep architecture has naturally evolved into the common solution: tempo estimation.

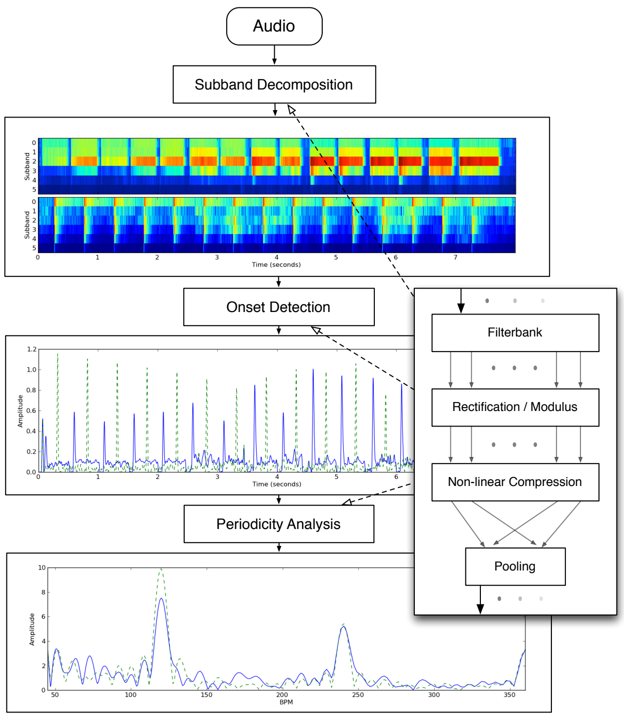


Figure 2: Tempo Estimation with Deep Signal Processing Architectures.

3.3 Deep Signal Processing in Practice

Upon closer inspection, modern tempo estimation architectures reveal deep architecture with strong parallels to CNNs and wavelets. Rhythmic analysis typically proceeds by decomposing an audio signal into frequency subbands [31]. This time-frequency representation is logarithmically scaled and subbands are pooled, reducing the number of components. Remaining subbands are filtered in time by what amounts to an edge detector, rectified, pooled along subbands and logarithmically scaled to yield a novelty function [23]. A third and final stage of filtering estimates tempo-rate frequency components in the novelty signal, producing a tempogram [13].

Over the course of a decade, the MIR community has collectively converged to a deep signal processing architecture for tempo estimation and, given this progress, it is possible to exactly illustrate the advantages of hierarchical signal analysis. In Figure 2, two waveforms with identical tempi but different incarnations – a trumpet playing an ascending D major scale and a series of bass drum hits, set slightly out of phase – are shown at various stages of the tempo estimation architecture. It is visually apparent that each stage in the architecture absorbs a different type of variance in the signal: pitch and timbre, absolute amplitude, and phase information, respectively. By first breaking the problem of tempo estimation into two sub-tasks – frequency estimation and onset detection – it becomes possible to characterize subsonic frequencies at both lower sampling frequencies and with a fewer number of components.

Realistically though, progress in tempo estimation is the result of strong intuition that could guide system design. The inherent challenge in building deep, hierarchical systems is that intuition and understanding quickly depart after more than even a few levels of abstraction. Therein lies the most exciting prospect of this whole discourse; given a well-defined objective function, it is possible to automatically learn both the right conceptual representation and the right system to produce it for a specific application.

4. FEATURE LEARNING

4.1 From Theory to Practice

For some time, a concerted effort in computer science has worked toward the development of convex optimization and machine learning strategies. Unfortunately, the initial surge of activity and excitement surrounding artificial intelligence occurred well before technology could handle the computational demands of certain methods, and as a result many approaches were viewed as being intractable, unreasonable, or both. Over the last two or so decades, the state of affairs in machine learning has changed dramatically, and for several reasons feature learning is now not only feasible, but in many cases, efficient.

Almost more importantly than its success as an image classification system, the work in [25] proved that stochastic gradient descent could be used to discriminatively train large neural networks in a supervised manner. Given a sufficient amount of labeled data, many applications in computer vision immediately benefited from adopting these approaches. Such datasets are not always available or even possible, and recent breakthroughs in unsupervised training of Deep Belief Networks (DBNs) have had a similar impact [17]. This work has also been extended to a convolutional variant (CDBNs), showing great promise for deep signal processing [26]. Additionally, auto-encoder architectures are a recent addition to the unsupervised training landscape and offer similar potential [21].

The significance of ever-increasing computational power is also not to be overlooked in the proliferation of automatic feature learning. Steady improvements in processing speed are now being augmented by a rise in parallel computing solutions and toolkits [5], decreasing training times and accelerating research. Taken together, these strategies encompass a set of deep learning approaches that hold significant potential for applications in music informatics.

4.2 Early Efforts in Music Informatics

It is necessary to note that leveraging data to automatically learn feature representations is not a new idea. The earliest effort toward automatic feature learning is that of [7, 33], where genetic algorithms were used to automatically learn optimized feature transformations.Though not a deep architecture in the classic sense, this work formally recognized the challenge of hand-crafting musical representations and pioneered feature learning in MIR.

With respect to deeper architectures, the first successful instance of deep feature learning is that of CNN-based onset detection by [24]. More recently, CNNs have been applied to automatic genre recognition [27], instrument classification [19] and automatic chord recognition [18]. Alternatively, DBNs have seen a noticeable rise in frame-level applications, such as instrument classification [15], piano transcription [30], genre identification [14] and mood prediction [32], out-performing other shallow, MFCC-based systems. Incorporating longer time-scales, convolutional DBNs have also been explored in the context of various speech and music classification tasks in [26], and for artist, genre and key recognition [10]. Predictive sparse coding has also been applied to genre recognition, earning "Best Student Paper" at ISMIR 2011 [16].

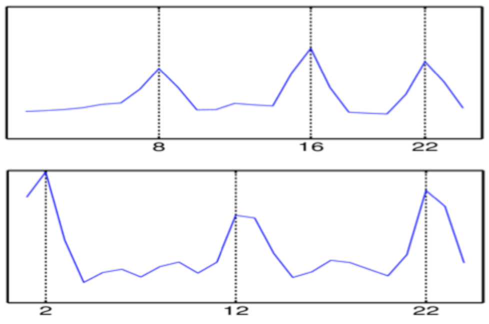


Figure 3: Learned Features for Genre Recognition (Reprinted with permission)

The most immediate observation to draw from this short body of work is that every system named above achieved state-of-the-art performance, or better, in substantially less time than it took to get there by way of hand-crafted representations. Noting that many of these systems are the first application of deep learning in a given area of MIR, it is only reasonable to expect these systems to improve in the future. For instance, DBNs have been primarily used for frame-level feature learning, and it is exciting to consider what might be possible when all of these methods are adapted to longer time scales and for new tasks altogether.

A more subtle observation is offered by this last effort in genre recognition [16]. Interestingly, the features learned from Constant-Q representations during training would seem to indicate that specific pitch intervals and chords are informative for distinguishing between genres. Shown in Figure 3, learned dictionary elements capture strong fifth and octave interval relationships versus quartal intervals, each being more common in rock and jazz, respectively. This particular example showcases the potential of feature learning to reformulate established MIR tasks, as it goes against the long-standing intuition relating genre to timbre and MFCCs.

5. THE FUTURE OF DEEP LEARNING IN MIR

5.1 Challenges

Realistically speaking, deep learning methods are not without their own research challenges, and these difficulties are contributing factors to limited adoption within our community. Deep architectures often require a large amount of labeled data for supervised training, a luxury music informatics has never really enjoyed. Given the proven success of supervised methods, MIR would likely benefit a good deal from a concentrated effort in the curation of sharable data in a sustainable manner. Simultaneously, unsupervised methods hold great potential in music-specific contexts, as they tend to circumvent the two biggest issues facing supervised training methods: the threat of over-fitting and a need for labeled data.

Additionally, there still exists a palpable sense of mistrust among many toward deep learning methods. Despite decades of fruitful research, these approaches lack a solid, foundational theory to determine how, why, and if they will work for a given problem. Though a valid criticism, this should be appreciated as an exciting research area and not a cause for aversion. Framing deep signal processing architectures as an extension of shallow time-frequency analysis provides an encouraging starting point toward the development of more rigorous theoretical foundations.

5.2 Impact

Deep learning itself is still a fledgling research area, and it is still unclear how this field will continue to evolve. In the context of music informatics, these methods offer serious potential to advance the discipline in ways that cannot be realized by other means. First and foremost, it presents the capacity for the abstract, hierarchical analysis of music signals, directly allowing for the processing of information over longer time scales. It should come as no surprise that determining the similarity of two songs based on small-scale observations has its limitations; in fact, it should be amazing that it works at all.

More practically, deep learning opens the door for the application of numerical optimization methods to accelerate research. Instead of slowly converging to the best chroma transformation by hand, an automatically trained system could do this in a fraction of the time, or find a better representation altogether. In addition to reframing well-known problems, deep learning also offers a solution to those that lack a clear intuition about how a system should be designed. A perfect example of this is found in automatic mixing; we know a "good" mix when we hear one, but it is impossible to articulate the contributing factors in a general sense. Like the work illustrated in Figure 3, this can also provide insight into what features are informative to a given task and create an opportunity for a deeper understanding of music in general.

6. REFERENCES

[1] J. Andén and S. Mallat. Multiscale scattering for audio classification. In *Proc. ISMIR*, 2011.

[2] J. J. Aucouturier. Music similarity measures: What's the use? In *Proc. ISMIR*, 2002.

[3] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler. A tutorial on onset detection in music signals. *IEEE Trans. Audio, Speech and Language Processing*, 13(5):1035–1047, 2005.

[4] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2:1–127, 2009.

[5] 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 *Proc. SciPy*, 2010.

[6] B. Buxton. *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann, 2007.

[7] G. Cabral and F. Pachet. Recognizing chords with EDS: Part One. *Computer Music Modeling and Retrieval*, pages 185 – 195, 2006.

[8] M. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proc. IEEE*, 96(4):668–696, 2008.

[9] T. Cho, R. J. Weiss, and J. P. Bello. Exploring common variations in state of the art chord recognition systems. In *Proc. SMC*, 2010.

[10] S. Dieleman, P. Brakel, and B. Schrauwen. Audio-based music classification with a pretrained convolutional network. In *Proc. ISMIR*, 2011.

[11] S. Essid, G. Richard, and B. David. Musical instrument recognition by pairwise classification strategies. *IEEE Trans. Audio, Speech and Language Processing*, 14(4):1401–1412, 2006.

[12] J. M. Grey. Multidimentional perceptual scaling of musical timbre. *Jnl. Acoustical Soc. of America*, 61:1270–1277, 1977.

[13] P. Grosche and M. Müller. Extracting predominant local pulse information from music recordings. *IEEE Trans. Audio, Speech and Language Processing*, 19(6):1688–1701, 2011.

[14] P. Hamel and D. Eck. Learning features from music audio with deep belief networks. In *Proc. ISMIR*, 2010.

[15] P. Hamel, S. Wood, and D. Eck. Automatic identification of instrument classes in polyphonic and poly-instrument audio. In *Proc. ISMIR*, 2009.

[16] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun. Unsupervised learning of sparse features for scalable audio classification. In *Proc. ISMIR*, 2011.

[17] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[18] E. J. Humphrey, T. Cho, and J. P. Bello. Learning a robust tonnetz-space transform for automatic chord recognition. In *Proc. ICASSP*, 2012.

[19] E. J. Humphrey, A. P. Glennon, and J. P. Bello. Non-linear semantic embedding for organizing large instrument sample libraries. In *Proc. ICMLA*, 2010.

[20] I. Karydis, M. Radovanovic, A. Nanopoulos, and M. Ivanovic. Looking through the "glass ceiling": A conceptual framework for the problems of spectral similarity. In *Proc. ISMIR*, 2010.

[21] K. Kavukcuoglu, P. Sermanet, Y. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Learning convolutional feature hierarchies for visual recognition. In *Proc. NIPS*, 2010.

[22] A. Klapuri and M. Davy. *Signal Processing Methods for Music Transcription*. Springer, 2006.

[23] A. P. Klapuri, A. J. Eronen, and J. T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Trans. Audio, Speech and Language Processing*, 14(1):342–355, 2006.

[24] A. Lacoste and D. Eck. A supervised classification algorithm for note onset detection. *EURASIP Jnl. on Adv. in Signal Processing*, pages 1–14, 2007.

[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[26] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proc. ICML*, 2009.

[27] T. Li, A. Chan, and A. Chun. Automatic musical pattern feature extraction using convolutional neural network. In *Proc. IMECS*, 2010.

[28] M. Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.

[29] M. Müller, D.P.W. Ellis, A. Klapuri, and G. Richard. Signal processing for music analysis. *Jnl. Selected Topics in Sig. Proc.*, 5(6):1088–1110, 2011.

[30] J. Nam, J. Ngiam, H. Lee, and M. Slaney. A classification-based polyphonic piano transcription approach using learned feature representations. In *Proc. ISMIR*, 2011.

[31] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *Jnl. Acoustical Soc. of America*, 103(1):588–601, 1998.

[32] E. M. Schmidt and Y. E. Kim. Modeling the acoustic structure of musical emotion with deep belief networks. In *Proc. NIPS*, 2011.

[33] A. Zils and F. Pachet. Automatic extraction of music descriptors from acoustic signals using EDS. In *Proc. AES*, 2004.

REUSE, REMIX, REPEAT: THE WORKFLOWS OF MIR

Kevin R. Page [1] **Ben Fields** [2,3] **David De Roure** [1] **Tim Crawford** [3] **J. Stephen Downie** [4]

[1]Oxford e-Research Centre, University of Oxford [2]Musicmetric (Semetric Ltd.)

[3]Department of Computing, Goldsmiths, University of London

[4]Graduate School of Library and Information Sciences, University of Illinois

ABSTRACT

Many solutions for the reuse and remixing of MIR methods and the tools implementing them have been introduced over recent years. Proposals for achieving the necessary interoperability have ranged from shared software libraries and interfaces, through common frameworks and portals, to standardised file formats and metadata. Each proposal shares the desire to reuse and combine repurposable components into assemblies (or "*workflows*") that can be used in novel and possibly more ambitious ways. Reuse and remixing also have great implications for the process of MIR research. The encapsulation of any algorithm and its operation – including inputs, parameters, and outputs – is fundamental to the repeatability and reproducibility of any experiment. This is desirable both for the open and reliable evaluation of algorithms (e.g. in MIREX) and for the advancement of MIR by building more effectively upon prior research. At present there is no clear best practice widely adopted throughout the community. Should this be considered a failure? Are there limits to interoperability unique to MIR, and how might they be overcome? In this paper we assess contemporary MIR solutions to these issues, aligning them with the emerging notion of Research Objects for reproducible research in other domains, and propose their adoption as a route to reuse in MIR.

1. INTRODUCTION

The integration of tools for Music Information Retrieval (MIR) into a "complete system" has been repeatedly identified as a key – if not the grand – challenge [5, 6] for our community. This stems from the predominance of tools that are designed to solve a specific task, often developed in different frameworks, and usually with incompatible formats for input, output, and parameters. Production of any more sophisticated application that combines several techniques therefore requires either a full reimplementation and combination of the constituent algorithms, potentially without source code or a sufficient published description of the method, or development of a mechanism through which the original tools can be reused or interoperate.

The benefits of the latter approach appear multiple and desirable, that is to:
1. realise any number of "complete systems" assembled from building block components; specialised versions of our tools for different music-related end-user communities.
2. "stand on the shoulders of giants" and advance research by building upon and reusing prior methods and results.
3. optimise systems through reuse of data, as well as functionality, at points of interoperability, e.g. to reuse already calculated features.
4. build distributed systems [12] through reuse of network exposed interoperability.
5. reuse the mechanisms of interoperability for the purposes of transparent comparability in evaluation systems such as those undertaking MIREX.

Yet despite the steady production of frameworks and toolkits over many years a de facto standard has failed to emerge. In this paper we assess reuse through consideration of MIR research as a data intensive scientific method, and assess how a selection of MIR tools might meet the requirements of scientific workflow systems. As such it is not a study of MIR capabilities or algorithms, but rather of the cogs and levers that together enable MIR *systems* to operate – of the effectiveness of our research processes and the scalability of MIR methods and data.

2. CHARACTERISING WORKFLOWS AND REUSE

To characterise reuse we draw on experience from the *scientific workflow systems* – tools that assist the composition and execution of computational or data manipulation steps. As a key tool for overcoming the issues of scale and usability associated with ad-hoc scripting when applied to data-driven science, Gil [9] identifies three requirements for assisted workflow composition: workflows described at **different levels of abstraction** to support varying degrees of reuse and repeatability; **expressive descriptions of workflow components** describing data input and output, constraints on interactions between components (interoperability), and relationships between alternate components; and **flexible workflow composition** mechanisms to assist the user in construction of complete executable flows. The principles of reuse and the deployment of scientific workflow systems go hand-in-hand: adherence to the latter encourages structured system design and interoperability, providing the principled framework within which the metadata and provenance required to support the former can be gathered.

Bechhofer *et al.* [1] go on to introduce seven characteristics required to satisfy reuse of the data and method that comprise an experimental workflow, capturing the motivations raised in the previous section through the notion of *Research Objects*: *(i)* **reuse** or redeployment as a whole or single "black box" entity; *(ii)* **repurposable** elements that can be reused independently of the whole; *(iii)* sufficient information describing data and method that the study is **repeatable**; *(iv)* the repeating of an experiment to **replicate** a result, bringing with it the need for comparability; *(v)* **replayable** examination of provenance of data and results (how they came to be); *(vi)* **referencable** and retrievable versions to support unambiguous citation of results; *(vii)* **revealable** provenance for auditing the integrity of the digitally captured data and method.

3. REUSABILITY OF MIR SYSTEMS

To inform our discussion of reuse within MIR we have studied many of the tools used across the community, examining publications, software documentation, and source code during our evaluation. There is a wide spectrum of purpose and architecture between these systems and as such direct implementation-level comparison becomes unwieldy and uninformative; rather, we make our judgement within the context characterised in Section 2, i.e. primarily with regard to reusability, workflow, and for interoperability.

We perform our comparison through the identification of what we have termed *realised abstractions,* summarising these for ten systems in Table 1 with further points of discussion within this Section.

A realised abstraction can take several forms: for a software library this might be a function or class definition, for a service a remote-procedure call or file serialization, or on the semantic web an ontology; but it must be, in some sense, a tangible resource that might be repurposed or called upon with or by other MIR software components. A realised abstraction is not synonymous with functionality implemented by the software: a framework or toolset might provide functionality in a manner completely practical and appropriate for its own use cases, but which is not recognised as a realised abstraction because we have been unable to identify a principled abstraction of the functionality that could be reused or that is suitable for interoperability. Neither is the study intended to be comprehensive – it is an illustrative sample of typical practice from across the community.

3.1 Implementation and scope

There is significant variety in the interaction by which a researcher or developer will reuse the provided functionality of the tools and systems in Table 1.

The **implementation environment** and **language** have a strong bearing on this. libXtract [3], for example, is a portable C library with Python and Java bindings providing feature extraction primitives, but requiring a developer to write the enacting skeleton of the software. jMIR [13] provides an extensible suite of components written in Java, while MIR Toolbox [11] and supporting toolboxes (Signal Processing, Auditory, Netlab and SOM) are written for the high-level MATLAB numerical computing environment. ChucK [22] is a programming language and environment using a time-based concurrent model designed with computer music in mind.

Some software provides a **framework** in which developers can structure reuse and extensions of existing code. Marsyas (C++ with Ruby, Python, and Java bindings) provides a comprehensive architecture for creating, managing, and visualising *dataflows* of audio, signal processing, and machine learning [20, 21]. sMIRk provides a toolkit of reusable functions for ChucK [8]. Once a developer has written a VAMP plugin (in C/C++; Python bindings available) it can be hosted and executed within the Sonic Annotator and Sonic Visualiser applications [4] – one such plugin exposes functionality from libXtract. The NEMA system [23] provides a language agnostic environment limited only by the Operating System and architecture of the underlying (virtual) machines: its framework uses the Meandre workflow system for distribution and execution of virtually any MIR algorithm (typically written using one of the other tools described here) and a Java-based data model for exchanging and consolidating inputs and outputs.

Scope of systems also varies, often depending on whether a general or specialised approach has been taken, and if it is operated as a stand-alone platform or in conjunction with other tools. Weka [10], for example, is a general purpose Java-based data-mining and machine learning toolset favoured within the MIR community for its experimentation environment and range of classifiers. AudioDB [18], on the other hand, is a specialised piece of database infrastructure for content-based similarity searches that relies upon the import of features extracted by other tools.

3.2 Reusable Method

At a basic level any piece of software with source code (or indeed machine code) can be considered reusable, along with the methods it embodies. In this study, we require more explicit recognition and encoding of concepts. In the first section of Table 1 we look for such realised abstractions representing MIR *methods* that are reusable and repurposable (and, for novel solutions, potentially referencable). Even when not developed for a workflow system we have also tried to identify the key characteristics of workflow components: different levels of abstraction, and explicit description of input, output, parameters, and interoperability. These are, of course, the same attributes that enable reuse at the level of a software library or development framework and which typically emerge from a principled software engineering effort to recognise the realised abstractions and encourage their reuse through implementation of, for example, a documented API.

Reusable MIR methods can be broadly grouped into three categories: signal processing derived *feature extraction*, within which we subdivide more deterministic *signal features* from less clearly defined *music features*; metric

	AudioDB [1]	ChucK & sMIRk	jMIR [2]	libXtract	M2K (inc. D2K)	Marsyas	Matlab & toolboxes [3]	NEMA/ Meandre	VAMP [4]	Weka
METHOD										
Signal Feature Extraction										
Basic signal		●		●	●	●	●		●	●
Basic maths		●			●	●	●			●
Basic filters		●		●	●	●	●			●
Envelopes and windowing	●	●			●	●	●			
Spectral distribution		●	●	●		●	●		●	
Error rate			●			●				
Power		●		●		●	●		●	
Transforms		●		●		●	●			
Linear Predictive Coding			●			●				
MFCC	●		●	●		●	●			
Music Feature Extraction										
Pitch	●			●		●	●			
Beat						●	●		●	
Correlation and Distance										
Correlation				●		●	●			
Distance	●					●	●			
Dimensional reducers	●					●				
Classification										
Predictive modelling		●			●	●				●
Regression					●	●				●
Clustering					●	●	●			●
Association Rule Learning					●					●
WORKFLOW										
Components		○	○		●	○	○	●	●	
Workflows		○	○		●	○	○	●	●	●
DATA EXCHANGE										
Abstract Signal		●		●		●		●	●	
Signal (values)		●				●	●	●	●	
Audio (playback, I/O)		●			●	●	●		●	
Abstract Feature	●	●	●					●	●	
Feature (values)	●	●	●			●		●	●	
Events / scheduling		●				●			●	
Abstract Classifier		●	●			●		●		
Classification (values)			●			●		●		
Aggregation (signal, feature)	●	○				●		●	●	
Annotation						●	●		●	

○ caveat described in Section 3. [1] including fftextract tool and AudioDB API library. [2] including jAudio, ACE, and ACE XML.
[3] including MIR toolbox, Signal Processing Toolbox, Auditory toolbox, Netlab toolbox, SOM toolbox.
[4] distribution including example plugins and Sonic Annotator.

Table 1: Presence of *Realised Abstractions* in MIR systems and tools.

based *correlation and distance* measures; and machine-learning based *classification*, which broadly includes any method taking as input features or distances and outputting item groupings. Coverage of these methods through realised abstractions varies widely between systems and is often a reflection of the intended scope and specialism of the tool: few have comprehensive coverage beyond a core competency, while others present no specialisation and rely on the ecosystem provided by their framework for method implementation, e.g. NEMA hosting of standalone algorithms, VAMP use of plugins, and toolboxes in Matlab. In these latter cases it also highlights a limitation of the survey, since including only a subset of extensions creates an artificial limit on methods unrepresentative of the tool's capabilities.

This highlights an opportunity for interoperable and replaceable *workflow components* when considering MIR systems as a single ecosystem, and starts to identify the grouping of methods for which expressive descriptions (Section 2) would be required to effect this process (a more comprehensive taxonomy of features, without the filter of realised abstractions, can be found in [15]).

3.3 Workflow

The second section of Table 1 appraises realised abstractions for the constituent parts of scientific workflow systems: the structure of workflows themselves, and the encapsulation of reusable components within them.

Several of the surveyed systems adopt a workflow approach in spirit: the dataflow and patching model at the core of Marsyas, and ACE (jMIR) Coordinator and Experimenter, provide facilities for chaining and adapting functionality but are strongly tied to their respective environments and do not easily generalise (Marsyas, for example, is tied to a synchronous tick model). MIRtoolbox follows a user centric procedural model with abstractions well suited to the MATLAB environment, but reflecting the process a (human) MIR researcher performs, rather than one that might map cleanly to a (machine-driven) workflow system.

Others tools embody more explicit examples of workflow technique: M2K [7] and NEMA build upon existing general purpose workflow environments (D2K and Meandre respectively) and their graphical management interfaces. However, with the exception of a genre classification proof of concept, NEMA has not made use of workflow components to encode a deconstructed method at the level described in the previous subsection, rather it utilises the distribution and scheduling features of the workflow systems when performing the MIREX evaluation. VAMP, a system designed for MIR but offering many traditional workflow system features, uses hosts such as Sonic Annotator which provide a flexible and extensible environment in which to compose and execute workflows consisting of VAMP plugin components. sMIRk and ChucK are also strongly workflow oriented, with their pervasive time-centric concurrent model providing ample illustration of how workflows can be applied across radically different approaches.

3.4 Data Exchange

Realised abstractions of specific methods and workflow elements can identify reuse within the bounds of a common environment (e.g. particular toolkit or software library). For reuse to occur *between* systems there must also be a mechanism for a mapping of method and workflow *between* systems, performed through some process of data exchange. To move beyond ad-hoc workflows components must be sufficiently described to support workflow composition. We have identified these higher-level concepts in the third section of Table 1 and, as in previous subsections, marked systems in which a realised abstraction correlates with the concept. The presence of a realised abstraction does not indicate an implementation of data exchange, merely that, within the software design, there is an explicit abstraction of the concept which could, in theory, form a basis for interoperability.

For Signals, Features, and Classifiers we highlight the need to represent both the abstract concept – required for flexible workflow composition and the provision of generic mechanisms for referencable and revealable reuse – and the values associated with an instance of that concept (signal input, feature data, classifier results) for repeatable and replayable reuse. The conceptual recognition of *events* and *scheduling* is also necessary for exchange of the temporal semantics often used in MIR applications. Aggregation of resources – be it collections of audio for analysis, computed features, or classified results – is a common requirement for scientific workflows systems and critical to systems interoperability, reuse (of data and results), and evaluation (including repeatability) in MIR. A particular facet of music, included here due to its common occurrence, is the explicit notion of exchange or playback of audio data.

The level at which the abstraction is found reflects the differing scope of the systems: for signal libXtract uses named pointers to data structures, whereas ChucK includes a sample primitive, and VAMP uses the Signal class from the Music Ontology [17]; for feature values Marsyas writes out from (the somewhat overloaded) realvec, jMIR defines a DataSet class, ChucK uses the (timesliced) unablob, while VAMP applies the Audio Features ontology. In all cases there is, if not a full model, a principled abstraction towards one.

Abstractions used for interoperability through *serialisation* of data to either file and network are a relevant subtopic. Serialisation can raise a number of requirements distinct from those considered purely for information modelling, including the reduction of parsing and transmission (size) overheads and the incorporation of mechanisms for efficient error checking. Several of the systems reviewed deploy abstractions designed with serialisation in mind, including ACE XML [14], the WEKA Attribute Relationship File Format (ARFF), and to a lesser extent the Audio Features Ontology used by VAMP. That these serialisations may not be optimal for data exchange *beyond* serialisation reinforces the need for varying levels of abstraction (Section 2) when building workflow systems – it is unlikely that a single abstraction will be appropriate for all operations.

4. REFLECTION

4.1 Reusable MIR: success or failure?

A superficial glance over Table 1 might highlight a significant level of duplication between MIR systems with an associated failure of reuse. This is not a failure. It is the mark of a strong and vibrant community that can support multiple toolkits catering to different preferences in development and deployment. There is no automatic benefit – nor apparent desire – to “standardise” on a single platform, toolkit, or programming language; indeed the rich variety of sophisticated software tailored to MIR specific problems indicates, if anything, the exact opposite.

Such a view would also overlook the successful software reuse exemplified in our study by libXtract, where a small well designed library with multi-language bindings has been reused by tools such as ChucK and VAMP. But more significantly, this would be a mischaracterisation of reuse which, as we have explored, goes beyond the redeployment and compatibility of source code.

4.2 Adoption of reuse

While our study has shown that no single MIR system provides comprehensive coverage across all notions of reuse, it also raises plentiful opportunities for systems that share common concepts to use these as a basis for abstraction and interoperability. Yet ISMIR proceedings indicate little cross-fertilization of most systems beyond the "home" lab and close collaborators. An explanation for this discrepancy might be the difference between the *potential* for reuse and the overhead of actual implementation: while we have highlighted the points at which there is conceptual alignment between systems, any implemented interoperability through the surveyed tools would require adoption of a software library, toolkit, or service, and the associated costs of building that interface.

At the level of an individual researcher selecting a tool, interoperability does not automatically follow reuse. The prevalence of Matlab – 52% of MIREX submissions in 2011 – demonstrates the preference for a familiar environment with a large body of basic methods, despite the lack of wider interoperability. Conversely, the authors of M2K believe the choice of Java was an unpopular one that limited uptake even through the system provided a workflow creation environment. In both cases the provision of interoperability, or the lack thereof, has not provided a sufficient motivation to override other preferences.

One approach, then, might be to lower interoperability overheads by switching from an "all or nothing" adoption model to something more akin to "pick and choose": selectively implementing interoperability where the benefits are clear and well scoped. Scientific workflow approaches can provide the principled framework to assist such conversion, exemplified at a technical level by the deployment of NEMA to run the MIREX evaluations: whilst wedded to a single implementation, the complexity of interoperability has been reduced to a single data abstraction appropriately selected and scoped for the evaluation and presentation of task results.

Another promising and flexible approach to reuse is the adoption of an agnostic modelling substrate upon which MIR specific abstractions can be developed. A prominent example of this is the use of RDF and other Semantic Web technologies in Sonic Annotator, VAMP plugins, and the the tools and ontologies they interoperate with and through. The use of a modelling layer that bridges into domains beyond MIR brings further benefits: the common model and distribution mechanism afforded by RDF and Linked Data can enable reuse and exchange of related data beyond that produced and consumed by the MIR system alone [16].

The uptake of Linked Data in industry and academia, including the scientific workflow and publishing communities, provides an opportunity to reuse and adapt tools and software developed elsewhere for similar purpose – although the burden and utility of adding compatible layers to MIR tools should not be overlooked. Nor, given its importance, should we ignore the task of selecting and scoping the appropriate level of abstraction for a model; it is not a panacea in itself, as evidenced by the lengthy gestation of standardised models such as MPEG-7.

4.3 Workflow centric research

We have presented our review of reuse within MIR through the lens of requirements originating in the scientific workflow community. We have seen that workflow systems are explicitly used as the basis for several MIR frameworks, and implicitly as an approach in others, however in both cases they are primarily employed for the distribution and scheduling of "black box" workflow components.

The increase in data driven science and the associated introduction of scientific workflow systems has led to a reflection on the nature of scientific method and its dissemination in a digital world – the question of how we can open these "black boxes". The principles for reuse outlined earlier in Section 3 are also the defining characteristics of a *Research Object* [1] – a semantically rich principled aggregation of resources bringing together the essential information relating to an experiment or investigation. This includes not only the data used, the methods employed to produce and analyse that data, but also the people involved in the investigation.

In our study of contemporary MIR systems we have surveyed for the principles of reuse, repurposing, and repeatability. While providing a foundation for data-driven research, it is when they are supplemented to encompass replication, replay, referencing and revealability that we see how the method and provenance captured by Workflow-centric Research Objects [2] can radically enhance the research environment and process.

By identifying realised abstractions for method, workflow, and data exchange in MIR systems we have demonstrated that the underlying conditions for Research Objects in MIR are already present: one can easily imagine a future in which MIREX entries are developed, submitted, evaluated and published as Research Objects.

5. CONCLUSIONS

Interoperability has not been – and should not be – achieved through the adoption of a single portal, toolkit, or programming language. Plurality of systems and the different approaches they embody is as important in avoiding skewed research and results as the plurality of datasets.

MIR embodies a process of digital research. While workflows provide a platform for principled reuse, they are also the building blocks for Research Objects, and through these the opportunity to conduct our research in new transparent, reusable, repurposable, and repeatable ways. In this paper we have demonstrated MIR is well positioned to take advantage of these approaches.

Workflows and Research Objects can provide a framework, but as a community we must define the levels of reuse and interoperability we wish to achieve through them. This does not imply a single level of abstraction nor an associated single level of modularised software, but multiple models appropriate to each task at hand. As the survey in this paper has shown, the basis for these encapsulations already exists at different levels within MIR systems.

Adopting a "pick and choose" approach to reuse, the identification of boundary objects [19] – points of shared understanding through standardised method and translation between viewpoints – may prove helpful. So too can MIREX as a process through which the community must reach consensus regarding tasks and output – and where the benefits of reuse might be most keenly felt. In this context we suggest a first step should be taken at the data level: describing and exchanging input, output, and parameters using community agreed vocabularies encoded in RDF.

6. ACKNOWLEDGEMENTS

This work was carried out through the EPSRC funded *e-Research South* platform grant (Grant No. EP/F05811X/1) and the *Structural Analysis of Large Amounts of Musical Information* project funded by the JISC Digitisation and e-Content programme and the National Science Foundation (Grant Nos. IIS 10-42727 and IIS 09-39253). We are also grateful for the helpful comments from reviewers of earlier drafts of the paper.

7. REFERENCES

[1] S. Bechhofer, I. Buchan, D. De Roure, P. Missier, et al. Why linked data is not enough for scientists. *Future Generation Computer Systems*, In press/online. http://dx.doi.org/10.1016/j.future.2011.08.004.

[2] K. Belhajjame, O. Corcho, D. Garijo, J. Zhao, et al. Workflow-centric research objects: First class citizens in scholarly discourse. In *Proc. Workshop on the Semantic Publishing (SePublica)*, pages 1–12, 2012.

[3] J. Bullock. Libxtract: A lightweight library for audio feature extraction. In *Proc. International Computer Music Conference*, pages 25–28, 2007.

[4] C. Cannam, C. Landone, M. Sandler, and J.P. Bello. The sonic visualiser: A visualisation platform for semantic descriptors from musical signals. In *Proc. 7th International Conference on Music Information Retrieval*, pages 324–327, 2006.

[5] M.A. Casey, R. Veltkamp, M. Goto, M. Leman, et al. Content-based music information retrieval: current directions and future challenges. *Proc. IEEE*, 96(4):668–696, 2008.

[6] J.S. Downie, D. Byrd, and T. Crawford. Ten years of ISMIR: Reflections on challenges and opportunities. In *Proc. 10th International Society for Music Information Retrieval Conference*, pages 13–18, 2009.

[7] J.S. Downie, A.F. Ehmann, and X. Hu. Music-to-knowledge (M2K): a prototyping and evaluation environment for music digital library research. In *Proc. 5th ACM/IEEE Joint Conference on Digital Libraries*, pages 376–376, 2005.

[8] R. Fiebrink, G. Wang, and P. Cook. Support for MIR prototyping and real-time applications in the chuck programming language. In *Proc. 9th International Conference of Music Information Retrieval*, pages 153–158, 2008.

[9] Y. Gil. Workflow composition: Semantic representations for flexible automation. *Workflows for e-Science*, pages 244–257, 2007.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, et al. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[11] O. Lartillot and P. Toiviainen. MIR in Matlab (II): A toolbox for musical feature extraction from audio. In *Proc. 8th International Society of Music Information Retrieval Conference*, pages 127–130, 2007.

[12] D. McEnnis, C. McKay, and I. Fujinaga. Overview of OMEN. In *Proc. International Conference on Music Information Retrieval*, pages 7–12, 2006.

[13] C. McKay. *Automatic music classification with jMIR*. PhD thesis, McGill University, 2010.

[14] 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. In *Proc. International Society for Music Information Retrieval Conference*, pages 303–8, 2009.

[15] D. Mitrović, M. Zeppelzauer, and C. Breiteneder. Features for content-based audio retrieval. *Advances in Computers*, 78:71–150, 2010.

[16] K. R. Page, B. Fields, B .J. Nagel, G. O'Neill, et al. Semantics for music analysis through linked data: How country is my country? In *Proc. IEEE Sixth International Conference on e-Science*, pages 41–48, 2010.

[17] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson. The music ontology. In *Proc. International Conference on Music Information Retrieval*, pages 417–422, 2007.

[18] C. Rhodes, T. Crawford, M. Casey, and M. d'Inverno. Investigating music collections at different scales with audiodb. *Journal of New Music Research*, 39(4):337–348, 2010.

[19] S.L. Star and J.R. Griesemer. Institutional ecology, translations and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. *Social studies of science*, 19(3):387–420, 1989.

[20] G. Tzanetakis and P. Cook. Marsyas: A framework for audio analysis. *Organised sound*, 4(3):169–175, 1999.

[21] G. Tzanetakis, L.G. Martins, L.F. Teixeira, C. Castillo, R. Jones, and M. Lagrange. Interoperability and the marsyas 0.2 runtime. In *Proc. International Computer Music Conference*, 2008. http://hdl.handle.net/2027/spo.bbp2372.2008.149.

[22] Ge Wang. *The ChucK Audio Programming Language A Strongly-timed and On-the-fly Environ/mentality*. PhD thesis, Princeton University, 2008.

[23] K. West, A. Kumar, A. Shirk, G. Zhu, et al. The networked environment for music analysis (NEMA). In *Proc. 6th IEEE World Congress on Services*, pages 314–317, 2010.

MULTI-TEMPLATE SHIFT-VARIANT NON-NEGATIVE MATRIX DECONVOLUTION FOR SEMI-AUTOMATIC MUSIC TRANSCRIPTION

Holger Kirchhoff, Simon Dixon, Anssi Klapuri
Queen Mary University of London, Centre for Digital Music
{holger.kirchhoff, simon.dixon, anssi.klapuri}@eecs.qmul.ac.uk

ABSTRACT

For the task of semi-automatic music transcription, we extended our framework for shift-variant non-negative matrix deconvolution (svNMD) to work with multiple templates per instrument and pitch. A k-means clustering based learning algorithm is proposed that infers the templates from the data based on the provided user information. We experimentally explored the maximum achievable transcription accuracy of the algorithm and evaluated the prospective performance in a realistic setting. The results showed a clear superiority of the Itakura-Saito divergence over the Kullback-Leibler divergence and a consistent improvement of the maximum achievable accuracy when each pitch is represented by more than one spectral template.

1. INTRODUCTION

Automatic music transcription describes the process of transforming a recording of a piece of music into a score or an intermediate score-like representation. It has been an active area of research over the last decades and a multitude of approaches has been proposed. An overview of the main computational techniques for music transcription can be found in [1]. Despite this long research history, the accuracy of fully automatic music transcription systems is still considerably below the accuracy achieved by trained musicians.

As a step towards a more accurate transcription system, we address the task of *user-assisted* or *semi-automatic music transcription*. These terms refer to systems in which the user provides a certain amount of information about the recording under analysis which can then be used to guide the transcription process. In this paper, we assume that the user labels a certain number of notes for each instrument, which is then used to build instrument models that are tailored to the specific instruments in the mixture. In a practical application, the user could either be presented with a magnitude spectrogram and be asked to graphically mark a few fundamental frequency trajectories, or — if a more musical approach is desired — with the result of a fully-automatic transcription system for which he is asked to assign some of the detected notes to the instruments.

We address this task by means of a non-negative matrix deconvolution framework. Since the introduction of non-negative matrix factorisation (NMF) [2] which was first applied to music analysis by Smaragdis and Brown [3], a number of modifications to this algorithm have been proposed. In this work, we build on our shift-variant non-negative matrix deconvolution (svNMD) framework [4] which is itself a modification of Schmidt and Mørup's NMF2D model [5]. In the svNMD framework, a single spectral template for each pitch of each instrument is estimated which is then used to detect fundamental frequencies in the constant-Q magnitude spectrogram of the recording. Here, we extend the model to work with multiple templates per pitch. The motivation for having multiple templates per pitch is given by the fact that the spectral shape of a particular note can vary based on dynamics or playing style and to model a time-varying spectral envelope of a note.

Other related work can be found in NMF-based approaches to score-informed source separation, where mid-level score representations are used to infer models for the source instruments. Hennequin et al. [6] modify the NMF model to work with parametric spectral templates. The model allows templates to be shifted in frequency while preserving the overtone amplitudes. The parameters are learned by initialising the NMF gain matrix and successively applying update functions for the template parameters and the gains. In [7], Ganseman et al. use a synthesised and time-aligned score as priors for the PLCA system proposed in [8]. In addition to note information, this approach requires knowledge about the timbre of each source in order to facilitate a fast convergence.

The remainder of this paper is organised as follows: In the following section we present our extension to the svNMD framework that works with multiple templates per pitch (Sect. 2.1) and illustrate the algorithm for learning these templates (Sect. 2.2). In Sect. 3 we evaluate the proposed algorithm in two different experiments and discuss the results. Conclusions are finally drawn in Sect. 4.

2. MULTIPLE-TEMPLATE SHIFT-VARIANT NON-NEGATIVE MATRIX DECONVOLUTION

In this section we present our non-negative matrix deconvolution framework which decomposes a constant-Q spectrogram into a structured dictionary of instrument templates and corresponding gain values (see Sect. 2.1). The framework represents each pitch of each instrument by a predefined number of spectral templates. Furthermore, in Sect. 2.2 we describe a procedure that allows us to extract multiple templates for each note previously labelled by the user. This procedure is applicable to polyphonic material where partials might overlap.

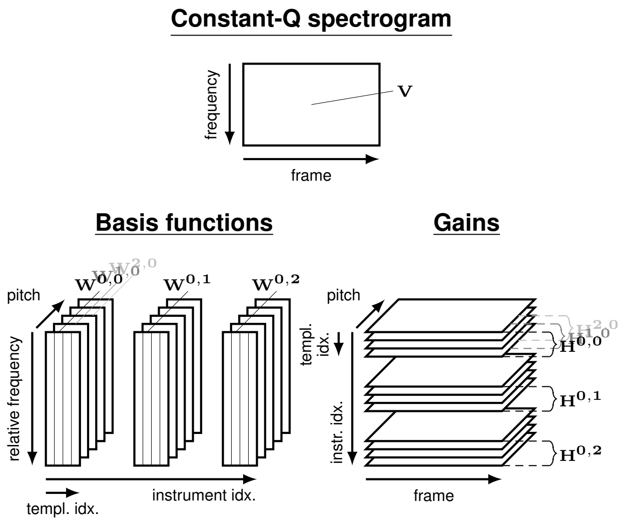


Figure 1: svNMD framework with multiple templates per instrument and pitch.

2.1 Framework

The proposed non-negative matrix deconvolution framework decomposes a constant-Q spectrogram into 4-dimensional structures for the basis functions and the gains, respectively. Figure 1 illustrates the framework graphically. Each instrument in the mixture under analysis is represented by a 3-dimensional structure (tensor) that contains a fixed number of basis functions for each pitch. The pitch resolution is determined by the frequency resolution of the constant-Q spectrogram under analysis and the number of templates per pitch can be chosen arbitrarily. Likewise, for each instrument a 3-dimensional structure contains the corresponding gains for the spectral templates. Each layer displayed on the right-hand side of Fig. 1 contains the gain trajectories at a fixed template index over time. In order to arrive at a single pianoroll-like representation for each instrument, the gains of the layers can be summed up vertically.

In mathematical terms, we denote the constant-Q magnitude spectrogram by $\mathbf{V} \in \mathcal{R}_+^{N\times M}$, where N is the number of frequency bins and M the number of frames. The matrix $\mathbf{W}^{\phi,i} \in \mathcal{R}_+^{N\times T}$ contains in its columns the spectral templates of instrument i at pitch ϕ (see Fig.1). T denotes the specified number of spectral templates. All templates have their first partial at the first row index of $\mathbf{W}^{\phi,i}$ and likewise all other partials appear each roughly at their corresponding row index due to the use of the constant-Q spectrogram. $\mathbf{H}^{\phi,i} \in \mathcal{R}_+^{T\times M}$ on the other hand denotes the matrix that contains the corresponding gains for the templates of instrument i at pitch ϕ over time. Note that in Fig. 1, this matrix corresponds to a slice through one of the banks of layers, as shown in the figure.

Given these matrices we approximate our original spectrogram $\mathbf{V}$ by

$$\mathbf{V} \approx \mathbf{\Lambda} = \sum_{i=0}^{I-1}\sum_{\phi=0}^{\Phi-1} \overset{\phi\downarrow}{\mathbf{W}^{\phi,i}}\mathbf{H}^{\phi,i}, \tag{1}$$

where $\mathbf{\Lambda} \in \mathcal{R}_+^{N\times M}$ has the same dimensions as $\mathbf{V}$. Here, I denotes the number of instruments in the mixture and Φ the number of pitches. Φ and N do not necessarily need to be the same, in our case, however, they are. The operator $\phi\downarrow$ denotes a downward shift of the matrix elements by ϕ rows while the upper ϕ rows are filled with zeros. This mixture model shifts each spectral template to the correct frequency position and scales them by the corresponding gains at each frame.

Update equations were derived for both $\mathbf{W}^{\phi,i}$ and $\mathbf{H}^{\phi,i}$ by computing the gradient of the β-divergence between $\mathbf{V}$ and $\mathbf{\Lambda}$. The β-divergence is given by

$$C_\beta = \sum_{n=1}^{N}\sum_{m=1}^{M} \frac{[\mathbf{V}]_{n,m}^\beta}{\beta(\beta-1)} + \frac{[\mathbf{\Lambda}]_{n,m}^\beta}{\beta} - \frac{[\mathbf{V}]_{n,m}[\mathbf{\Lambda}]_{n,m}^{\beta-1}}{\beta-1}, \tag{2}$$

for $\beta \in \mathcal{R}\backslash\{0,1\}$ and

$$C_0 = \sum_{n=1}^{N}\sum_{m=1}^{M} \frac{[\mathbf{V}]_{n,m}}{[\mathbf{\Lambda}]_{n,m}} - \log\left(\frac{[\mathbf{V}]_{n,m}}{[\mathbf{\Lambda}]_{n,m}}\right) - 1 \tag{3}$$

$$C_1 = \sum_{n=1}^{N}\sum_{m=1}^{M} [\mathbf{V}]_{n,m}\log\left(\frac{[\mathbf{V}]_{n,m}}{[\mathbf{\Lambda}]_{n,m}}\right) + [\mathbf{\Lambda}]_{n,m} - [\mathbf{V}]_{n,m}\,. \tag{4}$$

The update equations are given by

$$\mathbf{W}^{\phi,i} \leftarrow \mathbf{W}^{\phi,i} \bullet \frac{\left(\overset{\phi\uparrow}{\mathbf{V}} \bullet \overset{\phi\uparrow}{\mathbf{\Lambda}^{\beta-2}}\right)\left[\mathbf{H}^{\phi,i}\right]^T}{(\overset{\phi\uparrow}{\mathbf{\Lambda}^{\beta-1}})\,[\mathbf{H}^{\phi,i}]^T} \tag{5}$$

$$\mathbf{H}^{\phi,i} \leftarrow \mathbf{H}^{\phi,i} \bullet \frac{\left[\overset{\phi\downarrow}{\mathbf{W}^{\phi,i}}\right]^T\left(\mathbf{V}\bullet\mathbf{\Lambda}^{\beta-2}\right)}{\left[\overset{\phi\downarrow}{\mathbf{W}^{\phi,i}}\right]^T\mathbf{\Lambda}^{\beta-1}} \tag{6}$$

In these equations, $\bullet$ denotes an elementwise multiplication and all divisions and power operations are likewise carried out per element. We can obtain the well-known least squares (LS), Kullback-Leibler (KL) and Itakura-Saito (IS) cost functions by setting $\beta = 2$, $\beta = 1$ and $\beta = 0$, respectively. The derivation of Eqs. 5 and 6 is provided in a supplementary document [9].

2.2 Learning the basis functions

Figure 2 illustrates the iterative procedure of learning a number of templates for a single note labelled by the user. The user provides information about the start frame, the end frame and the pitch ϕ_0 of a note of a particular instrument i_0. This information can be illustrated by a pianoroll that contains a single line representing the note, as shown on the left-hand side of panel (a). Given this information, we can identify the matrix $\mathbf{W}^{\phi_0,i_0}$ in which the learned templates will be stored and the matrix $\mathbf{H}^{\phi_0,i_0}$ that contains the gains for each of the templates over time (grey-shaded matrices on the right-hand side of panel (a)). Since only those two matrices $\mathbf{W}^{\phi_0,i_0}$ and $\mathbf{H}^{\phi_0,i_0}$ are relevant for learning the templates from the labelled note, we isolate them from their tensors when illustrating the learning algorithm in panels (b)–(f).

Panels (b)–(f) display the algorithmic steps for estimating the spectral templates. This procedure is in fact very similar to applying *k-means clustering* to the spectra of a note at all time frames within the spectrogram $\mathbf{V}$. In this analogy, each spectral template corresponds to a cluster mean and thus represents a set of spectra at different time frames. Since the learning procedure is carried out within the nonnegative framework, the correponding k-means clustering steps might not be obvious. For that reason, we illustrate these on the right hand side of panels (b)–(f). In these graphs, each data point corresponds to a spectrum of the note at a particular time frame in the N-dimensional space which is here for the sake of illustration reduced to 2 dimensions.

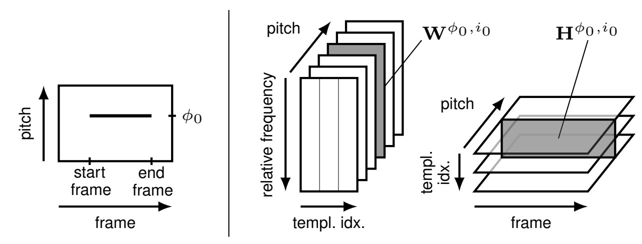
1. **Initialisation:** The algorithm starts by initialising the spectral templates in $\mathbf{W}^{\phi_0,i_0}$ with nonnegative random values (panel (b)). In the gain matrix $\mathbf{H}^{\phi_0,i_0}$ each frame of the note is randomly assigned to exactly one spectral template by setting the corresponding gains to a value of 1 while all other entries of the matrix are set to 0. In the k-means example, this corresponds to assigning the data points randomly to one of the three clusters: crosses, circles and squares.

2. **Update:** In the second step (panel (c)), we update the spectral templates in $\mathbf{W}^{\phi_0,i_0}$ based on the gains that were set in the previous step. This modifies the spectral templates in such a way that the resulting templates minimise the β-divergence at the assigned frames. Thus, each resulting spectral template can be seen as an average of the instrument spectra at the time frames that were assigned to it. In k-means clustering terms, this is equivalent to computing the average of the data points that were assigned to the same class. Note that in order to eliminate scale-ambiguities in the nonnegative framework, all spectral templates in $\mathbf{W}^{\phi_0,i_0}$ are scaled to have a power of 1 and the gains are adjusted accordingly.

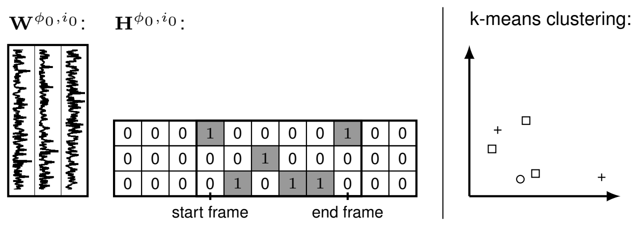
3. **Assignment:** In order to assign the spectra of the note at all frames to the template that best resembles their spectral shape, we set the template gains at each note frame to equal values (panel (d)) and update the gains based on the given spectral templates (panel (e)). This way, the gain matrix contains the contributions of each template to the audio spectra of each time frame when linearly combining the templates. This can be seen as a similarity measure between the templates and the spectra. We assign each frame to the template with the highest gain value, here indicated by the grey-shaded entries. In the k-means clustering example, this corresponds to the assignment step, in which each data point is assigned to the closest mean. We setup a new matrix $\mathbf{H}^{\phi_0,i_0}$ (panel (f)) that contains at each frame and each assigned template index the gains from step 2 (cf. panel (d)).

The algorithm iterates over steps 2 and 3.

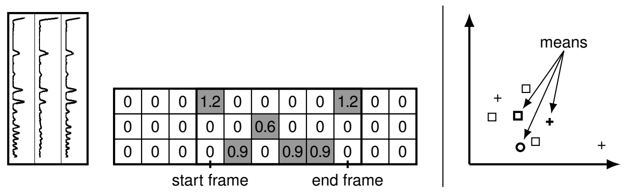
The reason for assigning each frame to just a single spectral template in step 1 and 3 is that we want to avoid the partials of a note to be split among the different templates. A template that only contains a subset of partials might be used by the algorithm to explain partials of other notes from the same or another instrument. An intuitive example for this case would be a spectral template that only contains a single partial (i.e. a single spectral peak) which can be used by the algorithm to approximate a partial of any note



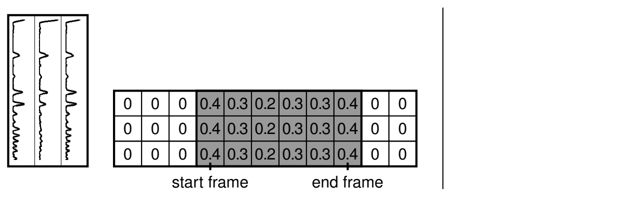
(a) Piano roll and svNMD framework



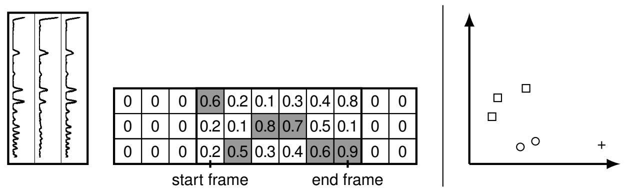
(b) Initialisation



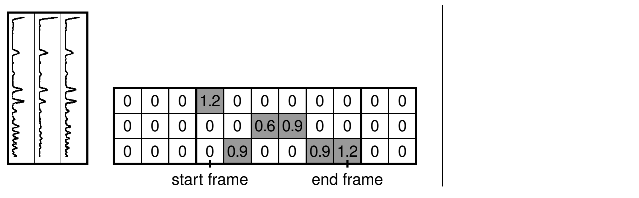
(c) Update



(d) Assignment (1)



(e) Assignment (2)



(f) Assignment (3)

Figure 2: Learning algorithm

at that position of the same or another instrument. This would produce a gain value either at the wrong fundamental frequency or the wrong instrument or both and thereby adulterate the transcription accuracy.

In k-means clustering, there is a chance of producing empty clusters when assigning the data points to the new means. The same problem applies to our proposed learning algorithm. In our algorithm this problem can occur in panel (e), when for a certain template none of the frames contains the largest gains. In this case, we detect the largest cluster (i.e. the template with the largest number of assigned frames) and randomly assign half of its frames to the empty cluster. The spectral template of the empty cluster is then discarded and replaced by a duplicate of the spectral template of the largest cluster.

Although the learning procedure was here illustrated by an individual note of a single instrument, the procedure is applicable to and intended for polyphonic audio. A MATLAB implementation of the learning algorithm is available at `http://code.soundsoftware.ac.uk/projects/svnmdmt`.

3. EVALUATION

The evaluation of the proposed framework and the template learning algorithm was carried out in two experiments. In the first experiment (Sect. 3.3) we explored the upper limit of performance of the algorithm when used for semi-automatic transcription. The results of this experiment provide some intuition about the potential of the framework to accurately approximate a spectrogram. The second experiment (Sect. 3.4) looked at a more realistic semi-automatic transcription setting in which only a part of the notes are employed for learning the templates which are then applied to transcribe the remainder of the recording.

3.1 Dataset

For both experiments described below, the same dataset as in [4] was used. The dataset was based on monophonic recordings of musical phrases from 12 different instruments, each with a length of approximately 30s. Mixtures of 2 to 5 instruments were produced by combining the monophonic signals. For each polyphony level (2 to 5 instruments), 50 different combinations were generated. At the same time, the hand-annotated notes of the 12 monophonic files were available in MIDI format. Those MIDI files acted as the ground truth for the evaluation.

In addition to that, we evaluated the algorithm on more harmonically related instrument parts and computed results for a wind quintet excerpt (cf. [10]). This example had a length of 54s and for each instrument part hand-annotated MIDI ground-truth was available.

3.2 Accuracy

In order to measure the transcription accuracy, we refrained from using the common measures *precision*, *recall* or *F-score*. Those measures are used to compare detected note events to ground truth notes. Combining gains into note objects, however, would require a subsequent note-tracking algorithm which will have an influence on the results. Our aim is here to study the performance of the proposed algorithm in isolation.

As an accuracy measure, we therefore compute the percentage of energy in the gain matrices that is concentrated in the ground truth fundamental frequencies. This is done for each instrument individually. In order to achieve that, a summary gain matrix $\mathbf{G}^i$ is computed for each instrument i in the mixture by

$$\left[\mathbf{G}^i\right]_{\phi,n} = \sum_{t=1}^{T} \left[\mathbf{H}^{\phi,i}\right]_{t,n}. \tag{7}$$

Intuitively, in Fig. 1 this corresponds to summing all the displayed gain layers for each instrument. Based on the summary gain matrices $\mathbf{G}^i$, the per-instrument accuracies Acc_i are computed by

$$\mathrm{Acc}_i = \frac{\sum_{n=1}^{N} \sum_{\phi \in \mathcal{F}_n} \left(\left[\mathbf{G}^i\right]_{\phi,n}\right)^2}{\sum_{n=1}^{N} \sum_{\phi'=1}^{\Phi} \left(\left[\mathbf{G}^i\right]_{\phi',n}\right)^2}. \tag{8}$$

In this equation, $\mathcal{F}_n$ denotes the set of frequency bins of the annotated pitches in the n-th frame. Since the test set only contains monophonic instruments, $\mathcal{F}_n$ only contains the bins of at most one note at each time frame. Ideally, we would like to see all energy concentrated in the fundamental frequencies which would make it easy to detect notes within the gain matrices. This case would correspond to an accuracy Acc_i of 1.

3.3 Experiment 1: Exploring the upper performance limit

In the first experiment we explored the upper performance limit of the nonnegative framework when used for a semi-automatic transcription task. The upper performance limit is given when a user labels *all* notes of *all* instruments in the mixture under analysis. Although this scenario may seem trivial, because no transcription algorithm would be required if all notes were known beforehand, this evaluation provides an intuition about the expressivity of the algorithm and reveals any methodological flaws.

3.3.1 *Experimental setup*

For each file in the dataset, we extracted $T = 1$, 3 and 5 templates per pitch, by running 50 iterations of the template learning algorithm described in Sect. 2.2. The user information was given by the ground truth MIDI files of the instruments contained in the mixture which contained onset, offset and pitch information of the notes of the instruments. Once the basis functions were learned from the constant-Q magnitude spectrogram of the recording, the gain matrices were computed. This was done by randomly initialising all matrices $\mathbf{H}^{\phi,i}$ with nonnegative values and applying 10 iterations of the update equation for the gains (Eq. 6). Transcription accuracies were computed as described in Sect. 3.2. The experiment was conducted for the IS-divergence ($\beta = 0$) and the KL-divergence ($\beta = 1$).

3.3.2 *Results*

The results of this experiment are displayed in Fig. 3. The upper panels display the results obtained by using the Itakura-Saito (IS) divergence, the lower panels the results of the Kullback-Leibler (KL) divergence. From left to right, the panels show the results of the different polyphony levels — from 1 to 5 instruments — and on the right-hand side the results of the wind quintet. In each panel, we compare the per-instrument transcription accuracies of all instruments of all files when represented with different numbers of templates per pitch. The results are displayed as box-plots: the upper and lower edges of the box represent the first (Q_1) and third quartile (Q_3), the median is displayed in between. The whiskers extend to the data points that

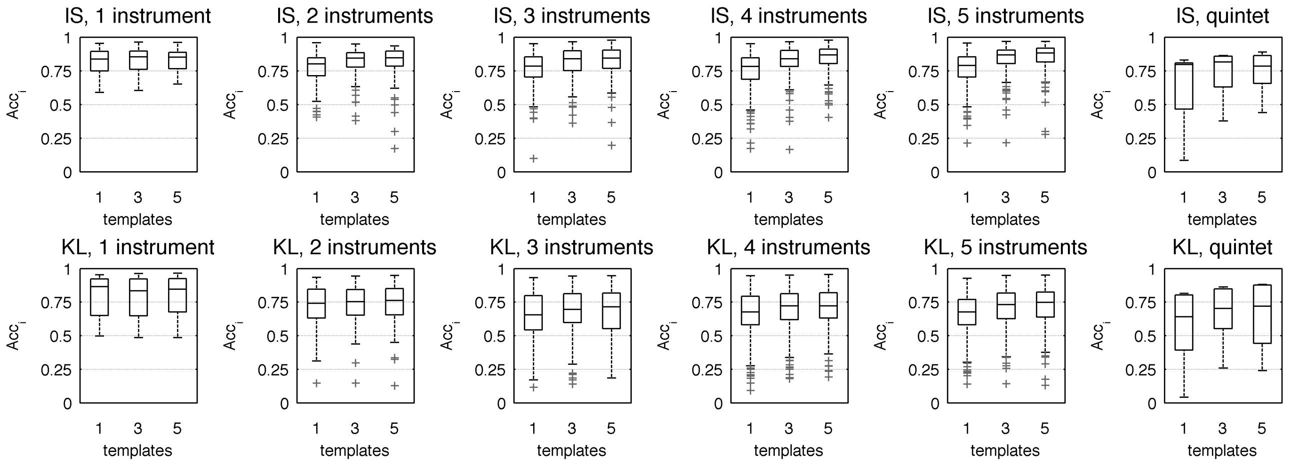


Figure 3: Results of experiment 1. The upper and lower rows display the per-instrument accuracies for the IS-divergence and KL-divergence, respectively. From left to right, the panels contain the accuracies for different polyphony levels and for the wind quintet. Within each panel the results for different numbers of templates per pitch are presented as boxplots.

are furthest away from the median, but within the interval $[Q_1 - 1.5 \cdot (Q_3 - Q_1) \ldots Q_3 + 1.5 \cdot (Q_3 - Q_1)]$. All data points outside that range are marked by crosses and considered as *outliers*.

When comparing the different cost functions for the random instrument mixtures, it becomes obvious that the Itakura-Saito divergence outperforms the Kullback-Leibler divergence in all cases. A possible explanation for the good performance of the IS-divergence is its scale-invariance property (cf. [11]) which is in compliance with Weber's law applied to the perception of loudness. An interesting aspect we found here is that by using the IS-divergence, the accuracies do not even noticeably decay when the number of instruments is increased.

When we compare the results for different numbers of spectral templates per pitch, a clear tendency towards higher accuracies can be observed when more templates are learned for each note. The improvement is consistent when the number of templates is increased from 1 to 3 and ranges between 2% and almost 10% for different poyphony levels when considering the median accuracies for the IS-divergence. Increasing the number of templates from 3 to 5 improves the accuracy even further, but not in the same consistent way as from 1 to 3.

The results of the wind quintet generally confirm the above findings, particularly the increasing accuracy when multiple templates are used. The median accuracy is however slightly lower than for the data set of random instrument mixtures, which can be attributed to the larger number of overlapping partials.

3.4 Experiment 2: Real case scenario

In the second experiment, we estimated the performance of a semi-automatic transcription system in a more realistic environment. We assumed that the user had labelled a certain number of notes for each instrument, which we use to estimate template spectra at the corresponding pitches. These template spectra are then used to build complete models for the instruments which are then applied to the remainder of the piece in order to obtain the transcription.

3.4.1 Experimental setup

For this experiment, we split each file in the dataset in two halves, each containing approx. 15 s of audio. We assumed that the user had labelled all notes of all instruments in the first half and used these to learn the basis functions as described above. The basis functions were then replicated at the surrounding pitches to cover the whole pitch range and were applied to estimate the gains of the second half of the audio.

As in the first experiment, we applied all combinations of cost functions (IS-divergence and KL-divergence), number of instruments (1–5) and number of templates per pitch (1,3 and 5). We again ran 50 iterations of the learning algorithm and 10 iterations for the estimation of the gain matrices.

3.4.2 Results

Figure 4 shows the results for the second experiment. The order of the results is the same as for the previous results.

For the random instrument mixtures, the results of this experiment differ from the results of the previous experiment. In general, there is a considerably larger variance in the results for each configuration. Several trends are clearly visible in the diagram: For both cost functions, the accuracy decreases when the number of instruments in the mixture is increased. The impression from the first experiment that the IS-divergence generally yields better results than the KL-divergence is here confirmed, the only exception being the polyphony level of one instrument. However, since the results for the monophonic audio files are only based on 12 accuracies, this fact needs to be put in perspective.

In terms of the different numbers of templates per pitch, the results for 1, 3 and 5 templates consistently stay in the same range and no clear trend can be found. It has to be considered here that the results of this experiment are not only influenced by the number of templates, but also by the fact that templates of non-annotated pitches were estimated by replicating adjacent pitches. It seems that the error introduced by this rough assumption outweighs the gain of having multiple templates per pitch.

The results for the quintet recording only show a small loss in accuracy to the previous experiment. The reason for

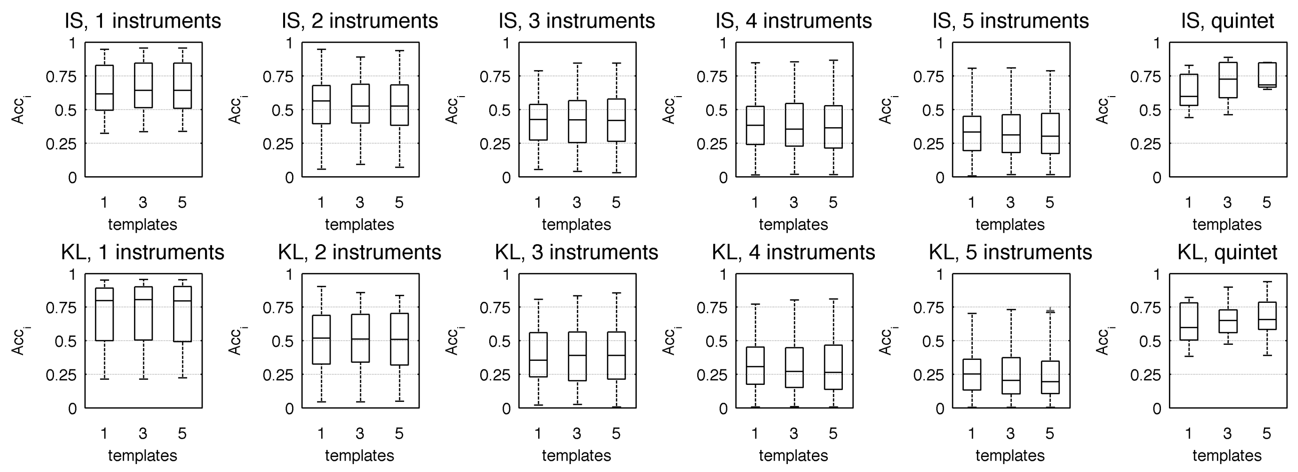


Figure 4: Results of experiment 2. The results are displayed in the same order as the ones in Fig. 3.

this can be seen in the fact that in this excerpt large parts of the first half are repeated in the second half, so that almost the same pitch range was covered for training and testing.

The experiments show a certain discrepancy between the maximum achievable accuracy and the accuracies that can be expected in a more realistic setting. There are several explanations for the fact that the accuracy of the second experiment is decreased: First, there was twice more training data in experiment 1. Second, in the first experiment the basis functions will have been better adjusted to the spectra of the second half of the audio files, which were not used in the learning process in the second experiment. And third, as indicated above, filling the gaps in the basis function tensors by merely replicating the estimated basis functions in the second experiment leads to a loss in accuracy.

4. CONCLUSION

We presented a shift-variant non-negative matrix deconvolution (svNMD) framework that represents each note of each instrument by multiple spectral templates. A learning algorithm was presented that allows the different templates to be estimated within the svNMD framework. The steps of this algorithm are comparable to a k-means clustering algorithm. We investigated the use of the framework for the task of semi-automtic music transcription in which the user provides a priori information about some notes in the mixture under analysis. Two experiments were carried out. In the first experiment, the upper performance limit of the algorithm was investigated which is given when the user provides information about all notes of all instruments. The results showed the superiority of the IS-divergence over the KL-divergence and a consistent improvement when more than one template per pitch was used. The second experiment expoited a more realistic use case in which the user merely labels a subset of the notes. Here, the superiority of the IS-divergence could be confirmed. In this experiment, however, no improvement could be found by using multiple templates per pitch.

5. REFERENCES

[1] A. Klapuri and M. Davy, ed.: *Signal Processing Methods for Music Transcription*, Springer, New York, 2006.

[2] D. D. Lee and H. S. Seung: "Learning the parts of objects by non-negative matrix factorization," *Nature*, Vol. 401, Nr. 6755, pp. 788-–791, 1999.

[3] P. Smaragdis and J.C. Brown: "Non-negative matrix factorization for polyphonic music transcription," *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 177–180, 2003.

[4] H. Kirchhoff, S. Dixon, and A. Klapuri: "Shift-variant non-negative matrix deconvolution for music transcription," *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2012.

[5] M. Schmidt and M. Mørup: "Nonnegative matrix factor 2-D deconvolution for blind single channel source separation," *6th International Conference on Independent Component Analysis and Blind Source Separation*, pp. 700–707, Charleston, USA, 2006.

[6] R. Hennequin, B. David, and R. Badeau: "Score informed audio source separation using a parametric model of non-negative spectrogram,". *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 45-–48, 2011.

[7] J. Ganseman, G. J. Mysore, J. S. Abel, and P. Scheunders: "Source separation by score synthesis," *International Computer Music Conference*, pp. 462–465, 2010.

[8] P. Smaragdis and G. J. Mysore: "Separation by humming: User-guided sound extraction from monophonic mixtures," *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, USA, 2009.

[9] H. Kirchhoff, S. Dixon, and A. Klapuri: "Derivation of update equations for multiple-template shift- variant non-negative matrix deconvolution based on β-divergence," Tech. Rep. C4DM-TR-06-12, Queen Mary University of London, 2012, `http://www.eecs.qmul.ac.uk/~holger/C4DM-TR-06-12`.

[10] E. Benetos and S. Dixon: "Joint multi-pitch detection using harmonic envelope estimation for polyphonic music transcription," *IEEE Journal of Selected Topics in Signal Processing*, Vol. 5, No. 6, pp. 1111–1123, 2011.

[11] C. Févotte, N. Bertin, and J. L. Durrieu: "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural Computation*, Vol. 21, No. 3, pp. 793–830, 2009.

TRACKING MELODIC PATTERNS IN FLAMENCO SINGING BY ANALYZING POLYPHONIC MUSIC RECORDINGS

A. Pikrakis
University of Piraeus, Greece
pikrakis@unipi.gr

F. Gómez, S. Oramas
Polytechnic University of Madrid, Spain
{fmartin, soramasm}@eui.upm.es

J. M. D. Báñez, J. Mora, F. Escobar
University of Sevilla, Spain
{dbanez, mora, fescobar}@us.es

E. Gómez, J. Salamon
Universitat Pompeu Fabra, Spain
{emilia.gomez, justin.salamon}@upf.edu

ABSTRACT

The purpose of this paper is to present an algorithmic pipeline for melodic pattern detection in audio files. Our method follows a two-stage approach: first, vocal pitch sequences are extracted from the audio recordings by means of a predominant fundamental frequency estimation technique; second, instances of the patterns are detected directly in the pitch sequences by means of a dynamic programming algorithm which is robust to pitch estimation errors. In order to test the proposed method, an analysis of characteristic melodic patterns in the context of the flamenco fandango style was performed. To this end, a number of such patterns were defined in symbolic format by flamenco experts and were later detected in music corpora, which were composed of un-segmented audio recordings taken from two fandango styles, namely Valverde fandangos and Huelva capital fandangos. These two styles are representative of the fandango tradition and also differ with respect to their musical characteristics. Finally, the strategy in the evaluation of the algorithm performance was discussed by flamenco experts and their conclusions are presented in this paper.

1. INTRODUCTION

1.1 Motivation and context

The study of characteristic melodic patterns is relevant to the musical style and this is especially true in the case of oral traditions that exhibit a strong melodic nature. Flamenco music is an oral tradition where voice is an essential element. Hence, melody is a predominant feature and many styles in flamenco music can be characterized in melodic terms. However, in flamenco music the problem of characterizing styles via melodic patterns has so far received very little attention. In this paper, we study characteristic melodic patterns, i.e., melodic patterns that make a given style recognizable.

In general, it is possible to adopt two main approaches to the study of characteristic melodic patterns. According to the first approach, music is analysed to discover characteristic melodic patterns [2] (distinctive patterns in the terminology of [2]); see, for example, [3] for a practical application of this approach to finding characteristic patterns in Brahms' string quartet in C minor. Typically, the detected patterns are assessed by musicologists to determine how meaningful they are. Therefore, this type of approach is essentially an inductive method. The second approach is in a certain sense complementary to the first one: specific melodic patterns, which are known or are hypothesized to be characteristic, are tracked in the music stream. The results of this type of method allow musicologists to study important aspects of the given musical style, e.g., to confirm existing musical hypotheses. The techniques to carry out such tracking operations vary greatly depending on the application context, the adopted music representation (symbolic or audio), the musical style and the available corpora. This type of approach can be termed as deductive.

In this paper, we adopted the second approach. Specifically, certain characteristic melodic patterns were carefully selected by a group of flamenco experts and were searched in a corpus of flamenco songs that belong to the style of fandango. Tracking patterns in flamenco music is a challenging task for a number of reasons. First of all, flamenco music is usually only available as raw audio recordings without any accompanying metadata. Secondly, flamenco music uses intervals smaller than a half-tone and is not strict with tuning. Furthermore, due to improvisation, a given abstract melodic pattern can be sung in many different ways, sometimes undergoing dramatic transformations, and still be considered the same pattern within the flamenco style. These facts obviously increase the complexity of the melody search operation and demand for increased robustness.

Preliminary work on detecting ornamentation in flamenco music was carried out in [6], where a number of predefined ornaments were adapted from classical music and were looked up in a flamenco corpus of *tonás* styles. In [9] a melodic study of flamenco a cappella singing styles was performed.

1.2 Goals

Two main goals were established for this work: the first one was of technical nature -transcription of music and location of melodic patterns-, and the second one of musicological nature -the study of certain characteristic patterns of the Valverde fandango style.

From an algorithmic perspective, two major problems had to be addressed. The first problem was related to the transcription of music, since flamenco is an oral music tradition and transcriptions are meagre. In addition, our corpus consisted of audio recordings that contained both guitar and voice and predominant melody (pitch) estimation was applied in order to extract the singing voice. The output of this processing stage was a set of pitch contours representing the vocal lines in the recordings. Note that even though we use a state-of-the-art algorithm, theses lines will still contain estimation errors, and our algorithm must be able to cope with them. The second problem was related to the fact that the patterns to be detected were specified by flamenco experts in an abstract (symbolic) way and we had to locate the characteristic patterns directly on the extracted pitch sequences. To this end, we developed a tracking algorithm that operates on a by-example basis and extends the context-dependent dynamic time warping scheme [10], which was originally proposed for pre-segmented data in the context of wind instruments.

Musicologically speaking, the goal was to examine certain melodic patterns as to being characteristic of the Valverde fandango style. Those patterns were specified in a symbolic, abstract way and were detected in the corpus. Both the pattern itself and its location were important from a musicological point of view. The tracking results were reviewed and assessed by a number of flamenco experts. The assessment was carried out with respect to a varying similarity threshold that served as means to filter the results returned by the algorithm. In general, the subjective evaluation of the results (experts' opinion) was consistent with the algorithmic output.

2. THE FANDANGO STYLE

Fandango is one of the most fundamental styles in flamenco music. In Andalusia, there are two main regions where fandango has marked musical characteristics: Malaga (verdiales fandangos) and Huelva (Huelva fandangos).

Verdiales fandangos are traditional folk *cantes* related to dance and a particular sort of gathering. The singing style is melismatic and flowing at the same time [1].

Huelva fandangos are usually sung in accompaniment with a guitar. The oldest references about Huelva fandangos date back to the second half of the XIX century. At present, Huelva fandangos are the most popular ones and display a great number of variants. They can be classified based on the following criteria: (1) Geographical origin: from the mountains (Encinasola), from Andévalo (Alosno), from the capital (Huelva capital fandango); (2) Tempo: fast (Calañas), medium (Santa Barbara), or slow (valientes from Alosno); (3) Origin of tradition: village (Valverde), or personal, i.e., fandangos that are attributed to important singers (Rebollo and other important singers, for example). More information on the different styles of fandango can be found in [7].

From a musicological perspective, all fandangos have a common formal and harmonic structure which is composed of an instrumental refrain in flamenco mode (major Phrygian) and a sung verse or *copla* in major mode. The interpretation of fandangos can be closer to the folkloric style, or to the flamenco style, with predominant melismas and greater freedom in terms of rhythm. The reader may refer to [5] for further information on their musical description.

The study of the fandangos of Huelva is of particular interest for the following reasons: (1) Identification of the musical processes that contribute to the evolution of folk styles to flamenco styles; (2) Definition of styles according to their melodic similarity; (3) Identification of the musical variables that define each style; this includes the discovery of melodic and harmonic patterns.

3. THE CHARACTERISTIC PATTERNS OF FANDANGO STYLES

Patterns heard in the exposition (the initial presentation of the thematic material) are fundamental to recognizing fandango styles. The main patterns identified in the Valverde fandango style are shown in Figure 1 (chords shown in Figure 1 are played by the guitar; pitches are notated as intervals from the root) . These patterns are named as follows: *exp-1*, *exp-2*, *exp-4*, and *exp-6*. The number in the name of the pattern refers to the phrase in which it occurs in the piece.

Pattern *exp-1* is composed of a turn-like figure around the tonic. Pattern *exp-2* basically goes up by a perfect fifth. First, the melody insists on the B flat, makes a minor-second mordent-like movement, and then rises with a leap of a perfect fourth. Pattern *exp-4* is a fall from the tonic to the fourth degree by conjunct degrees followed by an ascending leap of a fourth. Pattern *exp-6* is a movement from B flat to the tonic. Again, the B flat is repeated, then it goes down by a half-tone and raises to the tonic with an ascending minor third. The rhythmic grouping of the melodic cell is ternary (three eighth notes for B flat and three eighth notes for A).

Again, notice that this is a symbolic description of the actual patterns heard in the audio files. Any of these patterns may undergo substantial changes in terms of duration, sometimes even in pitch, not to mention timbre and other expressive features.

3.1 The Corpus of Fandango

The corpus of our study was provided by *Centro andaluz de flamenco de la Junta de Andalucía*, an official institution whose mission is the preservation of the cultural heritage

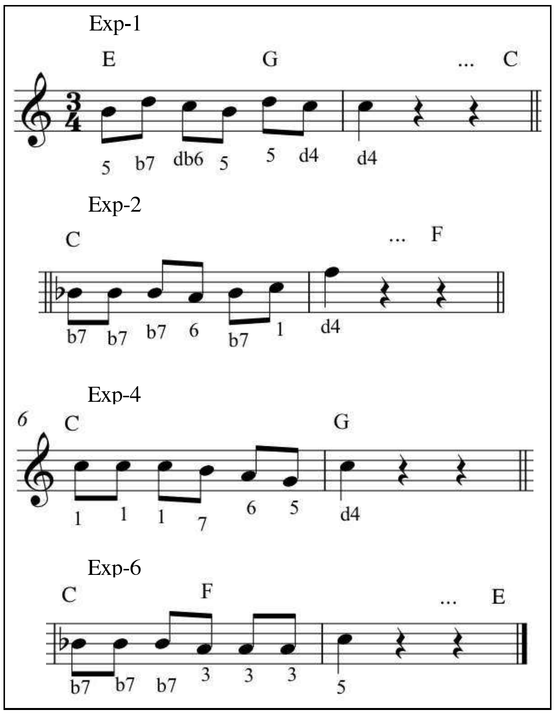


Figure 1. Characteristic patterns in the Valverde fandango style.

of flamenco music. This institution possesses around 1200 fandangos, from which 241 were selected. The selection was based on the following four criteria: (1) Audio files must contain guitar and voice; (2) Audio files are of acceptable recording quality to permit automatic processing; (3) Fandangos must be interpreted by singers from Huelva or acknowledged singing masters; (4) The time span of the recordings must be broad and in our case it covers six decades, from 1950 to 2009.

The corpus was gathered for the purposes of a larger project that aims at investigating fandango in depth. The sample under study is broadly representative of styles and tendencies over time. The current paper is an attempt to study 60 fandangos in total (30 Valverde fandangos and 30 Huelva capital fandangos). In this experimental setup we excluded Valientes of Huelva fandangos, Valientes de Alosno fandangos, Calañas fandangos, and Almonaster fandangos. All recordings were available in PCM (wav) single-channel format, with a 16 bit-depth per sample and 44 kHz sampling rate.

4. COMPUTATIONAL METHOD

4.1 Audio Feature Extraction

As mentioned earlier, written scores in flamenco music are scattered and scant. This can be explained to some extent by the fact that flamenco music is based on oral transmission. Issues related to the most appropriate transcription method have been quite controversial in the context of the flamenco community. Some authors, like Hurtado and Hurtado [8], are in favour of Western notation, whereas others propose different methods, e.g., Donnier [4], who advocates the use of plainchant neumes. In view of this controversy, we adopted a more technical approach that is based on audio feature extraction.

We now describe how the audio feature extraction algorithm operates. Our goal was to extract the vocal line in an appropriate, musically meaningful format that would also serve as input to the pattern detection algorithm. The audio feature extraction stage was mainly based on predominant melody (fundamental frequency, from now on $F0$) estimation from polyphonic signals. For this, we used the state-of-the art algorithm proposed by Salamon and Gómez [11]. Their algorithm is composed of four blocks. First, they extract spectral peaks from the signal by taking the local maxima of the short-time Fourier transform. Next, those peaks are used to compute a salience function representing pitch salience over time. Then, peaks of the salience function are grouped over time to form pitch contours. Finally, the characteristics of the pitch contours are used to filter out non-melodic contours, and the melody $F0$ sequence is selected from the remaining contours by taking the frequency of the most salient contour in each frame. Further details can be found in [11].

4.2 Pattern Recognition Method

The pattern detection method used in this paper builds upon the “Context-Dependent Dynamic Time Warping” algorithm (CDDTW) [10]. While standard dynamic time warping schemes assume that each feature in the feature sequence is uncorrelated with its neighboring ones (i.e. its context), CDDTW allows for grouping neighboring features (i.e. forming feature segments) in order to exploit possible underlying mutual dependence. This can be useful in the case of noisy pitch sequences, because it permits canceling out several types of pitch estimation errors, including pitch halving or doubling errors and intervals that are broken into a sequence of subintervals. Furthermore, in the case of melismatic music, the CDDTW algorithm is capable of smoothing variations due to the improvisational style of singers or instrument players. For a more detailed study of the CDDTW algorithm, the reader is referred to [10].

A drawback of CDDTW is that does not take into account the duration of music notes and focuses exclusively on pitch intervals. Furthermore, CDDTW was originally proposed for isolated musical patterns (pre-segmented data). The term isolated refers to the fact that the pattern that is matched against a prototype has been previously extracted from its context by means of an appropriate segmentation procedure, which can be a limitation in some real-world scenarios, like the one we are studying in this paper. Therefore, we propose here an extension to the CDDTW algorithm, that:

- Removes the need to segment the data prior to the application of the matching algorithm. This means that the prototype (in our case the time-pitch representation of the MIDI pattern) is detected directly in the pitch sequence of the audio stream without prior segmentation, i.e. the pitch sequence that was extracted from the fandango.

- Takes into account the note durations in the formulation of the local similarity measure.
- Permits to search for a pattern iteratively, which means that multiple instances of the pattern can be detected, one per iteration.

A detailed description of the extension of the algorithm is beyond the scope of this paper. Instead, we present the basic steps:

Step 1: The MIDI pattern to be detected is first converted to a time-pitch representation

$$P = \{[f_1, t_1]^T, [f_2, t_2]^T, \dots, [f_J, t_J]^T\},$$

where f_i is the frequency of the i-th MIDI note, measured in cents (assuming that the reference frequency is 55 Hz) and t_i is the respective note duration (in seconds), for a MIDI pattern of J notes.

Step 2: Similarly, the pitch sequence of the audio recording is converted to the above time-pitch representation,

$$R = \{[r_1, tr_1]^T, [r_2, tr_2]^T, \dots, [r_I, tr_I]^T\},$$

where r_i is a pitch value (in cents) and tr_i is always equal to the short-term step of the feature extraction stage ($10ms$ in our case), for an audio recording of I notes. In other words, even if two successive pitch values are equal, they are still treated as two successive events, each of which has a duration equal to the short-term step of the feature extraction stage. This approach was adopted to increase the flexibility of the dynamic time warping technique at the expense of increased computational complexity. For the sake of uniformity of representation, each time interval that corresponds to a pause or to a non-vocal part is inserted as a zero-frequency note and is assigned a respective time duration.

Step 3: Sequences R and P are placed on the vertical and horizontal axis of a similarity grid, respectively. The CDDTW algorithm is then applied on this grid, but, this time, the cost to reach node (i, j) from an allowable predecessor, say $(i-k, j-1)$, depends both on the pitch intervals and the respective note durations. More specifically, the interpretation of the transition $(i-k, j-1) \to (i, j)$ is that the pitch intervals in the MIDI pattern and audio recording are equal to $f_j - f_{j-1}$ and $r_i - r_{k-1}$, respectively. Note that on the y-axis, the pitch interval only depends on the end nodes of the transition and not on any intermediate pitch values, hence the ability to cancel out any intermediate pitch tracking phenomena. In the same spirit, the time duration that has elapsed on the $x-$axis and $y-$axis is equal to t_j and $\sum_{i-k+1}^{i} tr_k$, respectively. It is worth noticing that we do not permit omitting notes from the MIDI pattern, and therefore any allowable predecessor of (i, j) must reside in column $j-1$. The pitch intervals and respective durations are fed to the similarity function of Eq. (1), that yields a score, $S_{(i-k,j-1)\to(i,j)}$, for the transition $(i-k, j-1) \to (i, j)$, i.e.,

$$\begin{aligned} S_{(i-k,j-1)\to(i,j)} = 1 - f(\frac{\sum_{i-k+1}^{i} tr_k}{t_j}) \\ -g(r_i - r_{k-1}, f_j - f_{j-1}) \end{aligned} \quad (1)$$

where

$$f(x) = \begin{cases} (1-x)^{1.1}, & 1 \le x \le 4 \\ 1.5^{1.1}(1-x)^{1.1}, & \frac{1}{3} \le x < 1 \\ 3 - 6x, & 0 < x < \frac{1}{3} \\ \infty, & \text{otherwise} \end{cases}$$

and

$$g(x_1, x_2) = \begin{cases} (1 - \frac{x_1}{x_2})^{0.7}, & \text{if } 0.98 \le \frac{x_1}{x_2} \le 1.02 \\ \infty, & \text{otherwise} \end{cases}$$

The interpretation of this function is that it penalizes excessive time warping and does not tolerate much deviation in terms of pitch intervals. More specifically, $f(x)$ is a piecewise function that operates on the basis that duration ratios are not penalized uniformly and that any ratio outside the interval $[\frac{1}{3}, 1)$ should receive a stronger penalty. Similarly, function $g(x)$ implies that, taking the music interval of the MIDI pattern as reference, the respective sum of intervals of the audio recording exhibits at most a 2% deviation. The scalars involved in the formulae of $f(x)$ and $g(x)$ are the result of fine-tuning with respect to the corpus under study. The computation of the transition cost is repeated for every allowable predecessor of (i, j). In the end, one of the predecessors is selected to be the winner by examining the sum of the similarity that has been generated by the transition with the accumulated similarity at the predecessor.

Step 4: After the accumulated cost has been computed for all nodes in the grid, the maximum accumulated cost is selected and normalized and, if it exceeds a predefined threshold, a standard backtracking procedure reveals which part of the audio recording has been matched with the prototype; otherwise, the algorithm terminates.

Step 5: All nodes in the best path are marked as stop-nodes, i.e. forbidden nodes and Steps 1-4 are repeated in order to detect a second occurrence of the prototype and so on, depending on how many patterns (at most) the user has requested to be detected.

5. EVALUATION

5.1 Methodology

Four different exposition patterns were defined by the experts, which are distinctive of the Valverde style. The Valverde fandango has 6 exposition phrases in each *copla* (sung verse), where 1, 3 and 5 are usually the same pattern, and 2, 4 and 6 have different patterns each. Therefore, 4 exposition patterns (1, 2, 4, and 6) were chosen to be put to the test. Again, we insist that these patterns are abstract representations of the actual patterns heard in the audio recordings. Our algorithm was then run to locate those four patterns in the corpus of Valverde fandangos and Huelva capital fandangos. Therefore, our ground-truth in this study consists of all the melodic patterns plus their specific locations. For example, exposition pattern 1 has to be located 90 times, as it occurs three times in each of the 30 pieces that make up the corpus of the Valverde fandangos. If this pattern is found elsewhere (not in the exposition phrase), then it will be considered as a true negative. Once the results of the experiments were obtained, they were manually checked by the flamenco experts, both in terms of pattern occurrence and respective position.

5.2 Results

Results are summarized in Tables 1 and 2 with respect to the similarity threshold, which is a user-controlled variable. Once the threshold is set to a specific value, the algorithm filters out any patterns whose similarity score does not exceed the threshold. In our study, we experimented with values of the similarity threshold ranging from 30% to 80%. In Table 1, T_e stands for the total number of expected occurrences of each pattern in the corpus of Valverde fandangos (based on the ground truth that is provided by the musicological knowledge), T_f is the total number of detected instances (both true and false), T_p is the number of true positives, F_p is the number of false positives, and *Prec.*, *Rec.* and F are the values of precision, recall and the F-measure, respectively. In Table 2, we focus on the corpus of Huelva fandangos.

Figure 2 shows the average F-measure (over all patterns) as a function of similarity threshold. The maximum value is obtained at threshold 50%.

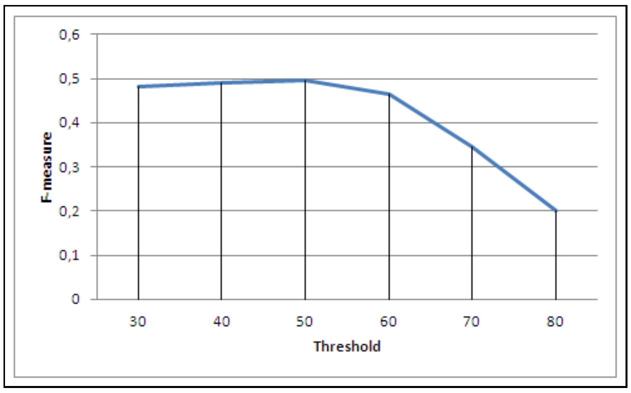


Figure 2. Average F-measure (over all patterns) with respect to the similarity threshold.

Next, we attempt to detect the Valverde patterns in the Huelva collection. Hence, one would expect that it would be otiose to reproduce computations like those in Table 1, as the total expected number of occurrences would be zero. However, Table 2 summarizes the detection results in the corpus of the Huelva capital fandangos for the four exposition patterns under study and we make an attempt to provide an interpretation of the detected occurrences.

Valverde fandangos								
	Sim.	T_e	T_f	T_p	F_p	Prec.	Rec.	F
Exp-1	30%	90	38	32	6	84%	36%	0.5
	40%	90	36	32	4	89%	36%	0.5
	50%	90	31	30	1	97%	33%	0.49
	60%	90	25	24	1	96%	27%	0.41
	70%	90	15	15	0	100%	17%	0.28
	80%	90	6	6	0	100%	7%	0.12
Exp-2	30%	30	13	13	0	100%	43%	0.6
	40%	30	13	13	0	100%	43%	0.6
	50%	30	13	13	0	100%	43%	0.6
	60%	30	13	13	0	100%	43%	0.6
	70%	30	7	7	0	100%	23%	0.37
	80%	30	1	1	0	100%	3%	0.06
Exp-4	30%	30	27	11	16	41%	37%	0.38
	40%	30	26	11	15	42%	37%	0.39
	50%	30	21	11	10	52%	37%	0.43
	60%	30	16	9	7	56%	30%	0.39
	70%	30	11	6	5	54.5%	20%	0.29
	80%	30	3	3	0	100%	10%	0.18
Exp-6	30%	30	34	14	20	41%	47%	0.43
	40%	30	31	14	17	45%	47%	0.46
	50%	30	27	13	14	48%	43%	0.45
	60%	30	15	10	5	66.6%	33%	0.44
	70%	30	8	8	0	100%	27%	0.42
	80%	30	3	3	0	100%	10%	0.18

Table 1. Experimental results for Valverde fandangos.

Huelva capital fandangos						
	30%	40%	50%	60%	70%	80%
Exp-1	7	4	1	1	0	0
Exp-2	1	0	0	0	0	0
Exp-4	29	27	23	17	8	4
Exp-6	27	22	19	17	9	5

Table 2. Experimental results for the Huelva capital fandangos.

Overall, from a quantitative point of view, the algorithm has exhibited a reasonably good performance in finding the patterns in the melody, despite the problems posed by the polyphonic source, the highly melismatic content, and the note-duration variation. Regarding performance measures, on the one hand, precision is quite high, but, on the other hand, recall is low. Most of the values of F-measure are around $0.3 - 0.45$, with a few isolated exceptions. In other words, the algorithm is capable of detecting well localized occurrences of the patterns, but fails to locate a significant number of occurrences. The best performance of the F-measure occurs with a threshold of 50%.

From a qualitative point of view, we make the following remarks.

Exp-1: This pattern is the exposition of the first phrase of the fandango. Interestingly enough, not only does the algorithm detect the pattern correctly in the first phrase of the Valverde fandango, but also in other phrases, as expected. Indeed, it identifies the pattern as a leit-motiv throughout the piece. This pattern was detected only a few times by the algorithm in the Huelva capital fandangos.

Exp-2: This is the pattern of the second exposition phrase in Valverde fandangos. This is the musical passage with the amplest tessitura. The algorithm detects it with high precision in the Valverde corpus (even for a similarity threshold equal to 30%), and very few matches are encountered in the Huelva capital fandangos.

Exp-4: In the Valverde corpus, for a threshold equal to 80%, the algorithm only detects the pattern in *cantes* sung by women who have received music training in flamenco clubs in Huelva. These clubs are called *peñas flamencas* and organize singing lessons. Women from *peñas* are trained to follow very standard models of singing and therefore do not contribute to music innovation like other fandango performers e.g., Toronjo or Rengel). For a 70% similarity threshold (and below), the pattern is also detected in the voices of well-known fandango singers.

In the Huelva capital fandango corpus this pattern is frequently detected by the algorithm in the transition between phrases. Note that we can state that the pattern is there, more or less blurred or stretched, but it is present, so these are not considered to be false positives.

Exp-6: This pattern is used to prepare the final cadence of the last phrase. In the Valverde corpus, irrespective of the similarity level, the algorithm returns correct results, although as stated above, many occurrences fail to be detected. In the Huelva capital corpus and when the threshold is low, the algorithm detects the pattern in the first, the middle and the final section. When the threshold is raised to 80%, it is only located in the final cadence.

6. CONCLUSIONS

In this paper we presented an algorithmic pipeline to perform melodic pattern detection in audio files. The overall performance of our method depends both on the quality of the extracted melody and the precision of the tracking algorithm. In general, the system's performance, in terms of precision and recall of detected patterns, was measured to be satisfactory, despite the great amount of melismas and the high tempo deviation. From a musicological perspective, we carried out a study of fandango styles by means of analyzing archetypal melodic patterns. As already mentioned, written scores are not in general available for flamenco music. Therefore, our approach was to design a system that operated directly on raw audio recordings and circumvented the need for a transcription stage. In the future, our study could be extended to other Huelva fandango styles. A more ambitious goal would be to carry out the analysis for the whole corpus of fandango music. Also, other musical features could be taken into account and thus perform a more general analysis, i.e., embrace more than what melodic descriptors can offer.

7. ACKNOWLEDGEMENTS

The authors would like to thank the *Centro Andaluz del Flamenco, Junta de Andalucía*[1] for providing the music collection. This work has been partially funded by AGAUR (mobility grant), the COFLA project[2] (P09 - TIC - 4840 Proyecto de Excelencia, Junta de Andalucía) and the Programa de Formación del Profesorado Universitario of the Ministerio de Educación de España.

8. REFERENCES

[1] M. A. Berlanga. *Bailes de candil andaluces y fiesta de verdiales. Otra visión de los fandangos*. Colección monografías. Diputación de Málaga, 2000.

[2] D. Conklin. Discovery of distinctive patterns in music. *Intelligent Data Analysis*, 14(5):547–554, 2010.

[3] D. Conklin. Distinctive patterns in the first movement of Brahms' string quartet in C minor. *Journal of Mathematics and Music*, 4(2):85–92, 2010.

[4] P. Donnier. Flamenco: elementos para la transcripción del cante y la guitarra. In *Proceedings of the IIIrd Congress of the Spanish Ethnomusicology Society*, 1997.

[5] Lola Fernández. *Flamenco Music Theory*. Acordes Concert, Madrid, Spain, 2004.

[6] F. Gómez, A. Pikrakis, J. Mora, J.M. Díaz-Báñez, E. Gómez, and F. Escobar. Automatic detection of ornamentation in flamenco. In *Fourth International Workshop on Machine Learning and Music MML*, NIPS Conference, December 2011.

[7] M. Gómez(director). Rito y geografía del cante flamenco II . Videodisco. Madrid: Círculo Digital, D.L., 2005.

[8] A. Hurtado Torres and D. Hurtado Torres. *La voz de la tierra, estudio y transcripción de los cantes campesinos en las provincias de Jaén y Córdoba*. Centro Andaluz de Flamenco, Jerez, Spain, 2002.

[9] J. Mora, F. Gómez, E. Gómez, F. Escobar Borrego, and J. M Díaz Báñez. Characterization and melodic similarity of a cappella flamenco cantes. In *Proceedings of ISMIR*, pages 9–13, Utrecht School of Music, August 2010.

[10] A. Pikrakis, S. Theodoridis, and D. Kamarotos. Recognition of isolated musical patterns using context dependent dynamic time warping. *IEEE Transactions on Speech and Audio Processing*, 11(3):175–183, 2003.

[11] J. Salamon and E. Gómez. Melody Extraction from Polyphonic Music Signals using Pitch Contour Characteristics. *IEEE Transactions on Audio, Speech and Language Processing*, 20(6):1759-1770, Aug. 2012.

[1] http://www.juntadeandalucia.es/cultura/centroandaluzflamenco/
[2] http://mtg.upf.edu/research/projects/cofla

A CROSS-VERSION APPROACH FOR STABILIZING TEMPO-BASED NOVELTY DETECTION

Meinard Müller[1,3], **Thomas Prätzlich**[2,3], **and Jonathan Driedger**[1,2]
[1] Bonn University, [2] Saarland University, [3] MPI Informatik
{meinard,tpratzli}@mpi-inf.mpg.de, driedger@iai.uni-bonn.de

ABSTRACT

The task of novelty detection with the objective of detecting changes regarding musical properties such as harmony, dynamics, timbre, or tempo is of fundamental importance when analyzing structural properties of music recordings. But for a specific audio version of a given piece of music, the novelty detection result may also crucially depend on the individual performance style of the musician. This particularly holds true for tempo-related properties, which may vary significantly across different performances of the same piece of music. In this paper, we show that tempo-based novelty detection can be stabilized and improved by simultaneously analyzing a set of different performances. We first warp the version-dependent novelty curves onto a common musical time axis, and then combine the individual curves to produce a single fusion curve. Our hypothesis is that musically relevant points of novelty tend to be consistent across different performances. This hypothesis is supported by our experiments in the context of music structure analysis, where the cross-version fusion curves yield, on average, better results than the novelty curves obtained from individual recordings.

1. INTRODUCTION

Music is highly structured data. Structure in music arises from repetitions, contrasts and homogeneity in musical aspects such as melody, dynamics, harmony, timbre or tempo [12]. The extraction of the musical structure from audio recordings is an important task in the field of music information retrieval. It consists of a segmentation problem, where the goal is to find the boundaries that mark the transitions between two structural parts, and a musically meaningful labeling (e.g. chorus, verse, first theme, second theme) of the segments, see [3, 12] for an overview. In many cases, segment boundaries are accompanied by a change in instrumentation, dynamics, harmony, tempo, or some other characteristics. The task of *novelty detection* is to specify points within a given audio recording where a human listener would recognize such a change [6,9,14,15].

Such points of novelty are not only of musical relevance, but also allow for speeding up further music analysis tasks [11].

In this paper, we present a general approach for stabilizing novelty-based segmentation techniques. Following [6], we first convert the audio signal into a suitable feature representation, compute a self distance matrix, and derive a novelty curve by detecting 2D corner points in this matrix. The choice of features (e. g. MFCCs, chroma features, tempogram features) depends on the musical aspects (e. g. timbre, harmony, tempo) of interest [9]. In the following, we consider the aspect of tempo using the cyclic tempogram features as proposed in [7] as an illustrative example. Particularly in classical music, there often exist many different recordings for a given piece of music. Even though all recordings follow the same musical score, two distinct versions may differ significantly in performance aspects regarding tempo, dynamics, or timbre. This is the reason why novelty detection results often vary across different audio versions.

The main contribution of this paper is to apply the novelty detection simultaneously to a set of different performances of a given piece. To this end, using a score-based MIDI reference, we convert the physical time axis (in seconds) of all version-dependent novelty curves into a common musical time-axis (in measures) . Then we combine the individual curves into a cross-version fusion curve, see Figure 1 for an overview. Assuming that the musically interesting points of novelty are consistent across the different versions, we expect the fusion curve to be more stable and musically meaningful than the individual curves. Applying our cross-version novelty detection approach for locating segment boundaries in music structure analysis, we show that the fusion curves yield, on average, better results than the version-dependent novelty curves of individual recordings. This effect becomes more prominent, when there is a high performance variance across the recordings, which is typically the case for the aspect of tempo.

Cross-version strategies have previously been applied for other music analysis tasks. For example, multiple performances are used in [1] to support tempo tracking, in [10] to stablize chord labeling, and in [8] to detect critical passages in a piece of music that are prone to beat tracking errors.

The remainder of this paper is organized as follows. In Section 2, we describe the various steps of our cross-version novelty detection procedure. Then, in Section 3,

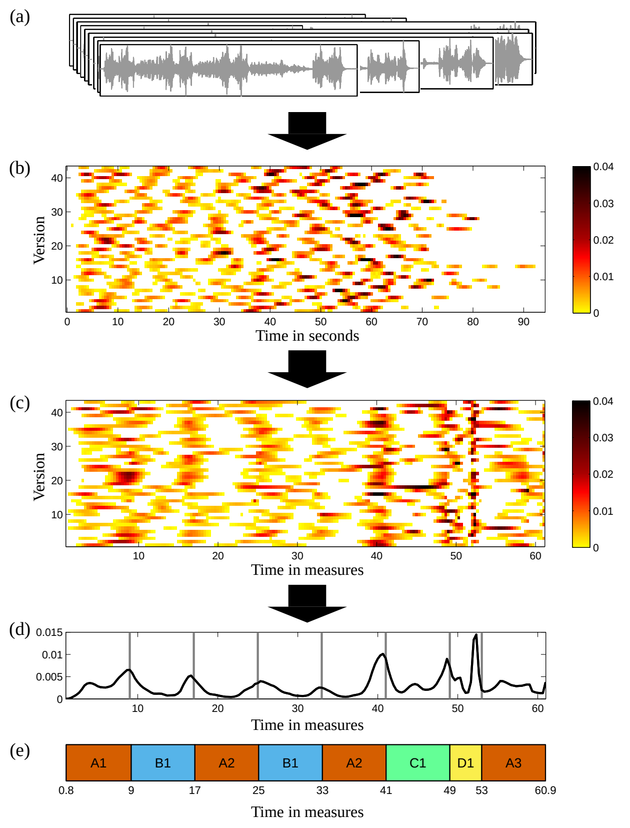


Figure 1: Overview of the cross-version novelty detection pipeline for Chopin's Mazurka Op. 7 No. 4. **(a)** Waveforms of several performances. **(b)** Individual novelty curves (color-coded) for 43 performances. Each row of the matrix corresponds to one novelty curve. **(c)** Individual novelty curves warped to a common musical time axis (in measures). **(d)** Fusion novelty curve. **(e)** Annotated structure and segment boundaries.

we give a detailed quantitative evaluation of our procedure within a structure analysis scenario for Chopin's Piano Mazurkas. Furthermore, we critically assess the results by a musically informed discussion of concrete examples. Finally, we conclude with Section 4 indicating future work.

2. CROSS-VERSION NOVELTY DETECTION

In this section, we describe the pipeline for our cross-version approach to novelty detection. For the purpose of illustration, we concentrate on the musical aspect of tempo using a cyclic tempogram feature representation (Section 2.1). As for the novelty detection, we follow a standard procedure based on 2D corner detection in self distances matrices (Section 2.2). Applying music synchronization techniques, we show how to warp the novelty curves onto a version-independent musical time axis (Section 2.3). Finally, we describe how to merge the novelty curves based on a late-fusion strategy (Section 2.4). This pipeline is also illustrated by Figure 1.

2.1 Cyclic Tempogram Features

In a first step, the given audio recording is transformed into a suitable feature representation that captures the musical aspects of interest. As an example, we consider the case of tempo-based novelty detection, even though our cross-version approach is applicable to any kind of feature representation. In the following, we revert to cyclic tempogram features as introduced in [7]. These features constitute a robust mid-level representation encoding local tempo information. In a first step, we capture changes in the signal's energy and spectrum [2] and then apply windowed autocorrelation methods [4]. Afterwards, the lag-axis is converted into a tempo axis specified in beats per minute (BPM), yielding a tempogram as shown in Figure 2c. Forming tempo equivalence classes by binning tempi that differ by a power of two and quantizing the values of the resulting cyclic tempogram yields an even more robust feature representation, see Figure 2d. In our experiments we use a feature resolution of 5 Hz (five feature vectors per second) and a feature dimension of 10 (ten feature values per vector). A free MATLAB implementation of these features is part of the tempogram toolbox.[1] For further details we refer to [7].

2.2 Novelty Curve

Let $X = (x_1, \ldots, x_N)$ denote the resulting feature sequence. To compute a novelty curve from this sequence, we employ a standard approach introduced by Foote [6]. To this end, an $N \times N$ self distance matrix $\mathbf{D}(n,m) := \mathbf{d}(x_n, x_m)$ is computed using the local distance function

$$\mathbf{d}(x_n, x_m) = 1 - \exp\left(\frac{\langle x_n, x_m \rangle}{\| x_n \| \| x_m \|} - 1\right),$$

for $1 \leq n, m \leq N$. Then, $\mathbf{D}$ is analyzed by correlating a kernel along its main diagonal. The kernel consists of an $M \times M$ matrix (with $M < N$) which has a 2×2 checkerboard-like structure weighted by a Gaussian radial function. This yields a *novelty curve*, the peaks of which indicate changes in the musical aspect represented by the feature type (in our case, tempo changes), see Figure 2e. We further process the novelty curve by subtracting a local average, see Figure 2f. In our experiments, a value M corresponding to 7 seconds has turned out to be suitable, see Section 3.2 and Figure 3 for a further discussion of the parameter M.

2.3 Time Axis Conversion

The computed novelty curve depends on the performance characteristics of the underlying music recording. To make novelty curves comparable across different recordings of the same piece of music, we convert the version-dependent physical time axis (in seconds) to a version-independent musical time axis (in measures). To this end, we assume that we are given a score-like MIDI version of the piece with explicit beat and measure positions. Then, for a given music recording, we apply music synchronization techniques to automatically align the MIDI version with the audio version.[2] The alignment result allows for transferring the beat and measure positions specified by the MIDI

[1] `www.mpi-inf.mpg.de/resources/MIR/tempogramtoolbox`
[2] In our implementation, we revert to the high-resolution music synchronization approach described in [5].

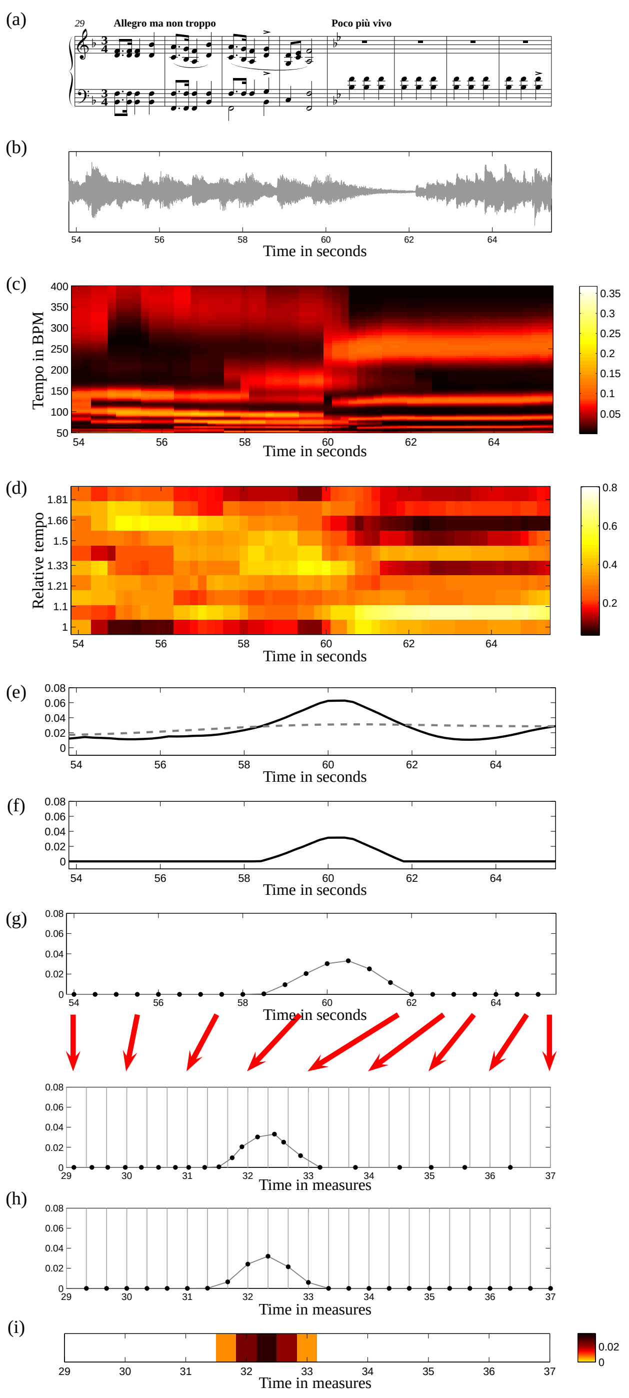


Figure 2: Novelty detection for a recording of Chopin's Mazurka Op. 68 No. 3. **(a)** Measures 29-36. **(b)** Waveform. **(c)** Tempogram. **(d)** Quantized cyclic tempogram. **(e)** Novelty curve (solid line) and its local average curve (dashed line). **(f)** Post-processed novelty curve. **(g)** Time axis conversion. **(h)** Resampled novelty curve. **(i)** Color-coded representation of (h).

version to the corresponding time positions in the audio version. Based on this information, we locally stretch and contract the time axis of the novelty curve computed from the recording to obtain a musical time axis, see Figure 2g. Finally, we interpolate and resample the novelty curve to obtain one value for each beat position of the piece of music, see Figure 2h and Figure 2i.

2.4 Fusion Novelty Curve

Being based on the same musical time axis, one can now directly compare novelty curves from different performances of the same piece of music. As an example, Figure 1b shows the original novelty curves (in some color-coded form) for 43 different performances of Chopin's Mazurka Op. 7 No. 4. No correlations across the different performances are visible. After the time axis conversion, as shown in Figure 1c, strong correlations between the different novelty curves become evident. For example, there is a tempo change at measure 52 for basically all performances.

To fuse the information across all novelty curves, we basically compute the average of the novelty curves. To become more robust to outliers, we first remove the 20 % smallest and largest novelty values for each beat position among all performances, and then compute the *fusion novelty curve* by taking the beat-wise arithmetic mean of the remaining values. The crucial observation is that a fusion novelty curve reveals a local maximum (peak) at those positions where a large number of individual novelty curves also possess a local maximum. In other words, the fusion novelty curve expresses the consistencies in the peak structures across the various recordings, see also Figure 1d.

3. EXPERIMENTS

Even though there are often significant differences in the way musicians interpret a piece of music, tempo changes are not arbitrary and there are musical reasons for a speed up or slow down. Our hypothesis is that the tempo changes that can be observed across a large number of different performances are of particular musical importance. Therefore, we conjecture that peaks of the fusion novelty curve are more relevant than the peaks of the individual novelty curves. To investigate our hypothesis, we have conducted various experiments on a dataset consisting of Chopin's Mazurkas (Section 3.1). Our quantitative evaluation in the context of music structure analysis (Section 3.2) as well as a discussion of various representative examples (Section 3.3) demonstrate that cross-version fusion curves yield, on average, better results than the novelty curves obtained from individual recordings.

3.1 Dataset and Annotations

We conduct our experiments on a Mazurka dataset, which consists of 2792 recorded performances for the 49 Mazurkas by Frédéric Chopin. These recordings were collected in the Mazurka Project[3] and have been previously used, e. g., for the purpose of performance analysis [13]. For each of the 49 Mazurkas, there are on average 57 different recordings (ranging from the early stages of music recording until today), as well as a MIDI file that represents the piece in an uninterpreted symbolic form. In particular, measure and beat positions are known in the MIDI file.

The Chopin Mazurkas are short piano compositions with a 3/4 time signature. These pieces have a relatively

[3] mazurka.org.uk

clear musical structure, where certain parts are repeated more or less in the same way. We have manually annotated each score-like MIDI file according to its musical structure. On average this leads to 9.4 segment boundaries per Mazurka (disregarding segment boundaries at the beginning and end of the piece) and an average duration of 11.9 measures per musical part, see also Table 1 for more details.

3.2 Quantitative Evaluation

As is the case for romantic piano music, most Mazurka performances reveal numerous local tempo changes which often indicate transitions of musical importance. Many of these transitions occur near segment boundaries between musical parts, where one can often observe tempo changes. Even though not all segment boundaries are characterized this way, we use them for a first quantitative evaluation to indicate the behavior of our cross-version fusion novelty curves. Let $\mathcal{B}$ denote the set of segment boundaries (specified in musical beats) for a given Mazurka.

For a novelty curve (with time axis given in musical beats), we perform some peak picking to determine a set $\mathcal{P}$ of relevant peak positions. Here a position is considered relevant if the novelty curve assumes at this position a global maximum over a window of length λ centered at the corresponding position. In our experiments, the value $\lambda = 19$ beats has turned out to be meaningful, see also Figure 3. A peak position in $\mathcal{P}$ is considered to be *true* if there is a segment boundary in $\mathcal{B}$ in a δ-neighborhood, otherwise it is considered to be *false*. This allows to define a precision (P), recall (R), and F-measure (F) for the set $\mathcal{P}$ relative to $\mathcal{B}$. In our experiments, we choose $\delta = 3$ beats corresponding to a musical measure. In our evaluation, we further ignored all boundaries and all peaks in the first four and last four measures of a piece of music. The main reason for excluding these measures is that many of the recordings start and end with non-musical content such as silence or applause, which leads to spurious peaks at the positions where the music starts or ends. Also, synchronization errors typically occur in these regions.

Before we investigate the role of the various parameters, we first look at the results for a fixed parameter setting as indicated by Table 1. To better understand the effect of the cross-version approach, we computed P/R/F-measures in two different ways. First, for a given Mazurka, we computed individual P/R/F-measures for each performance using the version-dependent novelty curves and then averaged over all performances to obtain averaged individual P/R/F-measures. Secondly, we computed these measures from the fusion novelty curve to obtain cross-version P/R/F-measures. Table 1 shows the resulting averaged individual as well as cross-version P/R/F-measures for all of the 49 Mazurkas. Furthermore, the last row of the table indicates the overall values averaged over all Mazurkas. As the main result, one can see that the overall F-measure obtained from individual novelty curves is $F = 0.39$, whereas the overall F-measure obtained from the fusion novelty curves is $F = 0.52$. In other words, the tempo-based novelty detection can indeed be improved when simultaneously analyzing a set of different performances.

				Ind.-Version			Cross-Version		
Piece	#P	#M	#B	P	R	F	P	R	F
M06-1	49	112	7	0.29	0.60	0.39	0.50	1.00	0.67
M06-2	51	96	9	0.42	0.63	0.50	0.47	0.89	0.62
M06-3	47	98	12	0.23	0.31	0.26	0.13	0.18	0.15
M06-4	46	40	9	0.65	0.40	0.49	0.83	0.63	0.71
M07-1	55	104	9	0.30	0.52	0.38	0.44	0.89	0.59
M07-2	51	120	14	0.40	0.45	0.42	0.36	0.36	0.36
M07-3	65	105	11	0.31	0.44	0.36	0.47	0.64	0.54
M07-4	43	60	7	0.51	0.60	0.55	0.75	0.86	0.80
M07-5	46	20	12	0.61	0.35	0.44	1.00	0.60	0.75
M17-1	52	100	10	0.25	0.35	0.29	0.07	0.10	0.08
M17-2	55	68	3	0.21	0.65	0.32	0.25	1.00	0.40
M17-3	51	168	10	0.26	0.64	0.37	0.42	1.00	0.59
M17-4	93	132	10	0.23	0.49	0.31	0.35	0.67	0.46
M24-1	61	96	10	0.30	0.40	0.34	0.46	0.60	0.52
M24-2	66	120	16	0.38	0.48	0.42	0.50	0.64	0.56
M24-3	55	79	6	0.26	0.46	0.33	0.45	0.83	0.59
M24-4	76	186	20	0.42	0.59	0.49	0.65	0.85	0.74
M30-1	50	53	4	0.34	0.57	0.42	0.50	0.75	0.60
M30-2	60	64	7	0.48	0.60	0.53	0.78	1.00	0.88
M30-3	63	111	10	0.30	0.45	0.36	0.50	0.60	0.55
M30-4	65	139	14	0.34	0.51	0.40	0.47	0.62	0.53
M33-1	55	48	4	0.43	0.62	0.51	0.50	0.75	0.60
M33-2	70	143	16	0.34	0.47	0.39	0.33	0.44	0.38
M33-3	50	48	3	0.28	0.61	0.38	0.33	1.00	0.50
M33-4	74	224	19	0.24	0.40	0.30	0.27	0.37	0.31
M41-1	56	139	14	0.26	0.39	0.32	0.19	0.29	0.23
M41-2	63	68	7	0.43	0.57	0.49	0.75	0.86	0.80
M41-3	40	78	13	0.44	0.45	0.44	0.57	0.73	0.64
M41-4	45	74	9	0.42	0.58	0.48	0.62	0.89	0.73
M50-1	49	104	6	0.18	0.48	0.26	0.20	0.50	0.29
M50-2	58	127	10	0.31	0.56	0.40	0.56	0.90	0.69
M50-3	74	208	10	0.18	0.57	0.27	0.21	0.70	0.33
M56-1	42	204	13	0.18	0.41	0.25	0.11	0.23	0.15
M56-2	53	92	8	0.31	0.60	0.41	0.54	1.00	0.70
M56-3	57	220	15	0.23	0.51	0.32	0.30	0.67	0.42
M59-1	63	142	9	0.19	0.42	0.26	0.17	0.44	0.25
M59-2	63	111	4	0.10	0.41	0.16	0.20	0.75	0.32
M59-3	66	154	11	0.20	0.44	0.28	0.30	0.55	0.39
M63-1	46	102	9	0.26	0.46	0.33	0.27	0.44	0.33
M63-2	65	56	4	0.33	0.61	0.43	0.38	0.75	0.50
M63-3	88	76	8	0.41	0.55	0.47	0.78	0.88	0.82
M67-1	44	60	7	0.23	0.31	0.26	0.14	0.17	0.15
M67-2	41	72	6	0.33	0.54	0.41	0.45	0.83	0.59
M67-3	46	56	3	0.30	0.75	0.43	0.43	1.00	0.60
M67-4	59	112	6	0.26	0.71	0.38	0.40	1.00	0.57
M68-1	46	84	11	0.53	0.63	0.57	0.50	0.50	0.50
M68-2	65	84	11	0.49	0.54	0.51	0.77	0.91	0.83
M68-3	51	60	9	0.61	0.55	0.57	0.78	0.78	0.78
M68-4	63	63	7	0.35	0.37	0.36	0.47	0.58	0.52
∅	57.0	103.7	9.4	0.33	0.51	0.39	0.45	0.69	0.52

Table 1: Overview of the Mazurka dataset and precision (P), recall (R), and F-measures (F) for two different settings. The first four columns specify the Mazurka (e.g. `M06-1` refers to Mazurka Op. 6 No. 1), the number of performances (#P), the number of measures (#M), and the number of annotated segment boundaries (#B). The next three columns show the average individual P/R/F-measures obtained from individual performances and the last three columns show cross-version P/R/F-measures obtained from the fusion novelty curves. The used parameters are: $M \sim 7$ seconds, $\lambda = 19$ beats, $\delta = 3$ beats.

In the next experiments, we investigate the role of the kernel size parameter M (see Section 2.2) and the neighborhood parameter λ used in the peak picking. Figure 3 shows the cross-version P/R/F-measures averaged over all 49 Mazurkas for various combinations of M and λ. Generally, when increasing λ, the precision increases (Figure 3a) and the recall decreases (Figure 3b). This is not surprising, since an increase in λ imposes stricter conditions on the peak picking (and the set of relevant peaks becomes smaller). The remaining peaks tend to be true (increase in precision), while fewer segment boundaries in $\mathcal{B}$ are detected (decrease in recall). The kernel size parameter M has a minor influence on the final results. Only for large values of λ, smaller kernel sizes tend to be favorable. As

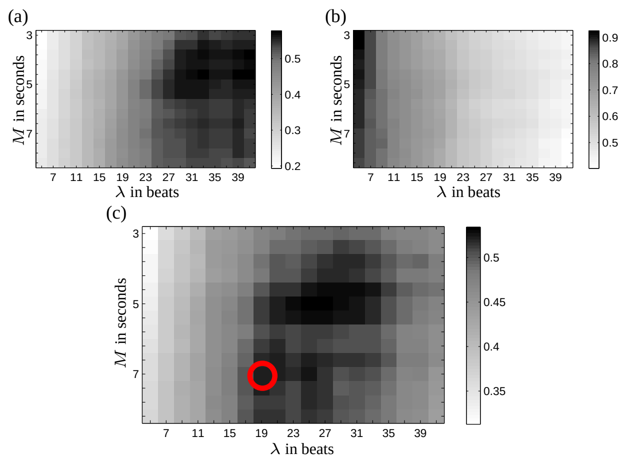


Figure 3: Average cross-version P/R/F-measures for different parameter settings. **(a)** Average precision values. **(b)** Average recall values. **(c)** Average F-measure values. The red circle indicates the parameter setting used in Table 1.

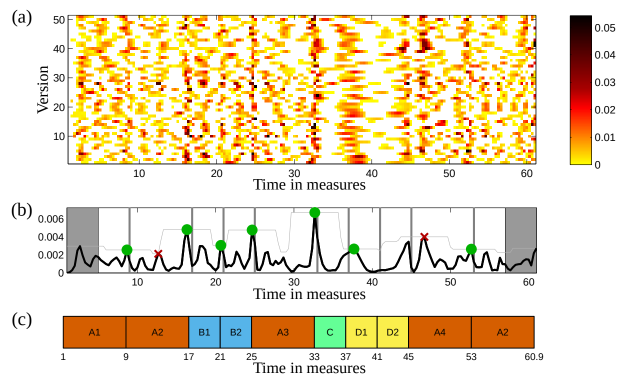


Figure 4: Novelty curves for `M68-3`. **(a)** Individual novelty curves (color-coded, musical time axis) for 51 performances. **(b)** Fusion novelty curve. True peaks are indicated by green discs and false peaks by red crosses. The gray areas at the beginning and end are left out in the evaluation. The thin gray curve indicates the peak picking condition introduced by the neighborhood parameter λ. **(c)** Annotated structure and segment boundaries.

for our main experiments, we favored comparatively larger kernel sizes (resulting in smoother novelty curves) and a smaller λ (being less restrictive in the peak picking) choosing $M \sim 7$ seconds and $\lambda = 19$ beats. However, as also indicated by Figure 3c, the specific paramter setting is not of crucial importance and slightly changing the settings yields similar experimental results.

3.3 Qualitative Evaluation

For some Mazurkas this improvement is significant. For example, for the Mazurka Op. 7 No. 4 shown in Figure 1, the F-measure increases from $F = 0.55$ (individual) to $F = 0.80$ (cross-version). Also for the Mazurka Op. 68 No. 3 (Figure 4) the cross-version fusion approach stabilizes the tempo-based novelty detection improving the F-measure from $F = 0.57$ (individual) to $F = 0.78$ (cross-version).

However, there is also a number of Mazurkas where one has rather low P/R/F-measures—for the individual curves as well as for the fusion novelty curves. For example, for the Mazurka Op. 56 No. 1 shown in Figure 5, the F-measure even decreases from $F = 0.25$ (individual) to $F = 0.15$ (cross-version). In this piece, the annotated segments are rather long in comparison to the other Mazurkas. Listening to the performances reveals that each A-part consists of several phrases, which are shaped by most pianists using a characteristic tempo progression with a slow down and speed up at phrase boundaries. These tempo changes lead to a large number of consistent peaks, which are not reflected by our structure annotations (even though the peaks are musically meaningful) and sometimes also not captured by our peak picking (λ being too restrictive). Also, in the other parts there are a number of false positive peaks of less musical significance. As this Mazurka shows, annotated segment boundaries do not need to go along with tempo changes and, vice versa, musically meaningful tempo changes may also occur within musical parts. Therefore, our quantitative evaluation within the structure analysis context, even though indicating meaningful general tendencies, is an oversimplification.

We now discuss some further typical examples where the fusion novelty curve reveals musically relevant tempo changes that do not concur with segment boundaries. Let us look at the fusion novelty curve for Mazurka Op. 7 No. 2 as shown in Figure 6a. Here one can notice strong peaks in

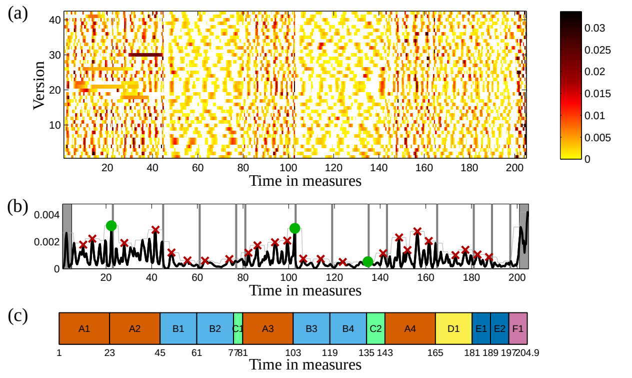


Figure 5: Novelty curves for `M56-1` as in Figure 4. **(a)** Individual novelty curves for 42 performances. **(b)** Fusion novelty curve. **(c)** Annotated structure and segment boundaries.

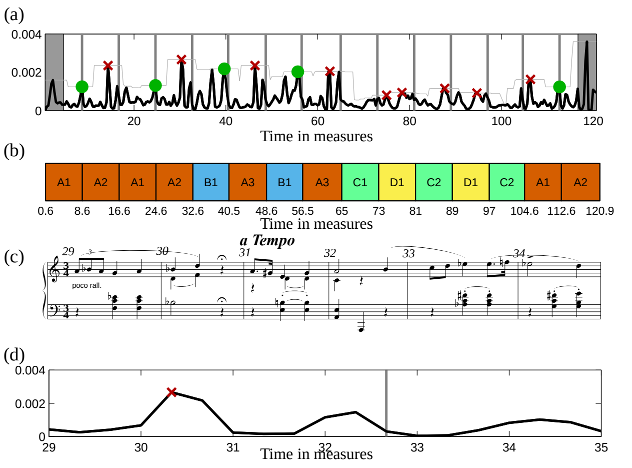


Figure 6: Detailed example on `M07-2`. **(a)** Fusion novelty curve. **(b)** Annotated structure and segment boundaries. **(c)** Score excerpt of measures 29-34. **(d)** Fusion novelty curve excerpt of measures 29-34.

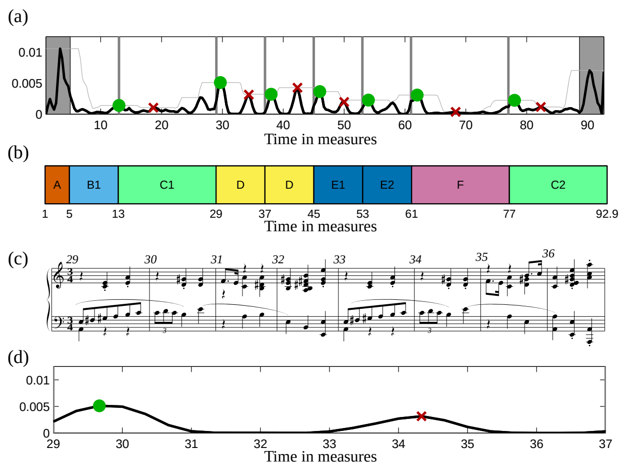


Figure 7: Detailed example on `M56-2`. **(a)** Fusion novelty curve. **(b)** Annotated structure and segment boundaries. **(c)** Score excerpt of measures 29-36. **(d)** Fusion novelty curve excerpt of measures 29-36.

the A_2-parts and A_3-parts located roughly two measures before segment boundaries, so that these peaks are considered false positives in our evaluation. Looking at the score of the piece reveals that there is actually a tempo instruction *a Tempo* just two measures before the respective segments boundaries, see Figure 6c. Most pianists realize this instruction by speeding up their performances, which leads to the musically relevant peaks captured by our cross-version novelty curve. As another example, let us look at the fusion novelty curve of Mazurka Op. 56 No. 2, see Figure 7. Here, two of the false peak positions in the fusion novelty curve are located in the middle of the two D-parts, see Figure 7a. A manual investigation showed that each of the eight-measure D-parts consists of two repeating four-measure phrases. This substructure is not reflected by our structure annotations. The pianists, however, shape the phrases by a pronounced tempo change. Furthermore, in the middle of the C-parts and the F-part, Figure 7 also shows some false positive peaks of no musical relevance. Here, an improved peak picking may remedy this problem.

4. CONCLUSIONS

In this paper, we introduced a cross-version approach for novelty detection capturing consistencies across different performances of a piece. Applying this concept to tempo-related audio features, we showed that the resulting fusion novelty curves perform better in revealing musically meaningful points of novelty than the individual curves. In the future, we plan to conduct similar experiments using different audio features that reflect not only tempo, but also harmony, timbre, and dynamics. Also, the described cross-version approach is generic in the sense that it can also be applied to other music analysis tasks beyond novelty detection. A stabilization effect has also been reported for chord labeling and beat tracking, and we plan to apply this concept to general structure analysis. Finally, we discussed that our evaluation of the novelty detection results based on segment boundaries indicates interesting general tendencies, but also constitutes an oversimplification. Here, future work must address the evaluation problem by including more musicological knowledge, e. g. by looking at expected tempo changes in the score, annotated by musically trained experts. On the other hand, our cross-version approach might not only be used for the task of audio segmentation, but may also aid as a performance analysis tool for musicologists.

Acknowledgments: This work has been supported by the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University and the German Research Foundation (DFG MU 2686/5-1).

5. REFERENCES

[1] Andreas Arzt, Johannes Kepler, and Gerhard Widmer. Simple tempo models for real-time music tracking. In *Proceedings of the Sound and Music Computing Conference (SMC)*, 2010.

[2] Juan Pablo Bello, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, and Mark B. Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047, 2005.

[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] Daniel P.W. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[5] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[6] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 452–455, New York, NY, USA, 2000.

[7] Peter Grosche, Meinard Müller, and Frank Kurth. Cyclic tempogram – a mid-level tempo representation for music signals. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 5522 – 5525, Dallas, Texas, USA, 2010.

[8] Peter Grosche, Meinard Müller, and Craig Stuart Sapp. What makes beat tracking difficult? A case study on Chopin Mazurkas. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 649–654, Utrecht, The Netherlands, 2010.

[9] Kristoffer Jensen. Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal on Advances in Signal Processing*, 2007(1):11 pages, 2007.

[10] Verena Konz, Meinard Müller, and Sebastian Ewert. A multi-perspective evaluation framework for chord recognition. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 9–14, Utrecht, The Netherlands, 2010.

[11] Jouni Paulus and Anssi P. 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.

[12] Jouni Paulus, Meinard Müller, and Anssi P. 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.

[13] Craig Stuart Sapp. Comparative analysis of multiple musical performances. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 497–500, Vienna, Austria, 2007.

[14] Douglas Turnbull, Gert Lanckriet, Elias Pampalk, and Masataka Goto. A supervised approach for detecting boundaries in music using difference features and boosting. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 51–54, Vienna, Austria, 2007.

[15] George Tzanetakis and Perry Cook. Multifeature audio segmentation for browsing and annotation. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 103–106, New Platz, NY, USA, 1999.

FAST IDENTIFICATION OF PIECE AND SCORE POSITION VIA SYMBOLIC FINGERPRINTING

Andreas Arzt[1], **Sebastian Böck**[1], **Gerhard Widmer**[1,2]
[1]Department of Computational Perception, Johannes Kepler University, Linz, Austria
[2]Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria
andreas.arzt@jku.at, sebastian.boeck@jku.at, gerhard.widmer@jku.at

ABSTRACT

In this paper we present a novel algorithm that, given a short snippet of an audio performance (piano music, for the time being), identifies the piece and the score position. Instead of using audio matching methods we propose a combination of a state-of-the-art music transcription algorithm and a new symbolic fingerprinting method. The resulting system is usable in both on-line and off-line scenarios and thus may be of use in many application areas. As the evaluation shows the system operates with only minimal lag and achieves high precision even with very short queries.

1. INTRODUCTION

Over the last few years efficient systems for content-based audio retrieval have been a major topic in music information retrieval research. These systems allow the user to browse and explore large music collections without the need for meta-data and other external information sources. In this context methods to automatically retrieve all pieces (and/or all the excerpts of the pieces) matching a given example query (in the form of a short audio clip) play an important role (and actually are in everyday commercial use). This task, most commonly called *audio identification* or *audio fingerprinting*, can be considered as solved (see e.g., [5, 10]).

Audio identification by definition only finds exact replicas of the query in the database, possibly distorted in some ways (e.g., compression artefacts, noise). Especially for classical music, this is not sufficient, because there are generally large numbers of different performances of the same piece, all different in terms of tempo, expressive timing, and other performance aspects. The relationship between these performances (that they derive from a common musical score) in general goes unnoticed by an audio identification algorithm. In this paper we propose a method for *score identification*: instead of identifying a particular performance it returns the musical score on which the query snippet is based. For example, if we present an audio excerpt of Vladimir Horowitz playing Chopin's Nocturne Op. 55 No. 1 to the system, it will return the name and data of the piece (Nocturne Op. 55 No. 1 by Chopin) rather than the data of the specific performance. Moreover, the system we propose returns not only the corresponding score, but also the exact position within the score. Accordingly, the database for this task does not contain audio recordings, but symbolic representations of musical scores (i.e., to identify the piece being played, the system only uses the symbolic score and has no information about the specific performance by Horowitz in the database).

This task is related to *cover song identification* (see [9] for an overview), where the goal is to identify different versions of one and the same song, in order to detect cover songs in popular music for commercial applications. To perform this task algorithms have to cope with large variations in parameters like timbre, tempo, timing and structure between different performances. Score identification can be seen as a special case of cover song identification. A MIDI version of the score of a classical piece of music can be synthesized and then be treated as a "normal" performance in this task, i.e., performances become "cover songs" of the synthesized version of the score. Still, the difference to our approach to score identification is that for our system very short queries (e.g., 5 seconds) are sufficient, while cover song identification algorithms generally assign similarity values to whole pieces of music.

As an alternative to our symbolic approach *audio matching* can be considered (see e.g., [7]). In this case the score is again first transformed into an audio file (or a suitable in-between representation). Then an audio matching algorithm, most commonly based on dynamic programming techniques, retrieves all excerpts from a database which musically correspond to a short query clip. In contrast to audio fingerprinting methods audio matching can also cope with (non-linear) timing deviations. The downside of audio matching is that in general these methods are very slow compared to fingerprinting methods. To cope with the computational costs, [6] presented clever indexing strategies that greatly reduce the computation time. Still, due to the coarse feature resolution, relatively large query sizes are needed.

Another related task, especially regarding the on-line capabilities of the proposed algorithm, is *score following* (see e.g., [3,8] for state-of-the-art score following systems). In contrast to the algorithm presented in this paper, a score follower needs to know a-priori which piece the performers are playing, and then tracks the on-going performance and continuously returns the current score position. To do so it relies both on access to the complete performance (up to the current point in time) and on the performers closely following the score (i.e., without any additional repeats or any jumps). It contrast to this, the algorithm presented in this paper is able to identify the piece being played, and to identify the actual (or at least an identical) score position from only a small, arbitrary snippet of the performance.

In the following we will describe a new symbolic approach to score identification. Instead of creating an audio representation of the score we create the database directly from the symbolic score information. Then we transform the audio query into a symbolic representation – a list of note onset times with their respective pitch – and use a symbolic fingerprinting algorithm, inspired by the algorithm described in [10], to find matching positions in the score database. This process is very fast, can be used in real-time, on-line applications, and yields very high precision (as can be seen in Section 4). Note that our algorithm involves music (audio) transcription – which is still basically an unsolved problem – as a query preprocessing step. For our system we use a state-of-the-art music transcription system for piano music which, despite many errors, provides us with transcriptions of sufficient quality for the robust symbolic fingerprinting algorithm. Still, this also means that our system currently only works for piano music (because of specific properties of the transcription system).

2. BUILDING THE SCORE DATABASE

Before actually processing queries the score database has to be built. In our system we use deadpan MIDI files as the basis for the score database. The duration of these MIDI files is similar to the duration of a ‘typical’ performance of the respective piece, but without encoded timing variations. From these files a simple ordered list of note events is extracted where for each note event the exact time in seconds and the pitch as MIDI note number is stored.

Next, for each piece fingerprint tokens are generated. In contrast to [10] we create them from 3 successive events according to some constraints (also see Figure 1) to make them tempo independent. Given a fixed event e we pair it with the first n_1 events with a distance of at least d seconds “in the future” of e. This results in n_1 event pairs. For each of these pairs we then repeat this step and again pair them with the n_2 future events with a distance of at least d seconds. This finally results in $n_1 * n_2$ event triplets. In our experiments we used the values $d = 0.05$ seconds and $n_1 = n_2 = 5$. Also inspired by [10] we further constrain the pair creation steps to notes which are at most 2 octaves apart.

Given such a triplet consisting of the events e_1, e_2 and e_3 the time difference $td_{1,2}$ between e_1 and e_2 and the time difference $td_{2,3}$ between e_2 and e_3 are computed. To get a tempo independent fingerprint token we compute the time difference ratio of the time differences: $tdr = \frac{td_{2,3}}{td_{1,2}}$.

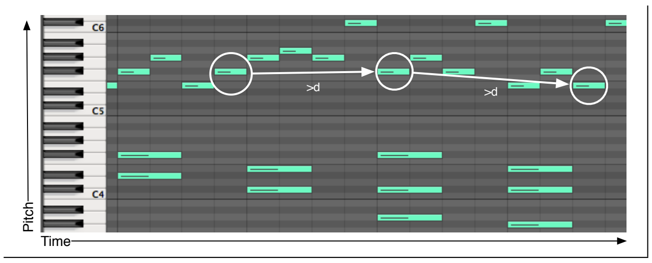


Figure 1. Fingerprint Token Generation: Example of 1 generated Token

This finally leads to a fingerprint token $[pitch_1 : pitch_2 : pitch_3 : tdr] : pieceID : time : td_{1,2}$, where the hash key $[pitch_1 : pitch_2 : pitch_3 : tdr]$ can be stored in a 32 bit integer. The purpose of storing $td_{1,2}$ in the fingerprint token will be explained in the description of the search process itself (see Section 3.2 below).

The result of the score preprocessing is our score database; a container of fingerprint tokens which provides quick access to the tokens via hash keys.

3. QUERYING THE DATABASE

3.1 Preprocessing: Transcribing the Query

Before querying the database the query (an audio snippet of a performance) has to be transformed into a symbolic representation. The algorithm we use to transcribe musical note onsets from an audio signal is based on the system described in [2], which exhibits state-of-the-art performance for this task. It uses a recurrent neural network to simultaneously detect the pitches and the onsets of the notes (see Figure 2 for an illustration of the algorithm).

For its input, a discretely sampled audio signal is split into overlapping blocks before it is transferred to the frequency domain with two parallel Short-Time Fourier Transforms (STFT). Two different window lengths have been chosen to achieve both a good temporal precision and a sufficient frequency resolution for the transcription of the notes. Phase information of the resulting complex spectrogram is discarded and only the logarithm of the magnitude values is used for further processing. To reduce the dimensionality of the input vector for the neural network, the spectrogram representation is filtered with a bank of filters whose frequencies are equally spaced on a logarithmic frequency scale and are aligned according to the MIDI pitches. The attack phase of a note onset is characterized by a rise of energy, thus the first order differences of the two spectrograms are used as additional inputs to the neural network.

The neural network consists of a linear input layer with 324 units, three bidirectional fully connected recurrent hidden layers, and a regression output layer with 88 units, which directly represent the MIDI pitches. Each of the hidden layers uses 88 neurons with hyperbolic tangent activation function. The use of bidirectional hidden layers enables the system to better model the context of the notes, which show a very characteristic envelope during their de-

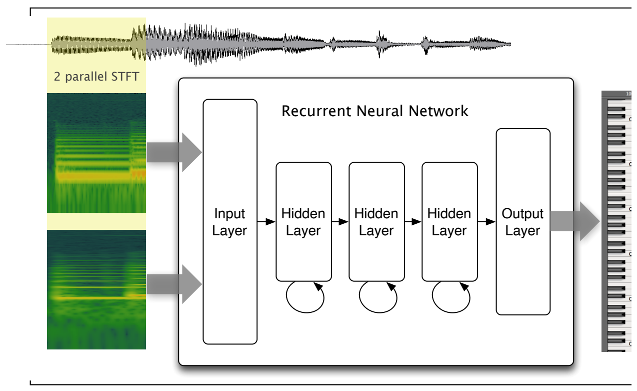


Figure 2. The Transcription System

Detection Window	Precision	Recall	F-measur
20 ms	0.586	0.489	0.533
40 ms	0.812	0.678	0.739
60 ms	0.851	0.710	0.774
80 ms	0.864	0.720	0.786
100 ms	0.869	0.725	0.790

Table 1. Results of the On-line Transcription Algorith for different detection window sizes.

cay phase.

The network is trained with supervised learning and e stopping. The network weights are initialized with r dom values following a Gaussian distribution with mea and standard deviation 0.1. Standard gradient descent w backpropagation of the errors is used for training. T network was trained on a collection of 281 piano pie recorded on various pianos, virtual and real (seven c ferent synthesizers, an upright Yamaha Disklavier, and a Bösendorfer SE grand piano).

To make the transcriber applicable also in on-line scenarios, instead of preprocessing the whole piece of audio at a time, the signal is split into blocks of 11 frames centered around the actual frame. The use of 11 frames is a trade-off between keeping the system's ability to model the context of the notes and to keep the introduced delay at a minimum. In the current system the constant lag caused by the query preprocessing amounts to about 210 ms.

Table 1 shows the on-line transcription results for the complete test set described later on in Section 4.1. A note is considered to have been discovered correctly if its position is detected within a 'detection window' of given size around the annotated ground truth position. As can be seen in the table, the results are far from perfect (though they are very good, considering the state of the art). If the proposed fingerprinting system is used in an off-line scenario, the use of an off-line transcription algorithm is an option to slightly improve the results.

3.2 Querying the Database

The transcription of the query results in a list of note pitches with timestamps. This list is then processed in the same way as described in Section 2 above to produce query tokens. Of course in this case no piece ID is known and furthermore each query starts at time 0. These query fingerprint tokens are now used to query the database. The method described below is again very much inspired by the audio fingerprinting method proposed in [10].

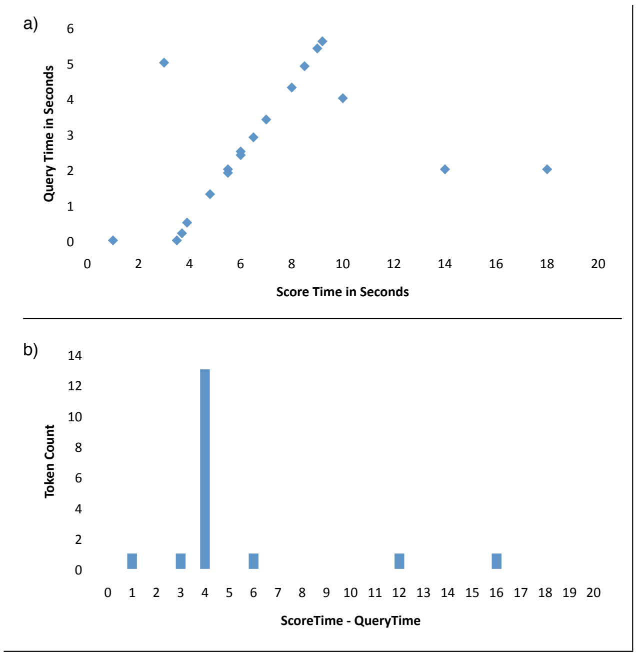


Figure 3. a) scatter plot of matching tokens and b) computed histogram for diagonal identification

The general idea is to find regions in the score database which share a continuous sequence of tokens with the query. To do so first all the score tokens which match the query tokens are extracted from the database. When plotted as a scatter plot against their respective time stamps (see Figure 3a) matches will be indicated by (rough) diagonals (i.e., these indicate that the query tokens match the score tokens over a period of time). As identifying these diagonals directly would be computationally expensive we instead use a simpler method described in [10]. This is based on histograms (one for each piece in the score database, with a time resolution of 1 second) into which the matched tokens are sorted in a way such that peaks appear at the start points of these diagonals (i.e., the start point of a query, see Figure 3b). This is achieved by computing the bin to sort the token into as the difference between the time of the score token and time of the query token. The complete process will be explained in more detail below.

For each of the query tokens qt with $[qpitch_1 : qpitch_2 : qpitch_3 : qtdr] : qtime : qtd_{1,2}$ the following process is repeated. First, matching tokens are extracted from the score database via the hash key. To allow for local tempo differences we permit the normalized time difference to be within $\frac{1}{4}$ of $qtdr$. This normally results in a large number of score tokens $[spitch_1 : spitch_2 : spitch_3 : stdr] : spieceID : stime : std_{1,2}$. Unfortunately directly sorting these tokens into bin $round(stime - qtime)$ of the histogram $spieceID$ does not necessarily make sense because of the query possibly having a different tempo than

Data Description	Number of Pieces	Notes in Score	Notes in Performance	Performance Duration
Chopin Corpus	154	325,263	326,501	9:38:36
Mozart Corpus	13	42,049	42,095	1:23:56
Additional Pieces	16	68,358	–	–
Total	183	435,670		

Table 2. Pieces in Database

	Query Length in Notes						
	5	10	20	30	40	50	60
Corr. Piece as Top Match	22.55%	78.33%	94.07%	96.70%	97.50%	98.01%	98.42%
Corr. Piece in Top 2	29.23%	83.22%	96.07%	97.67%	98.28%	98.64%	98.87%
Corr. Piece in Top 3	33.00%	85.50%	96.74%	98.12%	98.57%	98.91%	99.09%
Corr. Piece in Top 4	35.33%	86.88%	97.15%	98.32%	98.76%	99.09%	99.22%
Corr. Piece in Top 5	37.24%	87.86%	97.44%	98.49%	98.87%	99.17%	99.32%
Corr. Position as Top Match	14.41%	60.47%	80.35%	84.63%	84.86%	83.91%	83.70%
Corr. Position in Top 2	21.94%	75.09%	91.11%	93.39%	93.77%	93.39%	93.17%
Corr. Position in Top 3	25.77%	79.70%	93.69%	95.36%	95.73%	95.85%	95.84%
Corr. Position in Top 4	28.20%	81.94%	94.69%	96.14%	96.61%	96.84%	96.93%
Corr. Position in Top 5	30.02%	83.29%	95.22%	96.55%	97.05%	97.34%	97.47%
Mean Query Duration	0.60 sec	1.33 sec	2.78 sec	4.21 sec	5.63 sec	7.04 sec	8.48 sec
Mean Query Exec. Time	1.71 ms	5.13 ms	11.76 ms	16.86 ms	20.76 ms	26.36 ms	31.89 ms

Table 3. Results of the proposed piece and score position identification algorithm on the test database. Each estimate is based on 50,000 random audio queries.

expected by the score.

As an illustration let us assume a slower tempo for the query than for the respective score. Then the diagonal in Figure 3a would be steeper and when computing the bins via $round(stime - qtime)$ the first few tokens may fall into the correct bins. But soon the tokens, despite belonging to the same score position, would get sorted into lower bins instead.

Thus we first try to adapt the timing by estimating the tempo difference between the score token and the query token. First we compute the tempo ratio of both tokens $r = \frac{std_{1,2}}{qtd_{1,2}}$ and then adapt the time of the query event when computing the bin to sort the token into: $bin = round(stime - qtime * r)$.

We now have a number of histograms, one for each score in the database, and need a way of deciding on the most probable score position(s) (and, by implication, the most probable piece), for the query. We did experiments with different methods of computing the matching score but in the end simply taking the number of tokens in each bin as the score produced the best results.

4. EVALUATION

4.1 Dataset Description

For the evaluation of our algorithm a ground truth is needed. We need exact alignments of performances of classical music to their respective scores such that we know exactly when each note given in the score is actually played in the performance. This data can either be generated by a computer program or by extensive manual annotation but both ways are prone to annotation errors.

Luckily, we have access to two unique datasets where professional pianists played their performances on a computer-controlled piano [1] and thus every action (e.g., key presses, pedal movements) was recorded in a symbolic way. The first dataset (described in [11]) consists of performances of the first movements of 13 Mozart piano sonatas by Roland Batik. The second, much larger, dataset consists of nearly the complete solo piano works by Chopin performed by Nikita Magaloff [4]. For the latter set we do not have the original audio files and thus replayed the symbolic performance data on a Yamaha N2 hybrid piano and recorded the resulting performances.

As we have both symbolic and audio information about the performances, we know the exact timing of each played note in the audio files. The performances were manually aligned to electronic (symbolic) versions of the original sheet music. To build the score database we converted the sheet music to MIDI files with a constant tempo such that the overall duration of the file is similar to a 'normal' performance of the piece.

In addition to these two datasets we added some more scores to the database, solely to provide for more diversity and to make the task even harder for our algorithm (these include, amongst others, the Beethoven Symphony No. 5, the Mozart Oboe Quartet KV370, the First Mephisto Waltz by Liszt and Schoenberg Op. 23 No. 3). To the latter, we have no ground truth, but this is irrelevant since we

[1] Bösendorfer SE 290

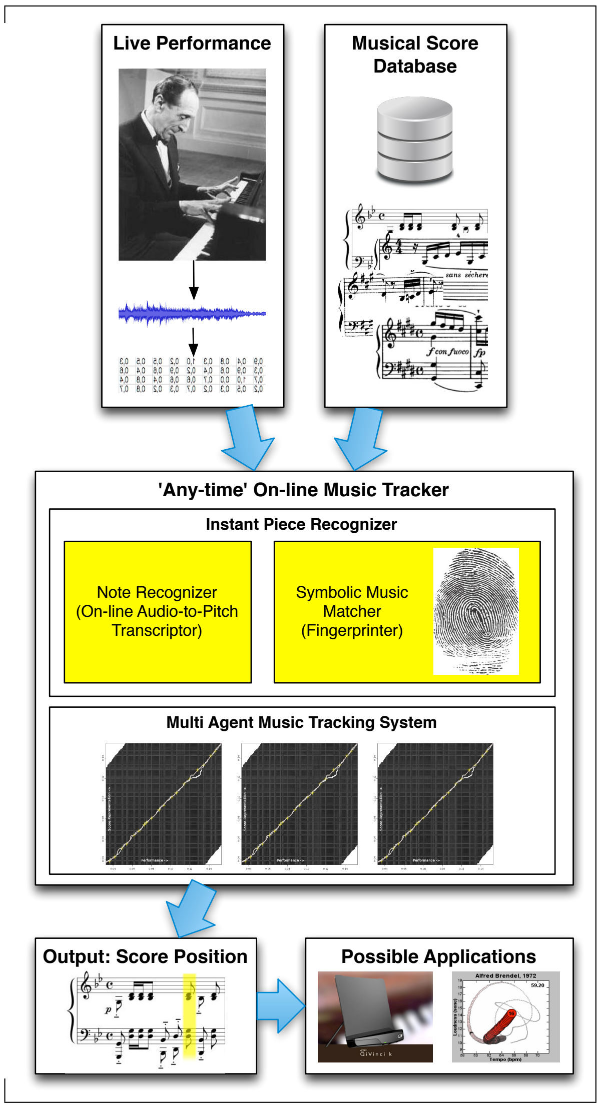

Live Performance
Musical Score Database
'Any-time' On-line Music Tracker
Instant Piece Recognizer
Note Recognizer (On-line Audio-to-Pitch Transcriptor)
Symbolic Music Matcher (Fingerprinter)
Multi Agent Music Tracking System
Output: Score Position
Possible Applications

Figure 4. The Ultimate Music Companion

do not actively query for them with performance data in our evaluation runs [2]. See Table 2 for an overview of the complete dataset.

4.2 Results

An evaluation of the transcription stage (query preprocessing) was already presented in Section 3.1 above. As Table 1 shows the results of this stage are rather noisy. Still, the quality of the transcription is sufficient to be used with our robust fingerprinting technique.

We tested the algorithm with different query lengths: 5, 10, 20, 30, 40, 50 and 60 notes (in number of transcribed notes during the preprocessing step). For each of the query lengths, we generated 50,000 queries by picking random points in the performances of our test database, and used them as input for the proposed algorithm.

[2] Additionally we performed some non-systematic experiments with data from different sources (e.g., Youtube videos, both by amateurs and by professional pianists, with differing recording qualities (including noisy 'old' recordings and noisy amateur recordings)), for which we have no ground truth data. The general impression is that the system works well too in these scenarios, but of course the performance worsens in the presence of noise.

	Score Tempo		
	Normal	Double	Half
Corr. Pos. as Top M.	84.63%	83.30%	85.15%
Corr. Pos. in Top 2	93.39%	92.61%	92.70%
Corr. Pos. in Top 3	95.36%	95.06%	94.42%
Corr. Pos. in Top 4	96.14%	96.11%	95.00%
Corr. Pos. in Top 5	96.55%	96.55%	95.26%

Table 4. Results of the algorithm with score representations with altered tempi. The results are based on query lengths of 30 notes.

The results of this experiment are shown in Table 3. In this table we present two measures: the percentage of correctly identified pieces, and the percentage of cases in which both the piece and the exact position in the score were correctly identified.

For the evaluation a score position X is considered correct if it marks the beginning (+/- 1.5 seconds) of a score section that is identical in note content, over a time span the length of the query (but at least 30 notes), to the note content of the 'real' score situation corresponding to the audio segment that the system was just listening to (we can establish this as we have the correct alignment between performance time and score positions — our *ground truth*). This complex definition is necessary because musical pieces may contain repeated sections or phrases, and it is impossible for the system (or anyone else, for that matter) to guess the 'true' one out of a set of identical passages matching the current performance snippet, given just that performance snippet as input. We acknowledge that a measurement of musical time in a score in terms of seconds is rather unusual. But as the MIDI tempos in our database generally are set in a meaningful way, this seemed the best decision to make errors comparable over different pieces, with different time signatures – it would not be very meaningful to, e.g., compare errors in bars or beats over different pieces.

As can be seen, even queries of only a length of 5 notes lead to a surprising number of correct position identifications, and already for a query length of 20 notes (which corresponds to a mean query duration of 2.78 seconds) the correct position in the score is contained in the top 5 matches for more than 95% of the cases.

To show the tempo independence of our method we also ran experiments with big tempo differences between the score and the performance. We simulated this by manipulating the scores to have double and half the original tempo. The results for these experiments are shown in Table 4. As can be seen the performance only decreases slightly and the proposed algorithm still recognizes the correct position in the vast majority of the cases.

A flaw of the current approach is that it cannot cope with non-linear tempo deviations (i.e., with tempo variations *within* a query). As we are using very short queries and a rather coarse resolution in the histograms this is only a minor problem. But for longer queries (e.g., with durations of over 10 seconds) explicitly dealing with non-linear tempo deviations becomes more of an issue. To make our approach useable with longer queries we propose to split the long query into smaller, overlapping queries (e.g., of size 30 notes with 15 notes overlap) and then use some simple tracking and scoring algorithm to combine the individual results into a single score. Preliminary experiments with this approach suggest that this leads to a very robust and accurate algorithm.

5. CONCLUSION

5.1 Applications

The proposed system is useful in a wide range of applications. For the off-line case it may either be used standalone or as a preprocessing step to audio alignment algorithms. This enables fast and robust (inter- and intra-document) searching and browsing in large collections of musical scores and corresponding performances.

Originally this work was motivated by an on-line scenario (see [1]). In connection with an on-line score following algorithm we are currently building a system that we somewhat immodestly call 'the ultimate classical music companion' (see Figure 4 for a sketch of the system). This system will be able to recognize arbitrary pieces of classical music, immediately identify the position in the score, provide meta-information, track the piece via an on-line score following algorithm, display the score, and visualize musical aspects of the performance like the structure of the piece, tempo and expressive timing deviations – all done on-line with minimal delay. This ultimate music companion may be used to enhance the listening experience but may also be of use for performers, especially during rehearsal. Besides this very specific application the proposed algorithm may also be of use in any application which requires monitoring of an audio stream of classical music on-line with minimal delay.

5.2 Future Work

Regarding the improvement of the algorithm, we see some future work in making it better useable with longer queries including tempo variations. As already mentioned above a combination of small overlapping subqueries with simple tracking of the results seems to be the way to go. Also a larger scale evaluation of the algorithm (with thousands of classical piano scores) has to be done.

The algorithm in the current state is able to recognize the correct piece and the score position even for very short queries of piano music. Based on this algorithm we have already implemented a real-time on-line piece and score position identification system that shows a level of performance that even human experts in classical music will find hard to match. This will be demonstrated live at the conference. We are now working on the integration of the proposed algorithm in our score following system as a next step towards our ultimate goal: 'the ultimate classical music companion'.

6. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Fund (FWF) under project numbers Z159 and P22856-N23.

7. REFERENCES

[1] A. Arzt, G. Widmer, S. Böck, R. Sonnleitner, and H. Frostel. Towards a complete classical music companion. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 2012.

[2] S. Böck and M. Schedl. Polyphonic piano note transcription with recurrent neural networks. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2012.

[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] S. Flossmann, W. Goebl, M. Grachten, B. Niedermayer, and G. Widmer. The Magaloff project: An interim report. *Journal of New Music Research*, 39(4):363–377, 2010.

[5] J. Haitsma and T. Kalker. A highly robust audio fingerprinting system. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, 2002.

[6] F. Kurth and M. Müller. Efficient index-based audio matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):382–395, 2008.

[7] M. Müller, F. Kurth, and M. Clausen. Audio matching via chroma-based statistical features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2005.

[8] C. Raphael. Current directions with music plus one. In *Proceedings of the Sound and Music Computing Conference (SMC)*, 2009.

[9] J. Serra, E. Gómez, and P. Herrera. Audio cover song identification and similarity: background, approaches, evaluation, and beyond. In Z. W. Ras and A. A. Wieczorkowska, editors, *Advances in Music Information Retrieval*, pages 307–332. Springer, 2010.

[10] A. Wang. An industrial strength audio search algorithm. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2003.

[11] G. Widmer. Discovering simple rules in complex data: A meta-learning algorithm and some surprising musical discoveries. *Artificial Intelligence*, 146(2):129–148, 2003.

A MULTIMEDIA SEARCH AND NAVIGATION PROTOTYPE, INCLUDING MUSIC AND VIDEO-CLIPS

G. Peeters, F. Cornu
Ch. Charbuillet, D. Tardieu, J.J. Burred
STMS IRCAM-CNRS-UPMC
{peeters,cornu}@ircam.fr

M. Vian[1], V. Botherel[2]
J.-B. Rault[2] and J.-Ph. Cabanal[2]
[1]Bertin Technologies, [2]Orange-Labs
jeanphilippe.cabanal@orange.com

ABSTRACT

Moving music indexing technologies developed in a research lab to their integration and use in the context of a third-party search and navigation engine that indexes music files, archives of TV music programs and video-clips, involves a set of choices and works that we relate here. First one has to choose technologies that perform well, which are scalable (in terms of computation time of extraction and item comparison for search-by-similarity), and which are not sensitive to media quality (being able to process equally music files or audio tracks from video archives). These technologies must be applied to estimate tags chosen to be understandable and useful for users (the specific genre and mood tags or other content-descriptions). For training the related technologies, relevant and reliable annotated corpus must be created. For using them, relevant user-scenarios must be created and friendly Graphical User-Interface designed. In this paper, we share the experience we had in a recent project on integrating six state-of-the-art music-indexing technologies in a multimedia search and navigation prototype.

1. INTRODUCTION

The objective of the MSSE project (Multimedia Search Services for European Portals) is to develop a multimedia search and navigation prototype, which gives access to several types of contents (catch-up TV, archives, videos, music) and which illustrates the benefits of advanced audio-video analysis technologies. The prototype is organized around three use-cases:

- Searching recently broadcasted TV programs ("Catch-up TV"); navigating inside the videos by chapters or keywords.
- Searching video extracts on a specific topic related to recent news and culture; browsing in relevant translated foreign videos and in public TV archives.
- Searching and exploring music pieces with the help of tags, music structure, summaries and similarity.

The prototype is based on video indexing, speech recognition and music indexing technologies. In this paper, we describe the works performed for the music indexing technologies. Those have to deal with three types of content:

- A music collection
- A collection of video clips from the W9 TV channel
- A collection of video archives of music programs from the INA [1] collection. (only the audio track of the video is processed by our indexing modules).

In this paper, we propose to review the technologies integrated into this search and navigation prototype, why they were chosen and how they were developed and integrated as well as the corresponding user-evaluations and GUI developed. We believe that sharing the experience of this work could provide a good example of integration of research modules in a real application scenario.

While many papers have been published on the independent elements this paper deals with (content-based, semantic tags, corpus creation, GUI, user-tests), few of them deal with all these elements as a whole to create a system. Among exceptions are the works made for the Music Browser [1], FM4-Soundpark [2], Musicream [3], MusicBox [4] or PlaySOM [5]. Our work differs from the previous in the number of integrated technologies, the integration into a whole video and music search engine accessible through a web-browser and the simplicity of the GUI.

2. OVERALL DESIGN PROCESS

Figure 1 represents the various elements of work (and interaction/dependency between them), needed for integrating the music technologies in the prototype.

The starting point is a set of requirements from the third-party developer and its users [2] −in terms of functionalities (such as searching-by/ filtering-by tags, search-by-similarity or summarization) and −in terms of types of content-description (genre, mood, instrumentation).

From this, a set of potential technologies are studied in terms of performances and scalability [3] . Candidate technologies are tested over the years in internal benchmarking or in public ones such as MIREX. For example, from

[1] French National Audio Visual Archiving Institution

[2] During the project, 1 or 2 user tests per year were performed, each corresponding to a new version of the prototype (see part 6).

[3] By scalability we mean computation time of content-extraction and of items comparison for search-by-similarity.

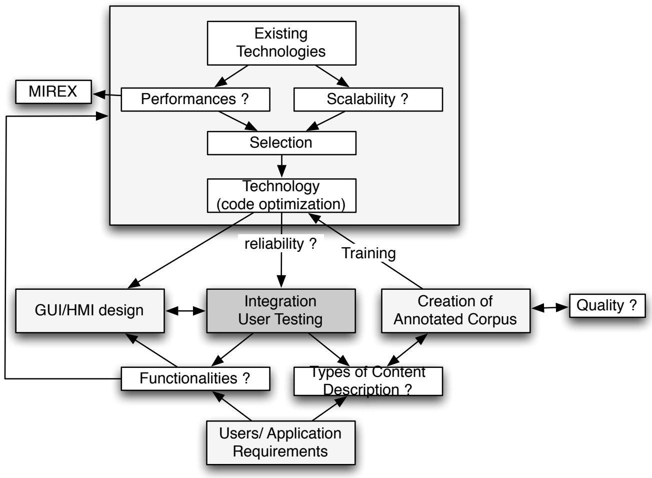


Figure 1. Interaction/dependency between the various elements of work needed integrating music technologies.

our tests in MIREX between 2008-2011, it appears that using Universal Background Model (UBM) to model audio features [6] has many advantages over other techniques: it achieves performances among the best for both auto-tagging [7] and similarity tasks [8]; it allows to share the same front-end for both tasks; it allows easy scalability in the case of similarity (items comparison remains in an Euclidean space). Therefore, we chose UBM for these tasks.

In parallel, the design of the GUI starts. Since the GUI directly infers on the usability of the functionalities, its design is mainly driven by those. It is also driven by the outputs of user-tests and by extra outputs that technologies can provide without extra-costs. For example, when computing audio summaries, music structures are estimated as an internal step. Therefore, it can easily be integrated to provide new functionalities (display interactive player).

In a latter stage, annotated corpora need to be created for each of the requested content description (genre, mood, instrumentation). This part forms a close feedback loop between: – annotation of a corpus, – measuring the reliability of the annotations (this can highlight the fact that some required concepts may appear unclear), – redefining the types of content with the third-party. After several iterations, this loop-process leads to a much clearer set of content-description concepts (the specific definition of genre, mood, instrumentation) and more accurate annotated corpora (their specific use for music tracks).

These annotated corpora are then used to the train the corresponding technologies and optimization is performed to reduce computation time, disk access and memory load.

The resulting prototype is then submitted to global user tests (testing both functionalities, the GUI and the underlying technologies to achieve the functionalities). The whole process is then started again (once a year in our project).

3. TECHNOLOGIES INTEGRATED

Resulting from the process explained in part 2, six different music-content-based technologies have been selected:

- auto-tagging based on training (for genre, mood, instrumentation tags and singing segmentation),
- two technologies for auto-tagging based on dedicated models (for tempo and key/mode tags),
- search by similarity (for music recommendation),
- music structure (for interactive browsing),
- audio summary creation (for content preview).

These modules are either applied to mp3 files or to the audio part of video archives or clips. The inter-connections between the various modules are indicated in Figure 2. It should be noted that the first five technologies were evaluated very positively in the recent MIREX-11 evaluations.

3.1 Audio feature extraction

In order to decrease the total computation time, auto-tagging based on training and search-by-similarity are based on the same audio features front-end. The audio features front-end is described in Figure 2 and corresponds to the proposals made in [8], [7] or [9]. It is based on two modeling techniques coming from speech processing:

• **Universal Background Model (UBM)** [6] [10]. The aim of this technique is to represent the “world” of features using a GMM and then deform[4] this “world” to represent a new feature vector. The resulting representation is the concatenation of the adapted μ-vectors of the GMM, the size of which depends on the dimensionality D of the initial feature vectors and the number m of mixtures used for the GMM. These concatenated-vectors are denoted by “Super-Vectors” (SV) in the following.

• **Multivariate Auto-Regressive Model (MAR)** [11]. As for the mono-dimensional AR-model, the goal is to represent the dependency of the values of a signal over time by an all-pole filter of order K[5]. In the case of the MAR, we consider the dependencies in time and between the various D dimensions of the feature vectors. The results of this is a matrix of coefficients $\underline{\underline{\alpha}}_{k,d}$.

The input to these two modeling techniques is a feature set made of 13 Mel Frequency Cepstral Coefficients and 4 Spectral Flatness Measure coefficients, extracted using a 40 ms Blackman window with a 20ms hop size. From those, two modeled feature sets are computed: (1) Super-Vector of MFCC/SFM, which we denote by SV(mfcc/sfm), (2) MAR of MFCC/SFM, which we denote by MAR(mfcc/sfm). The two modelings are performed using – either the whole set of features inside a track (in case of search-by-similarity and global auto-tagging) – or the set of feature inside successive windows of 2s duration (in case of segmentation, such as singing voice location). In each case, the UBM has been previously trained on a representative database. This training is the most time-consuming part but needs only to be performed once. The UBM configuration is a set of $m = 64$ (for search-by-similarity) or $m = 32$ (for auto-tagging) mixtures, each with a diagonal covariance matrix. The order of the MAR model is $K = 4$.

[4] Deforming means here adapting the μ-vectors of the GMM using an Expectation Maximization algorithm.

[5] $s(n) = \sum_{k=1}^{K} \alpha_k s(n-k) + \epsilon$ where s is a signal, n discrete time, ϵ a residual.

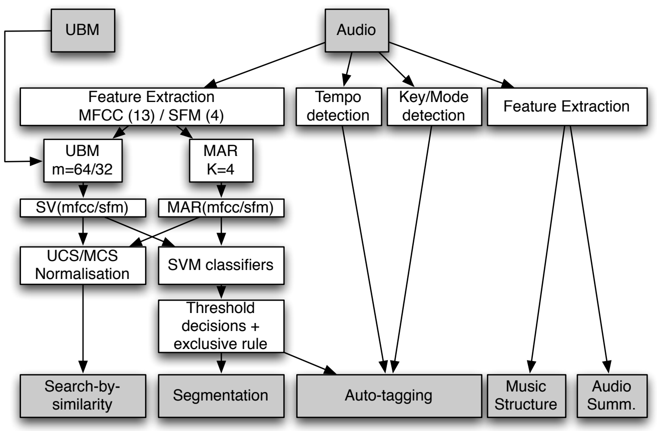


Figure 2. Modules used for music content extraction.

Auto-tagging based on training		
Categories	**Configuration**	**Tags**
Genre	single-label	Classical, Other Genres, NA
Other Genres	multi label	Pop/Rock, Blues, Electronica, Metal/Punk, Reggae, Jazz, Rap, Soul/Funk, Rhythm & Blues, Latin/Bossa
Mood	single label single label single label	Happy, Sad, NA Dynamic, Calm, NA Romantic, NA
Instrumentation	multi label	Brass, String, Piano, Electronic, Acoustic
Drum Kit	single label	No drum, Electronic, Pop/Rock, Hard/Metal
Guitar	single label	No guitar, Acoustic Guitar, Electric Guitar
Live Studio	segmentation + single label	Live, Studio
Singing	segmentation	Singing voice

Table 1. Categories, configurations and tags of the various classifiers used for the Auto-tagging modules

3.2 Search by similarity

As explained in [8], the main goal of using UBM and MAR for modeling the features (instead of the usual MFCC/GMM with EMD Kullback-Leibler divergence) is to remain in an Euclidean space. In the case of search by similarity, it therefore allows the use of standard techniques to decrease the search time in the database. In order to avoid hubs and orphans, various techniques have also been proposed. We have used the UCS-norm (UBM Centered Spherical normalization) and the MCS-norm (Mean Centered Spherical) proposed in [8]. Both techniques consist in projecting the features vector on a unit sphere (either centered on the mean of the UBM, or the mean of the database). After this, each track of the database sees the rest of the database with the same point of view (unit sphere). Using those, the similarity between two tracks is simply the correlation of their vectors. Two similarity matrices, corresponding to the two feature sets are then computed and combined linearly (late-fusion).

3.3 Auto-tagging based on training

Auto-tagging based on training aims at providing the tags indicated in Table 1. Tags can be exclusive (such as "dynamic" and "calm") or inclusive ("pop/rock" and "electronica"). In our system, all problems are solved using multi-label classifiers in a one-against-all strategy (true versus false class). For this, all problems are decomposed as set of binary SVM classifiers (with an RBF-kernel, σ=1) [12] [7]. The input to the classifiers is the concatenation of SV(mfcc/sfm) and MAR(mfcc/sfm) (early fusion).

Global Classifiers: Music **genre** classifier is a set of 11 binary classifiers (one for each genre) trained and evaluated independently. A given track t is said to belong to a tag-class c if the affinity-output $a_c(t)$ of the corresponding SVM classifier is above a threshold A_c. The estimation of each threshold A_c is based on the Recall/Precision curve obtained on a training set. Considering that the estimated tags are to be used as search criteria, it was decided to favor Precision over Recall: we chose the lowest A_c leading to a Precision greater than 0.8. In terms of usability, we also decided to make "classical music" mutually exclusive to the "other genres" (see Table 1). For a given track t, if both $a_{class}(t)$ and several $a_{other}(t)$ are above their respective threshold, the choice is based on the maximum between $a_{class}(t)$ and $\max(a_{other}(t))$. In case $\max(a_{other}(t))$ is selected, the corresponding subgenres above their respective thresholds are returned. The same process is applied for the 5 **mood** classifiers. In this case, the mutually exclusive classes are "happy" / "sad" and "calm" / "dynamic". The auto-tagging module also returns three view-points related to the **instrumentation** of the track: (1) a global instrumentation based on dominant instruments (brass, string, piano, electronic instruments, acoustic instruments), (2) a detailed description of the percussive part (electronic drum, pop/rock drum, hard/metal drum) (3) a detailed description of the guitar part (acoustic guitar, electric guitar).

Segmentation: The segmentation is obtained by detecting class-changes over time. For this, the same system as presented above is used, but the UBM/MAR models are applied to the set of features inside a succession of windows of 2s duration (hop size of 1s). Each 2s features is then classified using SVM classifiers. This segmentation is used to provide **singing/**non-singing segmentation over time. In order to avoid spurious class transitions over time, a 3rd-order median filter is applied to the estimated classes over time before segmentation. This segmentation is to be used to display singing segments in the interface.
We also use this segmentation to perform the **"live/studio"** auto-tagging. In our case, "live" is defined as the presence of "applauses, whistling ..." of audience in a bar, concert-hall, stadium. Since those do not occur over the whole time-duration of the track (usually at the beginning, ending or during a break), the decision is based on frame-classification. We use a minimum threshold of 26s frames being classified as "live" for the track to be classified as "live". A similar approach has been used in [13].

Each tag has also an associated "reliability" defined in the interval $[0, 1]$ (low/high reliability). For this, the affinity of each SVM is passed through a sigmoid and centered on its respective threshold. This reliability is to be used by the GUI for sorting the list of results.

3.4 Auto-tagging based on dedicated algorithms

For each track, we also estimate its global **tempo** in beats-per-minute. Note that this estimation does not rely on the set of UBM/MAR features but on a dedicated algorithm. We have used the algorithm proposed in [14]. We also assign a "reliability" to this estimated tempo. For this we used the "periodicity" features proposed by [15] (measurements of the amount of periodicity in the track).

We also estimate the global (most dominant over time) **key/mode** among a set of 24 key/mode classes (C Maj, C min . . . B Maj, B min). We have used the algorithm proposed in [16]. The "reliability" of the output is here estimated as the distance between the most-likely key/mode and the second most likely.

3.5 Music Structure and Summary

This module aims at providing two functionalities: (1) to display a map of the temporal organization of the track (music structure) which allows user to interact with it (skip forward/backward by parts) [17], (2) to provide a meaningful preview of the track content (music audio summary). The estimation of the music structure and of the audio summary are based on the same front-end. This front-end combines the three similarity matrices corresponding to MFCC, Spectral-Contrast and Spectral-Valley [18] measures and Chroma/Pitch-Class-Profile (see [19] for details).

Music Structure Estimation: For robustness reasons, the structure is estimated using a "state" approach. For this, a segmentation of the similarity matrix is first performed using a "checker-board" kernel [20]. The segments obtained are then grouped using a constrained hierarchical agglomerative clustering. The distance used for this clustering is a linear combination of − the distance between the average values inside the two segments (centroid linkage) − the smallest possible distance between one of the diagonals they may contain (sequence approach) − a constraint to minimize the departure of the duration of the merged segments from the average segment durations.

Music Audio Summary Generation: The technique used for the summary creation is based on an extension of the summary score of [21]. In this extension, the method of [21] is iteratively applied to the combined matrix of [19]. At each iteration, the two time corridors in the self-similarity-matrix corresponding to the previously chosen audio extract are canceled to avoid further re-uses. To generate the final audio signal, the selected segments are concatenated using a Downbeat Synchronous OverLap-Add (DSOLA) techniques.

4. ANNOTATED CORPORA FOR TRAINING

4.1 Corpus creation for the UBM training

Since both auto-tagging and search-by-similarity modules rely on Super-Vectors, the corresponding UBM needs to be trained in advance. The training of which needs to take into account the various types of contents (various genres and various audio qualities) that the system will need to deal with. For this, a large database of audio files has been used: including clean mp3 files at various bit-rates and audio tracks of TV archives.

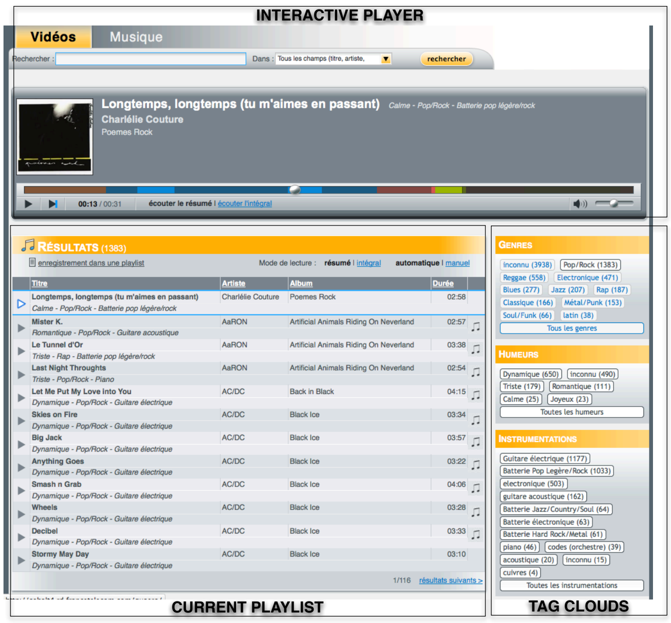
4.2 Annotated corpora for the auto-tagging problems

For the auto-tagging modules, statistical models (SVMs and related thresholds) need to be trained for each tag (genre, mood, instrumentation, singing, live). We explain here the data used for the training. For the creation of the list of **genres**, several attempts have been made: − from a purely acoustic definition of genres (pop-rock synthed, poprock hard, electronica ambient, electronica beat . . .) which guarantees a close proximity to content-based estimation algorithms but may be difficult to understand by users − to a purely application oriented definition. The final list indicated in Table 1 is the results of a feedback-loop between the two. The training-set has then been obtained by selecting tracks among a large music collection considered as prototypical of the chosen genres. By prototypical, we mean tracks representative of the exact genre and not cross-over between several genres. For the other tags (**mood, instrumentation**), 4000 tracks have been manually annotated by two individual professional annotators. Only labels for which the annotators agreed on the majority of the tracks are considered. For these labels, only tracks for which both annotators agreed have been selected for the training. This process lead to the five moods and three view-points on instrumentation indicated in Table 1. "**Live**" classifier has been trained on a dedicated training-set made of the concatenation of all possible audience noise derived from real recording. The **singing** segment classifier has been trained using the Jamendo corpus [22].

5. GRAPHICAL USER INTERFACE

The GUI is the central element that allows user to interact with the prototype and to test the proposed use-cases. Its design is crucial since a bad GUI can hinder a good technology or a good use-case. Its design must follow a close user-feedback loop (see part 6). The current GUI (see Figure 3) is organized in three main panels: the interactive-player (top), the current play-list (left), the various tag-clouds (right).

The **player** panel displays the classical editorial metadata (track-title, artist-name, album-title) and the cover. A large horizontal time-line displays the estimated structure of the tracks. In this, parts with similar content are indicated by rectangle with similar colors. The user can browse through parts by directly clicking on the corresponding colored rectangle. The time-line also indicates the segments used for the audio summary by highlighting the corresponding parts (independently of the color). Once selected (using the play-list panel), a track automatically starts playing in the player either in full-duration or in audio summary mode. This choice is based on user preferences. A search-by-text panel is placed on the top of the


INTERACTIVE PLAYER
Vidéos
Musique
Rechercher :
Dans : Tous les champs (titre, artiste,
rechercher
Longtemps, longtemps (tu m'aimes en passant) Calme - Pop/Rock - Batterie pop légère/rock
Charlélie Couture
Poemes Rock
00:13 / 00:31
écouter le résumé | écouter l'intégral
CURRENT PLAYLIST
TAG CLOUDS

Résultats (1383)

enregistrement dans une playlist

Mode de lecture : résumé | intégral automatique | manuel

Titre	Artiste	Album	Durée
Longtemps, longtemps (tu m'aimes en passant) *Calme - Pop/Rock - Batterie pop légère/rock*	Charlélie Couture	Poemes Rock	02:58
Mister K. *Romantique - Pop/Rock - Guitare acoustique*	AaRON	Artificial Animals Riding On Neverland	02:57
Le Tunnel d'Or *Triste - Rap - Batterie pop légère/rock*	AaRON	Artificial Animals Riding On Neverland	03:38
Last Night Throughts *Triste - Pop/Rock - Piano*	AaRON	Artificial Animals Riding On Neverland	02:54
Let Me Put My Love into You *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Back in Black	04:15
Skies on Fire *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Black Ice	03:34
Big Jack *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Black Ice	03:57
Anything Goes *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Black Ice	03:22
Smash n Grab *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Black Ice	04:06
Wheels *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Black Ice	03:28
Decibel *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Black Ice	03:33
Stormy May Day *Dynamique - Pop/Rock - Guitare électrique*	AC/DC	Black Ice	03:10

1/116 résultats suivants >

Genres: inconnu (3938) Pop/Rock (1383) Reggae (558) Electronique (471) Blues (277) Jazz (207) Rap (187) Classique (166) Métal/Punk (153) Soul/Funk (66) latin (38) Tous les genres

Humeurs: Dynamique (650) inconnu (490) Triste (179) Romantique (111) Calme (25) Joyeux (23) Toutes les humeurs

Instrumentations: Guitare électrique (1177) Batterie Pop Legère/Rock (1033) electronique (503) guitare acoustique (162) Batterie Jazz/Country/Soul (64) Batterie électronique (63) Batterie Hard Rock/Metal (61) piano (46) codes (orchestre) (39) acoustique (20) inconnu (15) cuivres (4) Toutes les instrumentations

Figure 3. GUI of the Music Interface of the prototype

interface. It allows either full-database search or search over restricted criteria (title, artist, album).

The **play-list** panel indicates the currently selected tracks which correspond either to – the results of a search-by-text, a search/filtering using the tag-clouds or a search-by-similarity, – or a previously stored play-list [6] . For each track, the estimated tags (genre, mood, instrumentation) are also indicated. The musical note icon next to each track allows performing search-by-similarity.

The **tag clouds** panel indicates the various viewpoints on the content: genre, mood, and instrumentation. The tags that are currently active in the filtering are indicated by highlighted colors. Next to each tag-name is indicated the number of corresponding items. It should be noted that the tags inside a cloud are not mutually exclusive.

6. USER TESTS

We define user experience as "the combination between the quality of the technology, the functionality based on it and the way to present it on Human Machine Interface". During the project, 1 or 2 user tests per year were performed, each corresponding to a new version of the prototype.

Many of the outputs of user-tests relate to the usability of the GUI: naming of the fields, their spatial organization, layout of the tag-clouds . . . This is of course essential; especially considering that the music search part is only one part of the whole search engine (which also includes TV and Web-Video search) and the presentation of the various search engines must be as much as possible homogeneous. User-tests are performed using two methods.

[6] The playlist, tempo and key/mode functionalities are not discussed here since their are currently subject to modifications of the GUI.

6.1 Qualitative tests

The first method consists in performing **qualitative** tests. Qualitative tests have three focuses: (1) to asses the usability of HMI (2) to asses users' judgment of functionalities (3) to imagine with users new use cases and maybe new functionalities based on the music technology. For this, users were asked to perform various scenario: "use the search engine to create a music play list of a specific mood", "to discover new music" . . . This is followed by interviews, which allows highlighting problem in the usability of the GUI, collecting judgments of functionalities (audio summary, genre, mood and similarity are found highly relevant while music structure displaying less relevant). This has also allowed highlighting missing functionalities. Displaying singing segments was one of those.

6.2 Quantitative tests: the case of audio summary

The second method consists in performing **quantitative** tests to compare several variations of a technology. An example of this are the "audio summary" user-tests.

For the creation of the summary, a set of user tests have been performed in order to select the best summary strategy. For this we have compared four different types of summary: –a 30s extract at the beginning of the file, –a random 30s extract, –the most representative 30s extract (denoted by 1x30), –a downbeat-synchronous concatenation of the three most representative 10s extracts (denoted by 3x10) [17]. 24 users had to listen to tracks of music they knew (7 tracks) and music they didn't know (6 tracks). Half of the songs were in their native language (French), the other half in English. They were then asked the questions - "which technique better summarized the track" (for music they knew), - "which technique is the most informative of its content" (for music they didn't know). In both cases, the 3x10 summary was judged better.

A quantitative evaluation has also been performed to compare the 1x30 and 3x10 summary. Over a 160-tracks database, we have measured the number of tracks for which each technique allowed to include the track title in the summary (the track title is considered here as the most memorable part of the track). The 3x10 summary achieved 95% correct location, while the 1x30 achieved 90%.

A user evaluation of the acoustical quality of the multi-parts (3x10) summary has also been performed. We have compared four configurations of the audio construction: –complete DSOLA –partial DSOLA (the loudness of the audio decreases during the transitions between parts to highlight them), –DSOLA with sound insertion (a prototypical sound is introduced at each transition to highlight them), –partial DSOLA with visual feedback. This experiment highlighted the fact that in some cases (especially Rap music), the complete DSOLA leads to an audio that sounds exactly like a real track. However, users prefer to feel a separation between the three 10-second parts to avoid having the feeling of listening to a new mix from a DJ. We also decided to add a visual presentation to increase the understanding of this summary functionality. This visual presentation consists in 3 highlighted segments of the complete music timeline corresponding to the three 10-second parts of the summary. The play cursor “jumps” from part to part. With these choices and modifications, user experience of the summary was improved.

7. INTEGRATION

The back-office of the prototype is based on a Service Oriented Architecture (SOA). This kind of architecture is flexible and particularly adapted for the integration of numerous and distant technologies. The main elements of this architecture are: −Metadata collectors, which collect metadata coming from content providers (TV Programs, INA archives, Web videos, music); −Technological modules, accessible as Web services (e.g. speech to text, named entities extraction, music analysis); −An XML transverse metadata base, which stores all metadata coming from collectors and technological modules; −An ESB (Enterprise Service Bus), which connects the metadata collectors, the technological modules and the metadata base; −A specific XML “pivot” format for all metadata manipulated by the ESB and the XML database. The search engine indexes are fed by the XML database through a metadata exporter. The search engine is directly connected to the application.

8. CONCLUSION

In this paper, we wanted to share our experience on integrating music-content indexing technologies, as developed in a research lab, into a third-part search and navigation engine. For this, we provided a panorama of the various elements of works implied and how they interact.

The lessons we learned from this experience is that this integration involves much more than good signal processing and machine learning technologies, which are of course essential. A side from the technical constraints (robustness, scalability), many of the works to be performed relate to make these technologies usable. This involves first proposing useful and understandable tags for users and creating the related annotated corpus to train the algorithms. This also involves tuning and modifying technologies: to favor precision over recall; or to provide reliability for all estimations (which is difficult for descriptions such as tempo or key). User tests allows to highlight new challenges, such as the need for a list containing only the similar items and not just a ranked-list from the most to the less similar items; or the fact that some innovative technologies may be found too specialized for users (music structure). We hope the information provided here would help the research community when trying to move from research applications to third-party applications.

Ackowledgements: J.J. Burred is now with Audionamix. This work was supported by the “Quaero” Program funded by Oseo French State agency for innovation.

9. REFERENCES

[1] F. Pachet, J. Aucouturier, A. La Burthe, A. Zils, and A. Beurive, “The cuidado music browser: an end-to-end electronic music distribution system,” *Multimedia Tools and Applications*, vol. 30, no. 3, pp. 331–349, 2006.

[2] M. Gasser and A. Flexer, “Fm4 soundpark: Audio-based music recommendation in everyday use,” in *Proc. of the 6th Sound and Music Computing Conference (SMC 2009), Porto, Portugal*, 2009.

[3] M. Goto and T. Goto, “Musicream: New music playback interface for streaming, sticking, sorting, and recalling musical pieces,” in *Proceedings of the 6th International Conference on Music Information Retrieval*, pp. 404–411, 2005.

[4] A. Lillie, *MusicBox: Navigating the space of your music*. PhD thesis, Massachusetts Institute of Technology, 2007.

[5] P. Knees, M. Schedl, T. Pohle, and G. Widmer, “Exploring music collections in virtual landscapes,” *Multimedia, IEEE*, vol. 14, no. 3, pp. 46–54, 2007.

[6] D. Reynolds, T. Quatieri, and R. Dunn, “Speaker verification using adapted gaussian mixture models,” *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[7] D. Tardieu, C. Charbuillet, F. Cornu, and G. Peeters, “Mirex-2011 single-label and multi-label classification tasks: Ircamclassification2011 submission,” in *MIREX Extended Abstract*, (Miami, USA), 2011.

[8] C. Charbuillet, D. Tardieu, and G. Peeters, “Gmm supervector for content based music similarity,” in *Proc. of DAFX*, (Paris, France), pp. 425–428, September 2011.

[9] C. Charbuillet, D. Tardieu, F. Cornu, and G. Peeters, “2011 ircam audio music similarity system 1,” in *MIREX Extended Abstract*, (Miami, USA), October 2011.

[10] C. Cao and M. Li, “Thinkit’s submissions for mirex2009 audio music classification and similarity tasks,” in *MIREX Extended Abstract*, (Kobe, Japan), 2009.

[11] F. Bimbot, L. Mathan, A. De Lima, and G. Chollet, “Standard and target driven ar-vector models for speech analysis and speaker recognition,” in *Proc. of IEEE ICASSP*, vol. 2, pp. 5–8, 1992.

[12] J.-J. Burred and G. Peeters, “An adaptive system for music classification and tagging,” in *Proc. of LSAS (Int. Workshop on Learning the Semantics of Audio Signals)*, (Graz, Austria), 2009.

[13] F. Fuhrmann and P. Herrera, “Quantifying the relevance of locally extracted information for musical instrument recognition from entire pieces of music,” in *Proc. of ISMIR*, (Miami, USA), 2011.

[14] G. Peeters, “Template-based estimation of time-varying tempo,” *EURASIP Journal on Applied Signal Processing*, vol. 2007, no. 1, pp. 158–158, 2007. doi:10.1155/2007/67215.

[15] G. Peeters, “Spectral and temporal periodicity representations of rhythm for the automatic classification of music audio signal,” *IEEE Trans. on Audio, Speech and Language Processing*, vol. 19, pp. 1242–1252, July 2011.

[16] G. Peeters, “Chroma-based estimation of musical key from audio-signal analysis,” in *Proc. of ISMIR*, (Victoria, Canada), pp. 115–120, 2006.

[17] G. Peeters, A. Laburthe, and X. Rodet, “Toward automatic music audio summary generation from signal analysis,” in *Proc. of ISMIR*, (Paris, France), pp. 94–100, 2002.

[18] D. Jiang, L. Lu, H.-J. Zhang, J.-H. Tao, and L.-H. Cai, “Music type classification by spectral contrast,” in *Proc. of IEEE ICME (International Conference on Multimedia and Expo)*, (Lausanne Switzerland), 2002.

[19] G. Peeters, “Sequence representation of music structure using higher-order similarity matrix and maximum-likelihood approach,” in *Proc. of ISMIR*, (Vienna, Austria), 2007.

[20] J. Foote, “Automatic audio segmentation using a measure of audio novelty,” in *Proc. of IEEE ICME (International Conference on Multimedia and Expo)*, (New York City, NY, USA), pp. 452–455, 2000.

[21] M. Cooper and J. Foote, “Automatic music summarization via similarity analysis,” in *Proc. of ISMIR*, (Paris, France), pp. 81–85, 2002.

[22] M. Ramona, G. Richard, and B. David, “Vocal detection in music with support vector machines,” in *Proc. of IEEE ICASSP*, pp. 1885–1888, 2008.

CHORD RECOGNITION USING DURATION-EXPLICIT HIDDEN MARKOV MODELS

Ruofeng Chen **Weibin Shen** **Ajay Srinivasamurthy**
Georgia Tech Center for Music Technology
{ruofengchen, weibin_shen, ajays}@gatech.edu

Parag Chordia
Smule Inc.
parag@smule.com

ABSTRACT

We present an audio chord recognition system based on a generalization of the Hidden Markov Model (HMM) in which the duration of chords is explicitly considered - a type of HMM referred to as a hidden semi-Markov model, or duration-explicit HMM (DHMM). We find that such a system recognizes chords at a level consistent with the state-of-the-art systems – 84.23% on Uspop dataset at the major/minor level. The duration distribution is estimated from chord duration histograms on the training data. It is found that the state-of-the-art recognition result can be improved upon by using several duration distributions, which are found automatically by clustering song-level duration histograms. The paper further describes experiments which shed light on the extent to which context information, in the sense of transition matrices, is useful for the audio chord recognition task. We present evidence that the context provides surprisingly little improvement in performance, compared to isolated frame-wise recognition with simple smoothing. We discuss possible reasons for this, such as the inherent entropy of chord sequences in our training database.

1. INTRODUCTION AND BACKGROUND

The problem of audio chord recognition has been explored for over a decade and thus there exists an established basic framework that is widely used. First, a chroma feature, or its variation is computed, followed by classifiers or sequential decoders to recognize the chord sequence. Certain enhancements to the basic feature computation and classification algorithms have been found to be useful.

The chromagram, a sequence of 12-dimensional vectors that attempts to represent the strength of each pitch class, is computed using a log-frequency spectrogram, estimated with a constant-Q transform (CQT). Several methods have been proposed to refine the basic features. Non-Negative Least Square estimation [1] and Harmonic Product Spectrum [2] reduce the power of non-octave overtones in the spectrum. Harmonic-Percussive Source Separation [3] decreases the power of percussive sounds, which typically do not contain chord information [4], [5]. The CRP chroma algorithm [6] attempts to produce a timbre-invariant feature and has been applied in [5] and [7]. Background Subtraction [1] removes the running mean of a given spectrum, which is based on the same principle as the CRP chroma algorithm in that they both conduct long-pass filtering on audio signal, damping timbral information in features. Loudness-based chroma performs A-weighting [5] on log-frequency spectrogram. Besides chroma, a 6-dimensional feature called "tonnetz", based on Neo-Riemannian theory, is also commonly used and has proven to be helpful [8], [9]. Finally, a machine-learned transformation matrix that converts log-frequency spectrogram to chromagram is shown to outperform an arbitrary transformation matrix in [10].

Structural information has also been utilized to help audio chord recognition. Many systems use beat-synchronous chromagrams that are computed over a beat or half-beat, rather than short frames [1], [9], [11], [12]. In [7], the authors smoothed the chromagram based on a regressive plot. In [11], the authors demonstrate that an attempt to find explicit repetition in a piece can improve performance.

In the domain of classifiers or decoders, many published works use Hidden Markov Models (HMMs). Recent papers have used Dynamic Bayesian Network (DBN) in conjunction with separate bass and treble chromas for recognition [1], [5]. In the past, two methods implementing key detection to assist chord recognition have been proposed. The first method builds a group of key-specific models [4], [9], while the other treats key information as one of the layers in its graphical model [1], [5]. In some cases, transition matrices were based on, or initialized, using principles from music theory rather than learned from the training set [12]. Apart from HMMs, a Pitman-Yor Language Model [13] has also been used to build a vocabulary-free model for chord recognition. Finally, another popular approach is the use of chroma templates of chords [14], [15].

In this paper, we present our approach which proposes a novel method to compute the chroma, and uses duration-explicit HMMs (DHMMs) for chord recognition. DHMMs are discussed in [16], but have rarely been used in MIR research. We also try to answer an important question: how much can transitional context knowledge (i.e. chord progressions) contribute to increasing the accuracy of the model?

This paper is organized as follows: Section 2 describes the chroma feature that we use, emphasizing on a novel way of computing chromagram; Section 3 presents the DHMM and its implementation; Section 4 evaluates our models and analyzes the contribution of duration constraints and transitional context knowledge; Section 5 presents the conclusions and sheds light on future research.

2. CHROMA FEATURE

Our chroma feature is based on the 60 dimensional log-frequency spectrogram computation proposed in [5], which uses perceptual loudness scaling of the spectrogram. We set our frame length to 512 ms with a hop size 64 ms. We only consider the energy within 5 octaves between 55 Hz and 1760 Hz. We propose a new method to compute the chromagram from the spectrogram.

2.1 Chroma Based On Linear Regression

Chroma is typically calculated by "folding" the spectrum into one octave and summing over frequencies corresponding to a quarter-tone around a given pitch-class. In [10], the authors show that a machine-learned transformation matrix outperforms this basic method. We developed a method with similar motivation. The ground truth chord label is converted into a 24 dimensional chroma template logical vector, where the first 12 dimensions represent whether one of the 12 notes exists in the bass label, and the last 12 dimensions represent whether one of the 12 notes exists in the chord label. For an example, a "C:maj/5" is converted to
$[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0]$.

The target vectors are the chroma templates and we fit a transformation matrix which converts the log-frequency spectrum to a chroma vector that is as close to the chroma template as possible. Similar to [10], we explored the use of neural networks, experimenting with various activation functions including sigmoid, logistic, linear and quadratic. We found that sigmoid and logistic functions compress the outputs, leading to additional confusion between 0 and 1. Linear and quadratic regressions return nearly the same results without compressing the output. Consequently, we chose linear regression to fit the transformation matrix. The regressed matrix shown in Figure 1 transforms a 60 dimensional log-frequency spectrum into a 24 dimensional chroma vector, which is a concatenation of the bass and treble chroma. It is worth noticing that the transformation matrix has weights for both base frequency and harmonics, leading to a certain degree of inhibition of harmonics in the final chroma. We have additionally tried some other proposed methods to inhibit harmonics (e.g. NNLS [1], HPS [2]), but none of them returned an improvement on the final results. Since the matrix relies on ground truth chord labels in the training set, the linear regression is performed every time we train the model.

2.2 Tonnetz Feature for Bass

The chromagram we have obtained through the aforementioned method is not full rank. This is because the information about the root note occurs in both bass and treble chroma templates. Given the common knowledge that bass chroma contains less chord related information than treble chroma, it might be more suitable for a lower-dimensional representation. So, we convert a bass chroma (i.e. the first 12 dimensions) into a 6-dimensional tonnetz feature, described in detail in [8]. We explored converting the treble chroma into tonnetz, but we did not observe any improvement on accuracy, which is consistent with [5]. Our feature vector is thus 18 dimensional, consisting of the treble chroma and the tonnetz features from bass chroma.

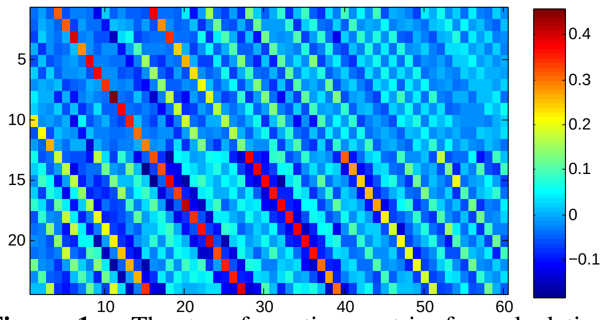


Figure 1. The transformation matrix for calculating chroma from CQTed-spectrum.

2.3 Beat Level Feature

In popular music, chord changes usually occur at beat and bar boundaries and thus, beat information can be useful in determining likely boundaries. We beat-tracked the audio using the dynamic programming approach of Ellis [17]. However, we extended the algorithms to allow for tempo variation within a song. A dynamic programming process was used to estimate the tempo before finding beat positions, similar to a method described by [18]. This resulted in a slight improvement in chord recognition accuracy.

We explored three approaches to calculate beat level features: (1) calculate chroma on the entire beat (large window for CQT); (2) calculate chroma on the frame level, then average over the frames in the beat; (3) calculate chroma on frame level, then take the median of each dimension within a beat. We found that approach (2) worked the best. In our experiments, we explore the use of both the frame level and beat level features in the HMMs.

3. DURATION-EXPLICIT HIDDEN MARKOV MODEL (DHMM)

In this section, we present a detailed discussion of the DHMM and its implementation, at the beat level. DHMMs estimate the chord sequence by simultaneously estimating chord labels and positions, which can be thought of as estimating on the chord level (See Figure 2). Initially, we applied DHMM hoping to reveal transitional context knowledge since at the frame and the beat level, self-transition is dominant. However, as we show in section 4.2, transitional context knowledge is not as important as we had hypothesized. Yet, modeling the duration of chords contributes to the majority of our improvement.

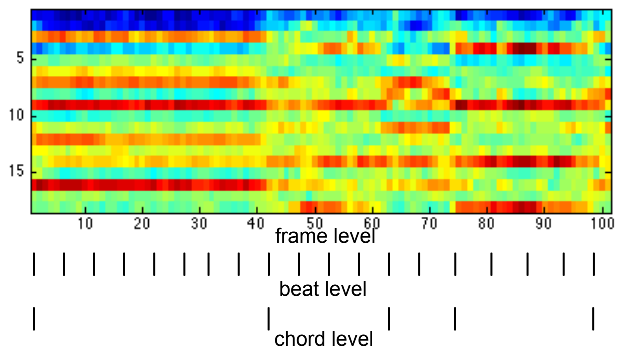


Figure 2. Frame level, beat level and chord level. Horizontal axis is frame level feature vector index; vertical axis is feature dimension index.

We adopt the notation used in [16]. To better understand the following expressions, readers are encouraged to briefly review III.A and III.B of [16]. In chord recognition, T is the number of beats in a song; observation sequence $O = \{O_1O_2 \dots O_T\}$ is the time sequence of feature vectors; state sequence $Q = \{q_1q_2 \dots q_T\}$ is the hidden chord label sequence; N is the number of chords being considered (i.e. the total number of states of the HMM); $S_1S_2 \dots S_N$ are possible states; $q_t \in \{S_1S_2 \dots S_N\}$; π is the vector of initial probabilities; $A = \{a_{ij}\}$ is the chord transition matrix, which denotes the probabilities of transitions from S_i to S_j; $B = \{b_i(O_t)\}$ is the emission matrix, which denotes the probabilities of emitting O_t from S_i; $p = \{p_i(d)\}$ is the duration distribution, which denotes the probabilities of S_i spanning d beats.

The model λ comprises of π, A, B and p. In our experiments, we found that π is unimportant so we set it to uniform distribution. A is trained by counting all the chord changes. A small value (0.05) is added to the diagonal of A before normalization, in order to bring the Viterbi algorithm back to sync when the actual duration has zero probability in p. A multivariate normal distribution is trained for each chord in order to calculate B. p is computed by counting the durations (i.e. number of beats) of each chord. The same duration distribution is used for all chords. However, the notation $p_i(d)$ is retained for better generalization. We limit the maximum duration D to 20 beats.

3.1 Viterbi Algorithm

Viterbi algorithm is a dynamic programming algorithm that finds the globally optimal state sequence $Q^* = \{q_1^*q_2^* \dots q_T^*\}$ explaining an observation sequence, given the model λ. We denote

$$\delta_t(i) = \max_i P(S_1S_2 \dots S_i \text{ ends at } t|\lambda)$$

A. Initialization ($t \le D$):

$$\delta^* = \max_{d=1}^{t-1} \max_{j=1, j\neq i}^{N} \delta_{t-d}(j)a_{ji}p_i(d) \prod_{s=t-d+1}^{t} b_i(O_s)$$

$$\delta_t(i) = \max\{\pi_i p_i(t) \prod_{s=1}^{t} b_i(O_s), \delta^*\}$$

B. Recursion ($D < t \le T$):

$$\delta_t(i) = \max_{d=1}^{D} \max_{j=1, j\neq i}^{N} \delta_{t-d}(j)a_{ji}p_i(d) \prod_{s=t-d+1}^{t} b_i(O_s)$$

In addition to $\delta_t(i)$, we need two other variables: $\psi_t(i)$ to track the last optimal state, and $\phi_t(i)$ to track optimal duration. If S_i ends at t, the optimal duration of S_i would be $\phi_t(i)$, and the optimal last state would be $\psi_t(i)$.

In initialization and recursion, we can get the index $\hat{j}$ and $\hat{d}$ that produce $\delta_t(i)$, then

$$\psi_t(i) = \hat{j}$$
$$\phi_t(i) = \hat{d}$$

If $\delta_t(i)$ equals $\pi_i p_i(t) \prod_{s=1}^{t} b_i(O_s)$ in initialization, then

$$\psi_t(i) = i$$
$$\phi_t(i) = t$$

C. Termination:

$$q_T^* = \arg \max_{1 \le i \le N} \delta_T(i)$$

D. Backtracking:

```
$d = \phi_T(q_T^*)$
$t = T$
**while** $t > d$ **do**
    $q_{t-d+1} \dots q_{t-1} = q_t$
    $q_{t-d} = \psi_t(q_t)$
    $t = t - d$
    $d = \phi_t(q_t)$
**end while**
$q_1 \dots q_{t-1} = q_t$
```

3.2 Probability of Observation Sequence

In some cases, it is necessary to know $P(O|\lambda)$, the probability that a model λ generates an observation sequence O. The computation of this probability is detailed in [16]. We applied the scaling method to prevent the probability from going below the machine precision.

The forward variable is defined as

$$\alpha_t(i) = P(O_1O_2 \dots O_t, S_i \text{ ends at } t|\lambda)$$

A. Initialization ($t \le D$):

$$\alpha^* = \sum_{d=1}^{t-1} \sum_{j=1, j\neq i}^{N} \alpha_{t-d}(j)a_{ji}p_i(d) \prod_{s=t-d+1}^{t} b_i(O_s)$$

$$\alpha_t(i) = \pi_i p_i(t) \prod_{s=1}^{t} b_i(O_s) + \alpha^*$$

$$c_t = \frac{1}{\sum_{s=1}^{t} \sum_{i=1}^{N} \alpha_s(i)}$$

$$\alpha_s(i) = \alpha_s(i)c_t, s = 1 \dots t$$

B. Recursion ($D < t \le T$):

$$\alpha_t(i) = \sum_{d=1}^{D} \sum_{j=1, j\neq i}^{N} \alpha_{t-d}(j)a_{ji}p_i(d) \prod_{s=t-d+1}^{t} b_i(O_s)$$

$$c_t = \frac{1}{\sum_{s=t-D+1}^{t} \sum_{i=1}^{N} \alpha_s(i)}$$

$$\alpha_s(i) = \alpha_s(i)c_t, s = t - D + 1 \dots t$$

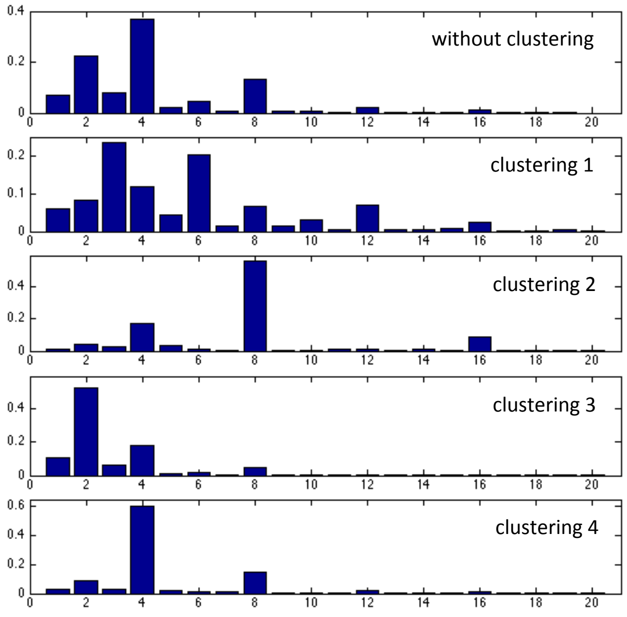


Figure 3. Top panel: Global duration distribution trained using the whole training set. Panel 2-5: Clustered duration distributions.

C. Termination:

$$\log P(O|\lambda) = -\sum_{t=1}^{T} \log c_t$$

Another precision problem (which does not exist in an ordinary HMM) is caused by $\prod b_i(O_s)$. Our solution is to divide all $b_i(O_s)$'s by the maximum value of B.

3.3 Time Signature Clustering

As will be shown in section 4.2, a global duration model has a limited contribution towards the accuracy improvement, because we train one single duration distribution using the whole training set. In fact, popular music is composed using a limited number of time signatures (e.g. 4/4, 6/8), and usually keeps its time signature unchanged for the whole length.

In other words, we train multiple duration distributions so that we have multiple models $\lambda_1, \lambda_2 \ldots \lambda_m$ (where only their p's are different), and we calculate $P(O|\lambda_1)$, $P(O|\lambda_2)$ $\ldots P(O|\lambda_m)$ and choose the model which maximizes likelihood, before running the Viterbi algorithm.

In order to train multiple duration distributions, we calculate a duration distribution for each song in the training set, and then cluster all the duration distributions into c categories using k-means algorithm (in our experiments, $c = 4$). We don't manually annotate time signatures because beat tracking algorithm is very likely to double or triple the tempo. Through clustering, we don't need to actually know the true time signature, and can account for potential errors caused by beat tracking. Figure 3 gives an example of clustered duration distributions compared to the global duration distribution.

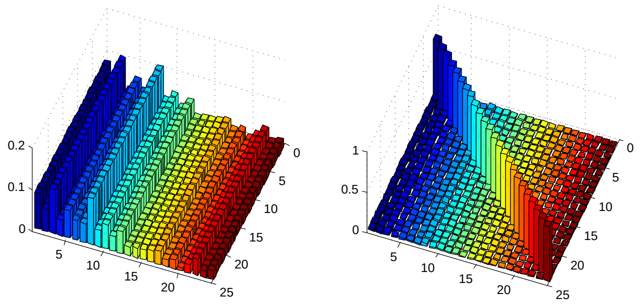


Figure 4. Left: Putting prior distribution to all rows of transition matrix. Right: Adding 3 to the diagonal of the matrix on the left and normalize each row.

4. EVALUATION

4.1 Experiments

We evaluate our models on two datasets: Beatles set by Harte [8] and Uspop set by Cho [7]. 44 songs in Uspop were excluded because we couldn't find audio of a length that matched the corresponding annotation. 12 songs were excluded from Beatles for reasons such as audio being off tune, or inconsistent time offsets of annotations. (See http://www.weibinshen.com/ISMIR2012Table.html for a full list of songs that were not used in this study).

We perform a 4-fold cross-validation experiment and report the average chord recognition accuracy on 24 major/minor chords. During training, all chord labels are remapped to 24 major/minor chords as in [5]. Each chord is trained with a single Gaussian distribution and corresponds to one state in HMM. During testing, each frame or beat is recognized as one of the major/minor chords. During evaluation, recognized labels on beat level are transformed to the frame level, and only frames with major/minor labels in the ground-truth annotations are counted for evaluation. The recognition accuracy metric is the frame-based recall rate - the number of frames that are recognized correctly, divided by the total number of frames, same as the evaluation metric used in MIREX evaluations.

In order to determine the contribution of chord progression information to the improved performance, we also baseline with Bayesian-type classifiers, where a chord prediction is determined by a MAP classifier independently at each frame or beat. We implement Bayesian-type classifiers by simply replacing every row of a transition matrix with the prior chord distribution, obtained by counting the unconditional occurrence of each chord (Figure 4).

4.2 Results

We compare the accuracy of different models in Table 1. In "Bayes", we train the transition matrix by applying the prior distribution to all the rows. In "Bayes+smooth", we apply a "majority" filter on the Bayesian classifier's output, in order to remove short-term deviations. In "Mod. Bayes", we add a relatively large number (arbitrarily, 3) to the diagonal elements of the "Bayes" transition matrix, and then normalize each row (see Figure 4-Right). In order to compare it with the state of the art, we also run the

Harmony Progression Analyser (HPA) proposed by Ni et. al [5] on the same datasets. It is a state of the art model using a three-layer HMM with key, bass note and chord estimation.

Level	Model	Uspop	Beatles
Frame level	Bayes	0.7518	0.7206
	Bayes+smooth	0.8285	0.8204
	HMM	0.8096	0.7966
Beat level	Bayes	0.7867	0.7733
	Mod. Bayes	0.8340	0.8331
	HMM	0.8365	0.8361
Duration model	Bayes	0.8371	0.8398
	DHMM	0.8377	0.8352
Time signature clustered duration model	Bayes	**0.8410**	**0.8413**
	DHMM	**0.8423**	**0.8419**
	HPA [5]	0.8401	0.8278

Table 1. A comparison of Accuracy

4.3 Analysis

We see that we achieve a performance comparable to the state of the art [5]. At the frame level, Bayesian classifier has a fairly low accuracy (75.18% on Uspop and 72.06% on Beatles). With smoothing (82.85% and 82.04%), it outperforms the basic frame-level HMM (80.96% and 79.66%). At the beat level, the Bayesian classifier attains a recognition rate of 78.67% for Uspop. However, when the self-transitions are emphasized in the "Modified Bayes" method, accuracy is on par with the beat-level HMM. Smoothing, as well as emphasizing self-transitions, essentially incorporate the knowledge that most chords last for more than a few frames or beats.

For DHMMs, duration information is decoupled from the transition matrix and results for the Bayesian classifier (83.71% and 83.98%) and the DHMM (83.77% and 83.52%) are similar. Using multiple duration models after clustering raises accuracy to 84.23% on Uspop, which is comparable the current state of the art.

The results suggest that the primary reason why HMMs are more effective than Bayesian classifier is that the strong self-transition probabilities emphasize continuity, rather than the knowledge of the chord progression represented in the transition matrix. In other words, when continuity is enforced by smoothing, or modeling durations separately, HMMs perform no better than a Bayesian classifier. Although there have been past works stating an improvement by using smoothing [22], we did not find any previous work discussing if the reason for HMMs outperforming Bayesian Classifiers is because of the smoothing effect of its transition matrix, or if the context information was really useful.

To further understand the contribution of chord progression knowledge we constructed an "oracle" condition in which the true transition matrix for a song was revealed (i.e. the transition matrix was computed using the ground truth labels for a particular song). This transition matrix was then used by the DHMM. The results are summarized in Table 2 and can be interpreted as an upper bound for chord recognition accuracy using a first-order DHMM. These results suggest that even in the case where the chord transitions are exactly known for a song, accuracy improves no more than 2%.

Level	Model	Uspop	Beatles
Duration model	Bayes	0.8735	0.8726
	DHMM	0.8863	0.8919

Table 2. Upper bound on performance

Why doesn't knowledge of the chord progression give greater improvements? In most cases, it seems the evidence provided in the local chromagram feature is quite strong, minimizing the need for top-down information. On the other hand, when the feature is noisy or ambiguous, it seems that the the prior imposed by the transition matrix is not very strong. In other words, chords progressions are less predictable than they seem.

We tested this hypothesis by estimating the entropy of chord progressions in the training set using a Variable Length Markov Model (VLMM) [19], [20]. In other words, given the context, we tested how sure we can be of the next symbol, on an average. A VLMM is an ensemble of Markov models which effectively captures variable length patterns in the chord sequence. A VLMM was used, as opposed to a first-order Markov model, because we wanted to ensure that long patterns were captured in addition to local chord transitions. Given the true symbol sequence till the current time frame, we obtain a predictive distribution over the chord labels for the next time frame, which is used to obtain the cross entropy of the test sequence. Using a VLMM, the minimum cross-entropy obtained was 3.74 on the Uspop dataset and 3.67 on the Beatles dataset (at a maximum VLMM order 2), in a leave-one-out cross validation experiment. It was found that the cross-entropy increased beyond order 2 in both datasets. An entropy value of 3.74 corresponds to a perplexity of 13.4, which can be interpreted as the average number of symbols the system was confused between. Thus, knowing the chord history does not, in general, narrow the possibilities greatly, and is unlikely to overcome a noisy or ambiguous feature vector.

5. CONCLUSIONS

In this paper, we presented an implementation of DHMMs and applied them to the chord recognition task. This model decouples the duration constraints from the transition matrix. We then build separate models for duration distributions that indicate different time signatures to improve the duration constraint in each model. Using this method, a comparable performance to the state of the art is demonstrated.

Though duration-explicit HMMs don't produce groundbreaking results, we believe that the proposed model may benefit other MIR tasks in the future, e.g. melody estimation and structural segmentation. Perhaps most importantly we show that state of the art results can be obtained using simple classifiers that do not use transition information. Further attempts to fully incorporate key and chord-progression knowledge (at least for popular songs of this type) using techniques such as high-order HMMs, are unlikely to yield significant improvements.

This material is based upon work supported by the National Science Foundation under Grant No. 0855758.

6. REFERENCES

[1] M. Mauch, S. Dixon: "Approximate Note Transcription For The Improved Identification Of Difficult Chords," In *Proceedings of the 10th International Conference on Music Information Retrieval*, Utrecht, Netherlands, 2010.

[2] K. Lee, "Automatic Chord Recognition Using Enhanced Pitch Class Profile," In *Proceedings of International Computer Music Conference*, New Orleans, USA, 2006.

[3] N. Ono, K. Miyamoto, H. Kameoka, S. Sagayama: "A Real-time Equalizer of Harmonic and Percussive Components in Music Signals," In *Proceedings of the 9th International Conference on Music Information Retrieval*, Philadelphia, USA, 2008.

[4] Y. Ueda, Y. Uchiyama, T. Nishimoto, N. Ono, S. Sagayama, "HMM-based Approach for Automatic Chord Detection Using Refined Acoustic Features," In *The 35th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Dallas, Texas, USA, 2010.

[5] Y. Ni, M. McVicar, R. Rodriguez and T. Bie: "An end-to-end machine learning system for harmonic analysis of music," In *IEEE Transactions on Audio, Speech and Language Processing*, In Press, 2012.

[6] M. Muller, S. Ewert, "Towards timbre-invariant audio features for harmony-based music," In *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 18(3):649–662, 2010.

[7] T. Cho and J. Bello, "A Feature Smoothing Method For Chord Recognition Using Recurrence Plots," In *Proceedings of the 12th International Conference on Music Information Retrieval*, Miami, Florida, USA, 2011.

[8] C. Harte, M. Sandler, M. Gasser: "Detecting Harmonic Change In Musical Audio," In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, Volume: C, Issue: 06,Santa Barbara, California, USA, 2006.

[9] K. Lee, M. Slaney, "A Unified System for Chord Transcription and Key Extraction Using Hidden Markov Models," In *Proceedings of the 8th International Conference on Music Information Retrieval*, Vienna, Austria, 2007.

[10] O. Izmirli, R. Dannenberg: "Understanding Features and Distance Functions for Music Sequence Alignment," In *Proceedings of the International Conference on Music Information Retrieval*, Utrecht, Netherlands, 2010.

[11] M. Mauch, K. Noland, S. Dixon: "Using musical structure to enhance automatic chord transcription," In *Proceedings of the 10th International Conference on Music Information Retrieval*, Kobe, Japan, 2009.

[12] J. Bello, J. Pickens, S. Pauws "A Robust Mid-Level Rerpresentation For Harmonic Content In Music Signals," In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, UK, 2005.

[13] K. Yoshi, M. Goto, "A Vocabulary-Free Infinity-gram Model For Nonparametric Bayesian Chord Progression Analysis," In *Proceedings of the 12th International Conference on Music Information Retrieval*, Miami, Florida, USA, 2011.

[14] L. Oudre, C. Fevotte, Y. Grenier, "Probabilistic Template-Based Chord Recognition," In *TELECOM ParisTech*, Paris, France, 2010

[15] T. Fujishima, "Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music," In *The 1999 International Computer Music Conference*, Beijing, China, 1999.

[16] L. Rabiner: "A Tutorial On Hidden Markov Models And Selected Applications In Speech Recognition," In *Proceeding of the IEEE*, Vol.77, No.2, February 1989.

[17] D. Ellis, "Beat Tracking by Dynamic Programming," In *Journal of New Music Research* Vol.36(1), 51-60, 2007.

[18] F. Wu, T. Lee, J. Jang, K. Chang, C. Lu, W. Wang, "A Two-Fold Dynamic Programming Approach to Beat Tracking for Audio Music With Time-Varying Tempo," In *Proceedings of the 12th International Conference on Music Information Retrieval*, Miami, Florida, USA, 2011.

[19] D. Conklin, I. H. Witten, "Multiple viewpoint systems for music prediction," In *Journal of New Music Research*, 24:51-73, 1995.

[20] M. Pearce, D. Conklin, and G. Wiggins, "Methods for Combining Statistical Models of Music," In *U. K. Wiil (Ed.), Computer Music Modelling and Retrieval (pp. 295-312).* Heidelberg: Springer.

[21] J. A. Burgoyne, J. Wild, I. Fujinaga, "An Expert Ground-Truth Set For Audio Chord Recognition And Music Analysis," In *Proceedings of the 12th International Conference on Music Information Retrieval*, Miami, Florida, USA, 2011.

[22] T. Cho, R. J. Weiss, J. P. Bello, "Exploring Common Variations in State of the Art Chord Recognition Systems," In *7th Sound and Music Computing Conference*, Barcelona, Spain, 2010.

FINDING REPEATING STANZAS IN FOLK SONGS

Ciril Bohak
University of Ljubljana
ciril.bohak@fri.uni-lj.si

Matija Marolt
University of Ljubljana
matija.marolt@fri.uni-lj.si

ABSTRACT

Folk songs are typically composed of repeating parts - stanzas. To find such parts in audio recordings of folk songs, segmentation methods can be used that split a recording into separate parts according to different criteria. Most audio segmentation methods were developed for popular and classical music, however these do not perform well on folk music recordings. This is mainly because folk song recordings contain a number of specific issues that are not considered by these methods, such as inaccurate singing of performers, variable tempo throughout the song and the presence of noise. In recent years several methods for segmentation of folk songs were developed. In this paper we present a novel method for segmentation of folk songs into repeating stanzas that does not rely on additional information about an individual stanza. The method consists of several steps. In the first step breathing (vocal) pauses are detected, which represent the candidate beginnings of individual stanzas. Next, a similarity measure is calculated between the first and all other candidate stanzas, which takes into account pitch changes between stanzas and tempo variations. To evaluate which candidate beginnings represent the actual boundaries between stanzas, a scoring function is defined based on the calculated similarities between stanzas. A peak picking method is used in combination with global thresholding for the final selection of stanza boundaries. The presented method was tested and evaluated on a collection of Slovenian folk songs from EthnoMuse archive.

1. INTRODUCTION

Folk music is receiving increased attention of the music information retrieval (MIR) community, as our awareness of the need for preserving cultural heritage and making it available to the general public grows. In order to process large quantities of folk song recordings gathered in ehtnomusicological archives, automated methods for analysis of these recordings need to be developed. Usually, such analysis starts with segmentation of recordings. Namely, folk songs are typically found within field recordings, which are integral documents of the process of folk music gathering and can, besides music, contain other kinds of content such as interviews with performers and background information. High-level segmentation of field recordings from EthnoMuse archive [13] into individual units can be done manually or by using automated methods [7].

For accurate analysis of individual songs, further segmentation into shorter parts is desirable. As folk songs typically consist of repetitions of melodically similar stanzas, segmentation boils down to finding the boundaries between repeating stanzas. This is quite different to segmentation of popular music, where songs typically consist of different parts, such as intro, verse, bridge and chorus.

While the structure of a popular song is usually more complex than the structure of a typical folk song, the segmentation of folk songs contains other challenges. While popular music is recorded by professional musicians in studios, folk songs are recorded in an everyday noisy environment (talking in the background, wind and other environmental noises, clapping . . .) and singers are mostly untrained and usually older people that may sing out of tune, forget parts of lyrics or melody, interrupt their performances, switch to speaking etc.

In this paper, a novel approach for segmentation of folk songs into stanzas is presented. The algorithm is based on finding the vocal pauses in a folk song recording, derive the likely candidates for stanza beginnings from the pauses, score these candidates and select the best matching ones to obtain the final segmentation.

2. RELATED WORK

Most segmentation algorithms were developed for segmentation of popular or classical music. A broad overview of implemented methods is given in [11], where authors present state-of-the-art approaches and results of segmentation and structure discovery in music recordings. Typically approaches are based on calculating different sound features, which are used for construction of similarity or self-similarity matrices. First use of such matrices in MIR is presented in [4]. By finding repeating parts in matrices the structure of a musical piece can be inferred. Approaches use different features for construction of self-similarity matrices, two of the more popular are Mel-frequency cepstral coefficients [2, 5, 12] and chroma vectors [1, 6].

In recent years, several approaches to segmentation of folk music were presented. In [10] authors present a method for robust segmentation and annotation of folk songs. The

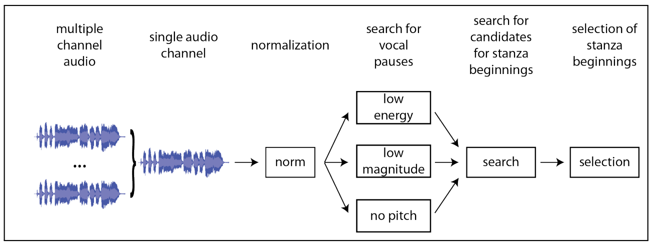


Figure 1. Outline of the proposed segmentation method.

presented approach uses chroma vectors in combination with a newly defined distance function for calculating the distance between individual stanzas and provided MIDI template. The method uses a MIDI representation of a single stanza to determine the length and expected pitch of each individual stanzas. Authors also present enhancements of the presented method, in which shifted chroma vectors are used to improve similarity between parts with shifted pitch.

A newer approach for detecting repetitive structures in music, presented in [9], introduces a novel fitness measure for defining the most representative part of song with the use of enhanced self-similarity matrix constructed from a variation of chroma-based audio features.

In [14], authors do not perform segmentation to search for repeating structure in folk songs, but are rather looking for their meaningful parts. Meaningful parts of folk songs are taken to be separated by breathing pauses, which are defined as parts of audio recording without detectable pitch.

EthnoMuse archive is a collection of audio field recordings, images, video recordings and metadata from Slovenian folklore. Archive contains more than 13.000 manuscripts, 1000 dance recordings, photos and more than 300 field recordings. Archive is not publicly accessible with exception of selected content. However parts of archive are published in book collections.

3. METHODOLOGY

Our segmentation method takes a folk song recording, as its input and outputs a set of boundaries, representing beginnings of individual stanzas in the recording. The method takes into account that performers are not professional singers, which may lead to pitch drifting over the duration of the piece, as well as considerable differences in tempo of individual stanzas.

The method consists of several steps: preprocessing, search for vocal pauses, search for possible beginnings of stanzas and selection of actual stanza beginnings. The general diagram of the suggested method is shown in Figure 1.

3.1 Preprocessing

The input audio signal is mixed from stereo to a single channel, the sample rate reduced to 11025 Hz and the amplitude normalized.

3.2 Detecting vocal pauses

Performers of folk songs are typically amateur singers which make characteristic breathing pauses, reflected in audio recordings as silence. These pauses are longer between stanzas, so they can be used to detect boundaries between stanzas. We use three approaches for detection of vocal pauses: short-term signal energy, amplitude envelope of the signal and the detected pitch. One can confirm such assumptions by listening to audio data from presented collection. This holds for solo and choir singing music, but does not hold for instrumental music.

3.2.1 Detecting vocal pauses according to signal energy

Vocal pauses in the audio signal are determined as parts of the signal where energy is below an experimentally determined threshold. Energy of the signal is computed on 200 ms long frames and the threshold is set to $\xi_1 = \frac{\overline{E}}{120}$, where $\overline{E}$ is the average energy of the signal. Consequent frames with energy values below the specified threshold are merged into one vocal pause. Vocal pauses shorter than $\xi_2 = 0.7$ times the average detected vocal pause length are ignored, to avoid the detection of short breathing pauses during singing. Parameter ξ_2 was also determined experimentally. Endings of detected vocal pauses, displayed red in Figure 2(a) (green are beginnings of vocal pauses), are later used as candidates for beginnings of stanzas.

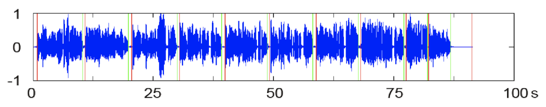
3.2.2 Detecting vocal pauses according to signal envelope

The amplitude envelope of a signal is obtained by filtering the full-wave rectified signal using 4th order Butterworth filter with a normalized cutoff frequency of 0.001. Vocal pauses are parts of the signal where the envelope falls below the threshold $\xi_3 = -60$dB, which was determined experimentally. Such parts of the signal are similarly as before merged into a single vocal pause, whereby we additionally merge all non-consequent parts that are less than $\xi_4 = 0.5$s apart, where the value ξ_4 was defined experimentally as well. As in the previous case, endings of detected vocal pauses are used as candidates for beginnings of stanzas and are displayed red in Figure 2(b) (green are the beginnings of vocal pauses).

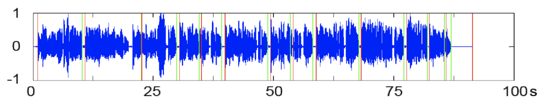
3.2.3 Detecting vocal pauses according to relative difference of pitch

For detecting parts of the signal without any detectable fundamental frequency we are using the approach presented in [14]. The input signal is first resampled to $f_s = 1024$Hz. The resampled signal is then used as input for the YIN algorithm [3] that calculates fundamental frequencies for each frame of the signal. Fundamental frequencies are smoothed with a low-pass filter. Parts of the signal that differ more than 20 semitones from the average signal frequency are selected as vocal pauses.

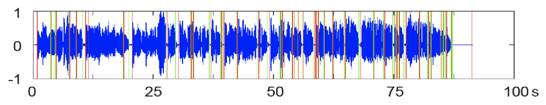
In our approach we are merging vocal pauses longer than an experimentally obtained value $\xi_5 = 4$ms, while shorter vocal pauses are ignored. We are also taking into account the minimal length of a vocal pause which is in our case $\xi_6 = 250$ms. Again, endings of vocal pauses are used as candidates for stanza beginnings. In Figure 2(c) the detected vocal pauses are shown (green are the beginnings and red are endings) for a sample recording.



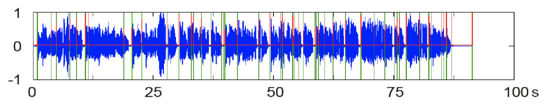
(a) Detected parts with low energy.



(b) Detected parts with low amplitude envelope.



(c) Parts with no detectable fundamental frequency.



(d) All detected stanza boundary candidates.

Figure 2. Comparison of methods for vocal pause detection. In images (a), (b) and (c) green are beginnings of vocal pauses and red are vocal pauses endings. In image (d) green are candidates for stanza boundaries and red is the value of fitness function for the candidates.

3.3 Finding candidates for stanza boundaries

In search for candidates for stanza boundaries we merge all sets of vocal pauses obtained with the previously described methods. An example of such a merged set is displayed in Figure 2(d) where the beginnings of vocal pauses are omitted and only their endings, which we consider as candidates for stanza boundaries, are shown in green. If candidates are present in several sets before merging, they are merged into a single candidate boundary.

We assume that the first candidate from the set represents the actual beginning of the first stanza. We then calculate the distance of the first $\xi_7 = 10$s of this first stanza to the 10s beginnings of all other stanza candidates determined by the candidates for stanza boundaries. The value of ξ_7 was chosen to represent approximately half of the average stanza length in our dataset. The calculation of distances between different stanza candidates takes the pitch drifting and tempo variations into consideration and is composed of four steps and is illustrated in Figure 3.

3.3.1 *Step 1*

We calculate 12 dimensional chromagrams, as defined in [8], for all stanza candidates using a window size of 50ms.

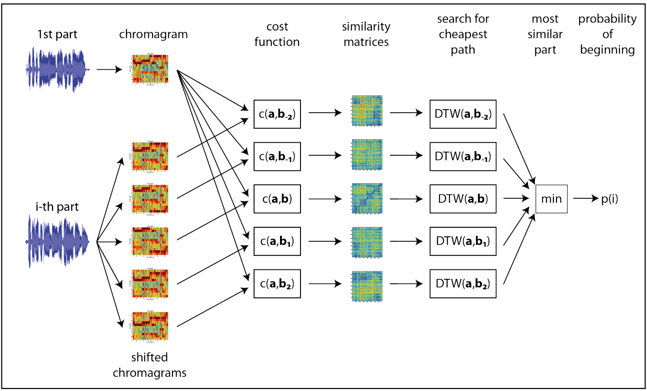


Figure 3. Outline of the algoritm for evaluating candidate stanza beginnings.

3.3.2 *Step 2*

We define a distance function between each pair of 12 dimensional chroma vectors as the root mean square (RMS) distance, which was also used for chorus detection in [6]:

$$c(\mathbf{a},\mathbf{b}) = \frac{\sqrt{(\sum_i (a_i - b_i)^2}}{\sqrt{12}}, \tag{1}$$

where c is the distance function between two chroma vectors $\mathbf{a}$ and $\mathbf{b}$, a_i and b_i are i-th elements of chroma vectors.

3.3.3 *Step 3*

The defined distance function is used by the Dynamic Time Warping (DTW) algorithm for calculation of the total distance between the selected stanzas as:

$$c_p(p_1, p_2) = \sum_{l=1}^{L} c(p_1(l), p_2(l)) \tag{2}$$

where p_1 and p_2 are candidate stanza beginnings. $p_1(l)$ and $p_2(l)$ are the corresponding chroma vectors (previously labeled as $\mathbf{a}$ and $\mathbf{b}$), the index l takes values from the first (1) to the last (L) chroma vector in the selected audio part. The DTW is used for calculating the total distance between two stanza candidates:

$$c_{min}(d_j) = DTW(d_0, d_j) = min\{c_p(d_0, d_j)\}, \tag{3}$$

where c_{min} is the minimal cost between parts d_0 and d_j. A similar approach that uses DTW for calculating the cost was used in [10].

3.3.4 *Step 4*

To account for pitch drifting during singing, we also calculate distances between stanza candidates with shifted chroma vectors. The chroma vectors are circularly shifted up to two semitones up and down to compensate for the out-of-tune singing. We then select the lowest DTW distance as:

$$\begin{aligned} dist_{min}(d_0) &= 0, \\ dist_{min}(d_j) &= \min_{d_j^f, f\in[-2,2]} \{c_{min}(d_0, d_j^f)\}, \end{aligned} \tag{4}$$

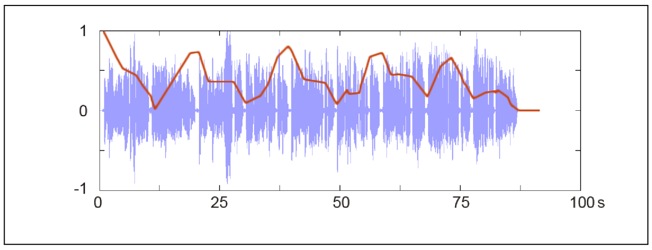


Figure 4. Fitness function for evaluating the candidate stanza beginnings.

where d_j^f represents a rotation of chroma vectors for the selected stanza candidate from two semitones downwards to two semitones upwards in steps of one semitone.

Finally, we define a fitness function for scoring the candidate stanza beginnings k_i as:

$$p(i) = \begin{cases} 0, & d_j \notin D \\ 1 - (\frac{dist_{min}(d_j)}{\max_j dist_{min}(d_j)})^2, & d_j \in D \end{cases} . \quad (5)$$

Figure 4 shows such a fitness function (Eq. 5) plotted on top of the audio signal. As the function is inversely proportional to the distance between the first stanza and a stanza candidate, higher fitness function values correspond to stanza boundaries which are more likely - stanzas are more similar and the candidate thus more likely represents a repetition of the original first stanza.

3.4 Selection of actual stanza beginnings

The selection of actual stanza beginnings is made with a simple peak picking algorithm in combination with a global threshold. In the defined fitness function, peaks represent the most likely stanza beginnings, so all peaks above a global threshold, corresponding to the average value of the fitness function, are picked as the actual boundaries between stanzas.

4. EXPERIMENTS AND RESULTS

The proposed method was tested on a set of folk songs from an ethnomusicological archive labeled as solo or choir singing, totalling 190 minutes in length and containing 135 units of solo or choir singing with an average duration of 100 seconds per unit. The average number of stanzas per unit was approximately 4, the average length of a stanza 18 seconds.

4.1 Evaluation of developed method

We performed an evaluation of vocal pause detection algorithms, as well as an evaluation of the whole segmentation method using the different detection algorithms.

The values of algorithm parameters $\xi_1 \ldots \xi_7$, used in vocal pause detection algorithms, were determined on a small set of recordings by evaluating a range of parameter values and choosing the ones for which the segmentation algorithm performed best. The algorithm itself is not very sensitive to changes in these parameters.

4.1.1 Evaluation of vocal pause detection

We evaluated individual approaches for detecting vocal pauses on the dataset containing 545 annotated vocal pauses. A detected vocal pause was considered as correctly detected, if it was within 2 seconds of the annotated vocal pause. Table 1 shows precision, recall and F-Measures of detection for individual methods and their combination. As shown in the table, combining all of the methods yields high recall and low precision. This is what we are aiming for at this first stage of the segmentation algorithm, because the second stage of the algorithm removes irrelevant vocal pauses and thus finding as many vocal pauses as possible is a priority.

Table 1. Comparison of vocal pause detection algorithms.

Algorithm	Precision	Recall	F-Measure
Energy	0,3336	0,8276	0,4755
Amplitude	0,5066	0,3729	0,4296
NoPitch	0,2793	0,5908	0,3793
Combination	0,0894	**0,9866**	0,1639

4.1.2 Evaluation of the method as a whole

Table 2 shows accuracy of the whole segmentation algorithm by using the four approaches to vocal pause detection described previously. One can see that the method, which uses the combination of all vocal pause detection algorithms, significantly outperforms the others. This result is expected, since individual vocal pause detection algorithms have lower recall, which means that we are already missing a number of annotated segment boundaries.

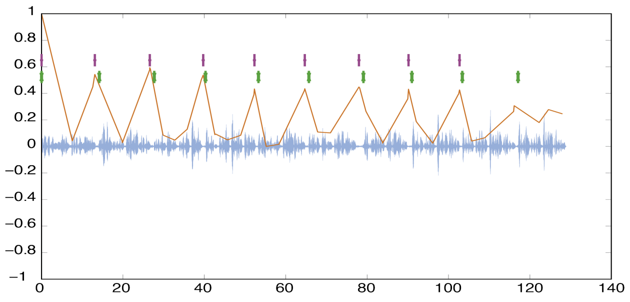
Table 2. Segmentation accuracy with different vocal pause detection algorithms.

Method	Precision	Recall	F-Measure
Energy	0,7592	0,4430	0,5595
Amplitude	0,6886	0,2574	0,3747
NoPitch	0,7447	0,3597	0,3597
Combination	0,6773	0,6435	**0,6600**

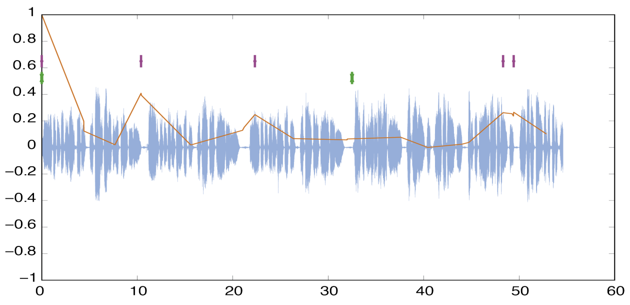
Our method performs well on songs that have strong vocal pauses, songs that consist of melodically similar stanzas and songs where the singing is approximately in tune. One can see an example of such vocal pause detection in Figure 5(a) where our method finds 9 out of 10 annotated vocal pauses. The 10th vocal pause is also clearly seen in the plotted fitness function, however the global thresholding prevents its detection.

The method fails on songs where the first stanza is incorrectly detected, because all stanzas are always compared to the first stanza. It also has difficulties in cases where stanzas are melodically very different, because comparison of chroma vectors relies on melodically similar stanzas. An example of such failure is shown in Figure 5(b).

In this example song consists of melodically significantly distinguished parts. First part of the song consists of three melodically similar parts that are also detected by our method, while same is true for the second part, its repeating parts are not similar to first part of the song.



(a) An example where our method performs well.



(b) An example where our method fails to find any annotated vocal pauses.

Figure 5. The figure shows an example where our method performs well (a) and an example where our method fails (b). The sound signal is plotted in blue, the fitness function is plotted in orange, true stanza beginnings are plotted with green stars (*) and the detected beginnings are plotted with pink plus signs (+).

4.2 Comparison with existing methods

We compared the proposed method with two existing folk song segmentation methods, results are shown in Table 3. The first method [14] only relies on detection of vocal pauses for segmentation and does not consider repetitions. Thus, the method covers most of the annotated vocal pauses (high recall), however it also detects many false positives, resulting in low precision.

We also compared our method with the method presented in [10] that uses symbolic transcription of a typical stanza as its input. Due to this additional prior knowledge, the method outperforms ours by a significant margin, as the approximate stanza melody and length are known to the algorithm. But, as this prior knowledge is not always available, the method cannot be used in all cases.

5. CONCLUSION AND FUTURE WORK

In this paper we presented a novel method for finding repeating stanzas in recordings of folk songs. The method relies on the detection of vocal pauses, which represent candidate stanza beginnings, which are then evaluated according to melodic similarity with the first stanza. The vocal pause detection algorithms and the method as a whole are separately evaluated on a dataset of folk song recordings. The method performs well, however several extensions are planned.

Our future work will include improvements with detection of the first stanza and processing of the fitness function, as well as adaptation of the method to instrumental tunes, where vocal pauses are not present. We also plan to integrate the developed method with our algorithm for high-level field recording segmentation, where it could be used to improve the accuracy of high-level segmentation. We will also try using more advanced preprocessing of the audio signal for environmental noise reduction. Other possibility is to try extracting pitch from the raw audio data and try finding repeating stanzas in symbolic domain.

Table 3. Comparison of our method to other approaches.

Method	Precision	Recall	F-Measure
Kranenburg	0,149	**0,930**	0,257
Mueller (Δ^{fluc})	0,865	0,748	**0,802**
Our method	0,442	0,646	**0,525**

6. ACKNOWLEDGEMENTS

Authors would like to thank all anonymous reviewers for thorough reviews with many suggestions on possible improvements and recommendations on how to improve this paper. This work could not been done without data provided by Institute of Ethnomusicology ar Research Centre of Slovenian Academy of Sciences and Arts. Research work was done as part of basic research and application project ETNOKATALOG: retrieval of semantic data from folk song and music, based on melodic and metro-rhythmic analysis funded by Slovenian research agency.

7. REFERENCES

[1] Mark A. Bartsch and Gregory H. Wakefield. To catch a chorus: Using chroma-based representations for audio thumbnailing. In *Applications of Signal Processing to Audio and Acoustics*, pages 15–19, New Platz, NY , USA, October 2001.

[2] Matthew L. Cooper and Jonathan Foote. Automatic music summarization via similarity analysis. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, pages 81–85, Paris, France, October 2002.

[3] Alain de Cheveigné and Hideki Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[4] Jonathan T. Foote. Visualizing music and audio using self-similarity. In *Proceedings of the 7th ACM international conference on Multimedia*, pages 77–80, 1999.

[5] Jonathan T. Foote and Matthew L. Cooper. Media segmentation using self-similarity decomposition. In *Proceedings of SPIE Storage and Retrieval for Multimedia Databases*, pages 167–175, 2003.

[6] Masataka Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Transactions on Audio, Speech & Language Processing*, 14(5):1783–94, 2006.

[7] Matija Marolt. Probabilistic segmentation and labeling of ethnomusicological field recordings. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 75–80, Kobe, Japan, October 2009.

[8] Meinard Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.

[9] Meinard Müller, Peter Grosche, and Nanzhu Jiang. A segment-based fitness measure for capturing repetitive structures of music recordings. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[10] 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 2009)*, pages 735–740, Kobe, Japan, October 2009.

[11] Jouni Paulus, Meinard Müller, and Anssi Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR 2010)*, pages 625–636, Utrecht, Netherlands, August 2010.

[12] Geoffroy Peeters. Toward automatic music audio summary generation from signal analysis. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, pages 94–100, Paris, France, October 2002.

[13] Gregor Strle and Matija Marolt. The ethnomuse digital library: conceptual representation and annotation of ethnomusicological materials. *International Journal on Digital Libraries*, pages 1–15, 2011. 10.1007/s00799-012-0086-z.

[14] Peter van Kranenburg and George Tzanetakis. A computational approach to the modeling and employment of cognitive units of folk song melodies using audio recordings. In *Proceedings of the 11th International Conference on Music Perception and Cognition*, pages 794–797, 2010.

DETECTING EPISODES WITH HARMONIC SEQUENCES FOR FUGUE ANALYSIS

Mathieu Giraud
LIFL, CNRS, Université Lille 1
INRIA Lille, France

Richard Groult
MIS, Université Picardie Jules Verne
Amiens, France

Florence Levé
MIS, Université Picardie Jules Verne
Amiens, France

ABSTRACT

Fugues alternate between instances of the *subject* and of other patterns, such as the *counter-subject*, and modulatory sections called *episodes*. The episodes play an important role in the overall design of a fugue: detecting them may help the analysis of the fugue, in complement to a subject and a counter-subject detection. We propose an algorithm to retrieve episodes in the fugues of the first book of Bach's *Well-Tempered Clavier*, starting from a symbolic score which is already track-separated. The algorithm does not use any information on subject or counter-subject occurrences, but tries to detect *partial harmonic sequences*, that is similar pitch contour in *at least two voices*. For this, it uses a substitution function considering "quantized partially overlapping intervals" [14] and a strict length matching for all notes, except for the first and the last one. On half of the tested fugues, the algorithm has correct or good results, enabling to sketch the design of the fugue.

1. INTRODUCTION

A fugue is a polyphonic piece built in imitation, where all voices appear successively sharing the same initial melodic material: a subject and, in most cases, a counter-subject. These patterns are repeated throughout the piece, either in their initial form or more often altered or transposed, building a complex harmonic texture. Many composers wrote fugues, or included fugal parts in larger pieces. The two books of Bach's *Well-Tempered Clavier* are a particularly consistent corpus, exploring the 24 major and minor tonalities in 48 preludes and fugues.

Fugues are often viewed as one of the pinnacle forms of Western music, and they are also used for pedagogical purposes, in music analysis as in composition. Their structure may look very formal, but still enable high levels of creativity. There are many treatises on fugues, or, more generally, on counterpoint, as for example [13] or [18]. Some of them include a complete musicological analysis of Bach's *Well-Tempered Clavier*, as the books of S. Bruhn [3, 4]. The fugues are thus perfect candidates for Music Information Retrieval (MIR) research, stimulating the development of algorithms on symbolic scores.

A first way to analyze fugues can be to use generic tools detecting repeating patterns or themes, possibly with approximate occurrences. Similarity between parts of a piece may be computed by the Mongeau-Sankoff algorithm [17] and its extensions, or by other methods for approximate string matching [6, 7, 19], allowing a given number of restricted mismatches. Several studies focus on finding *maximal repeating patterns*, limiting the search to *non-trivial* repeating patterns, that is discarding patterns that are a subpattern of a larger one with the same frequency [10,12,15]. Other studies try to find musically significant *themes*, with algorithms considering the number of occurrences [20], but also the melodic contour or other features [16].

More specifically, some MIR studies already focused on fugues. The study [21] builds a tool to decide if a piece is a fugue or not, with a method to find occurrences of thematic materials. The bachelor thesis [2] contains methods to analyze fugues, including voice separation. It proposes several heuristics to help the selection of repeating patterns inside the algorithms of [10] which maximizes the number of occurrences. The web site [9] also produces an analysis of fugues, extracting sequences of some repeating patterns, but without precise formal analysis. Finally, we proposed in [8] a method to detect subjects and counter-subjects, based on an analysis of repeating patterns with a diatonic substitution function and a specific length matching. This method finds the precise ends of these patterns in the majority of the fugues of the first book of Bach's *Well-Tempered Clavier*.

The subject and the counter-subject are focus of musical cognition, and will often be what is remembered from a fugue. However, the link between the different expositions of these patterns occurs in transitional sections called *episodes* that modulate from one tonality to another [13, 18]. The episodes have a part in the development of tension during the fugue.

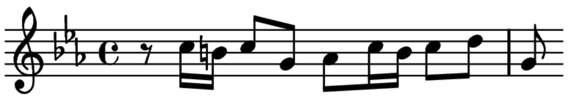
To our knowledge, no previous MIR study was devoted to analysis of episodes. Episodes can be detected by the absence of subjects and counter-subjects: A perfect detection of subjects and counter-subjects should yield a perfect episode detection. In this paper, we try to retrieve episodes without using any information on subject or counter-subject occurrences. We thus look for a *positive identification of* *episodes*. Starting from a symbolic score which is already track-separated, we propose an algorithm to retrieve episodes containing *partial harmonic sequences*, that is similar pitch contour in *at least two voices*. Harmonic sequences are commonly used to modulate, and are thus an essential feature of many episodes.

As in [8], the algorithm uses a strict length matching for all notes, except for the first and the last one. We tested several substitution functions to have a sensible and specific approximate matching. Our best results use the "quantized partially overlapping intervals" (QPI), introduced by Lemström and Laine in [14], that can be also seen as one case of the "General Pitch Interval Representation" defined by Cambouropoulos in [5].

The paper is organized as follows. Section 2 gives definitions and some background on fugues, Section 3 details our algorithm for episode detection through partial harmonic sequences, and Section 4 details the results on 21 fugues of the first book of Bach's *Well-Tempered Clavier*. These results were evaluated against a reference musicological book [4]. On half of the tested fugues, the algorithm has correct or good results, enabling to sketch the design of the fugue. The other cases are fugues where the episodes do not show enough harmonic sequences, or where the sequences are too short or too much altered.

2. PRELIMINARIES

A *note* x is described by a triplet (p, o, ℓ), where p is the pitch, o the onset, and ℓ the length. The pitches can describe diatonic (based on note names) or semitone information. We consider ordered *series of notes* $x_1 \ldots x_m$, that is $x_1 = (p_1, o_1, \ell_1), \ldots, x_m = (p_m, o_m, \ell_m)$, where $1 \leq o_1 \leq o_2 \leq \ldots \leq o_m$ (see Figure 1). The series is *monophonic* if there are never two notes sounding at the same onset, that is, for every i with $1 \leq i < m$, $o_i + \ell_i \leq o_{i+1}$. To be able to match transposed patterns, we consider relative pitches, also called *intervals*: the interval series is defined as ${}^{\Delta}x_2 \ldots {}^{\Delta}x_m$, where ${}^{\Delta}x_i = ({}^{\Delta}p_i, o_i, \ell_i)$ and ${}^{\Delta}p_i = p_i - p_{i-1}$.



pitch	72	71	72	67	68	72	71	72	74	67
interval		−1	1	−5	1	4	−1	1	2	−7
onset	2	3	4	6	8	10	11	12	14	16
length	1	1	2	2	2	1	1	2	2	2

Figure 1. A monophonic series of notes (start of Fugue #2, see Figure 4), represented by (p, o, ℓ) or $({}^{\Delta}p, o, \ell)$ triplets. In this example, onsets and lengths are counted in sixteenths, and pitches and intervals are counted in semitones through the MIDI standard.

Fugue. We now introduce some notions about fugue analysis. These concepts are illustrated by Fugue #2 of the first book of Bach's *Well-Tempered Clavier*. This fugue has a very regular construction.

A *fugue* is given by a set of *voices*, where each voice is a monophonic series of notes. In Bach's *Well-Tempered Clavier*, the fugues have between 2 and 5 voices, and Fugue #2 is made of 3 voices.

The fugue is built on a theme called *subject*. The first three *occurrences* of the subject in Fugue #2 are detailed in Figure 4: the subject is *exposed* at one voice (the alto), beginning on a C, until the second voice enters (the soprano, measure 3). The subject is then exposed at the second voice, but is now transposed to G. Meanwhile, the first voice continues with the first *counter-subject* that combines with the subject. Figure 3 shows a sketch of the entire fugue. The fugue alternates between other instances of the subject together with counter-subjects and development and modulatory sections called *episodes*.

Episodes and sequences. The episodes "effect a smooth transition from one key to the next [and] provide variety, as well as relief from a constant emphasis on the subject as a whole" [13]. They are often built on portions of material from the subjects of counter-subjects. S. Bruhn lists three roles for an episode in the design of the fugue: "It can link two subject statements by leading from one towards the next; it can be conclusive by resolving tension that was built up by the preceding subject statement; it can represent a different register, appearing basically independent of its surroundings and serving as a color contrast." [4].

The Figure 3 shows the two first episodes of Fugue #2. Note that the term "episode" can also be restrained to the ones after the exposition of all voices, the first episode being called *codetta* [18].

The episodes can include *cadential passages* for the release of tension. However, they are often composed with *harmonic sequences*, which are passages where a pattern is consecutively repeated starting on a different pitch. Figure 2 shows a simple harmonic sequence, outside of a fugue. Sequences can be diatonic (keeping the same key signature, possibly modulating to a neighbor tonality) or real (possibly gaining or losing some sharps or flats, often modulating to some other tonality).

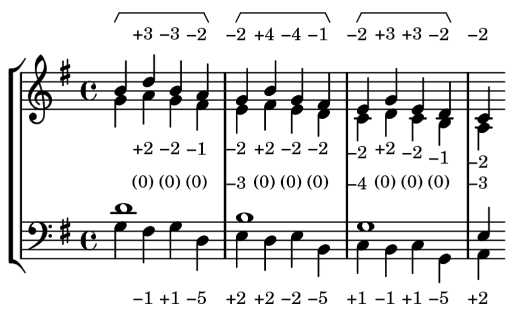
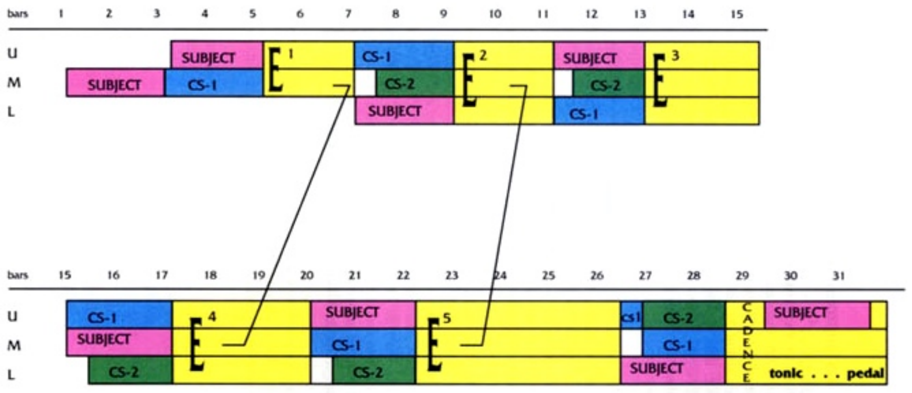


Figure 2. A simple diatonic sequence [1]. The values indicate the intervals from the preceding note of the same voice, in number of semitones. The occurrences #1 and #3 have exactly the same semitone intervals. The occurrence #2 is identical to these occurrences when one considers only diatonic information.



```
                                              1                                       2                                       3
          1   2   3   4   5   6   7   8   9   0   1   2   3   4   5   6   7   8   9   0   1   2   3   4   5   6   7   8   9   0   1
  S-truth         S-------       CS-------        S-------        CS------            S-------                  CSCS2---    S-------
  A-truth S-------CS------          CS2---          CS2---        S-------            CS------                    CS----
  T-truth                         S-------        CS------          CS2---              CS2---                  S-------

 detected                   ==--          ====----++      ==--++--        ==--++              ====----
sequences                                                 ====----              ==--++
```

Figure 3. Analysis of Fugue #2 in C minor in the first book of Bach's *Well-Tempered Clavier* (BWV 847). Top: ground truth (analysis by S. Bruhn, used with permission [4], [4, p. 80]). Bottom: the two lines named "detected sequences" show the output of the proposed algorithm, detecting partial harmonic sequences in 5 out of the 6 episodes and 68% of the concerned measures. The only false positive is the end of the second episode: at measure 11, it overlaps with the next occurrence of the subject (S) and counter-subject (CS).

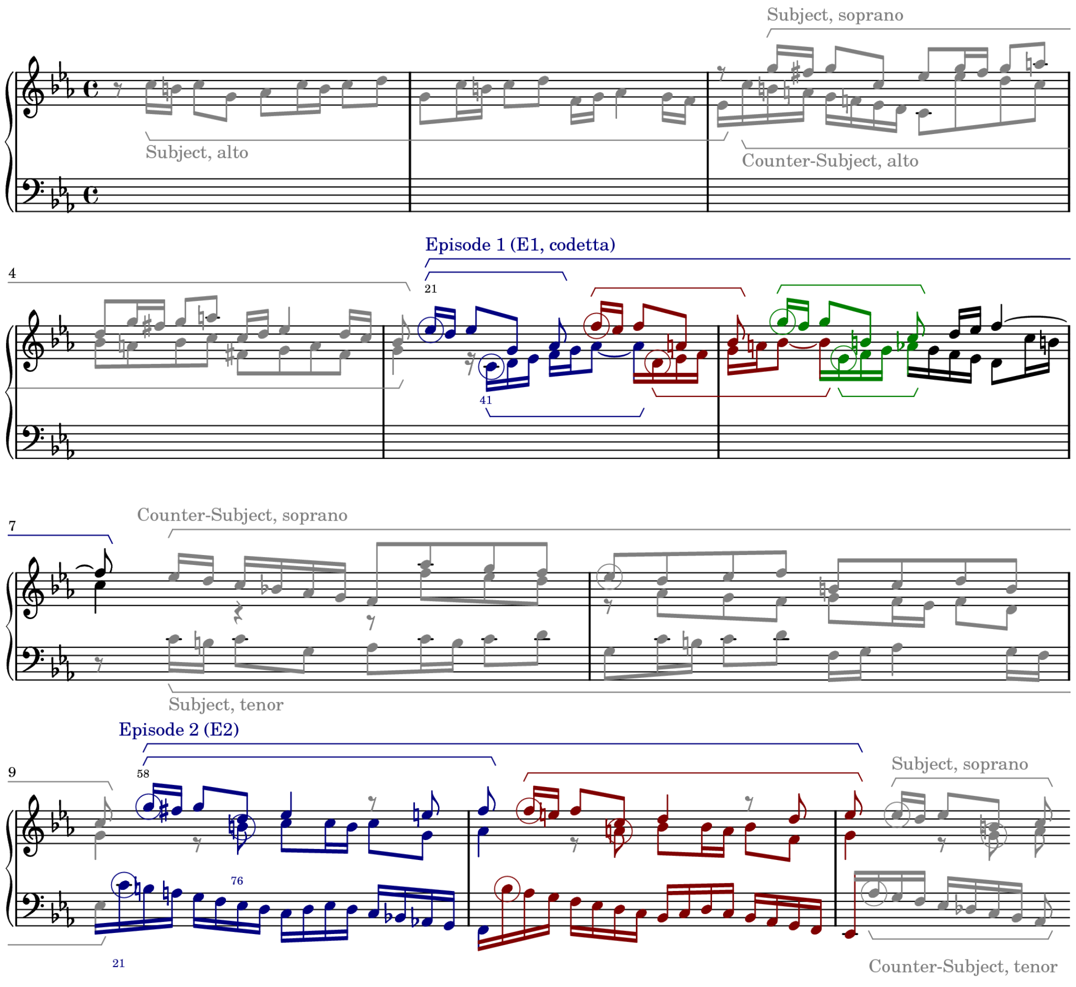


Figure 4. Start of Fugue #2 in C minor (BWV 847), showing the ground truth for the first two episodes. Non-episodic parts are grayed. The notes starting the initial patterns and the occurrences of the sequences are circled.

3. EPISODE DETECTION

We propose here to detect episodes containing partial harmonic matches in at least two voices. For this, we consider consecutively repeating patterns under a substitution function using a relaxed similarity for pitch intervals, and enforcing length equalities of all notes but the first one and the last one. These are very conservative settings, to have as few false positives as possible.

Consecutively repeating patterns. Formally, in a given voice x, we look for consecutively repeating patterns of p notes, starting at note x_e. The pattern $x_e...x_{e+p-1}$ has a candidate second occurrence $x_{e+p}...x_{e+2p-1}$, and, for larger episodes, we also check for a third ($x_{e+2p}...x_{e+3p-1}$) and fourth ($x_{e+3p}...x_{e+4p-1}$) occurrences.

The score $I(x, e, p, r)$ between the pattern $x_e...x_{e+p-1}$ and its candidate occurrence number r (r = 2, 3 or 4) is defined by the number of intervals matched between the pattern and its candidate occurrence:

$$\begin{aligned} I(x,e,p,r) &= \delta(\overset{\Delta}{}x_{e+1}, \overset{\Delta}{}x_{e+(r-1)p+1}) \\ &+ \delta(\overset{\Delta}{}x_{e+2}, \overset{\Delta}{}x_{e+(r-1)p+2}) \\ &+ \cdots \\ &+ \delta_f(\overset{\Delta}{}x_{e+p}, \overset{\Delta}{}x_{e+rp}) \end{aligned}$$

As in [8], we propose to use a strict length matching for all notes, except for the first and the last one – the length of these notes, at the extremities of the pattern, being more frequently altered. The substitution function δ checks thus pitch intervals and lengths, whereas the substitution function, δ_f, for the last note, only considers pitch intervals:

$$\delta((\overset{\Delta}{}p, o, \ell), (\overset{\Delta}{}p', o', \ell')) = \begin{cases} +1 & \text{if} \quad \overset{\Delta}{}p \approx \overset{\Delta}{}p' \quad \text{and} \quad \ell = \ell' \\ 0 & \text{if} \quad \overset{\Delta}{}p \not\approx \overset{\Delta}{}p' \quad \text{and} \quad \ell = \ell' \\ -\infty & \text{otherwise } (\ell \neq \ell') \end{cases}$$

$$\delta_f((\overset{\Delta}{}p, o, \ell), (\overset{\Delta}{}p', o', \ell')) = \begin{cases} +1 & \text{if} \quad \overset{\Delta}{}p \approx \overset{\Delta}{}p' \\ 0 & \text{otherwise } (\overset{\Delta}{}p \not\approx \overset{\Delta}{}p') \end{cases}$$

The actual comparison of lengths ($\ell = \ell'$) also checks the equality of the rests that may be immediately before the compared notes. The length of the first note of the pattern (x_e against $x_{e+(r-1)p}$) is never checked, as the score actually compares the series of intervals $\overset{\Delta}{}x_{e+1} \ldots \overset{\Delta}{}x_{e+p}$ against $\overset{\Delta}{}x_{e+(r-1)p+1} \ldots \overset{\Delta}{}x_{e+rp}$.

The relation $\approx$ is a similarity relation on pitch intervals. We use here the "quantized partially overlapping intervals" (QPI) [14], that defines *short* intervals (from one to three semitones), *medium* intervals (from three to seven semitones), and *large* intervals (starting from six semitones).

These classes can be considered for upward or downwards intervals, giving, with the unison intervals, a total of 7 different interval classes. Two pitch intervals $\overset{\Delta}{}p$ and $\overset{\Delta}{}p'$ will be considered as similar if there exists one class containing both of them.

There is an *exact occurrence* of the consecutively repeating pattern if $I(x, e, p, r) = p-1$. For example, on the sequence depicted on Figure 2, for any of the four voices x and for $r \in [2, 3]$, we have $I(x, 1, 4, r) = 3$, since intervals are perfectly similar under the QPI similarity relation. An *approximate occurrence* can be detected if $I(x, e, p, r)$ is at least equal to a given threshold $\tau(p)$.

Here the score $I(x, e, p, r)$ only considers substitution operations, and can be computed in time $O(p)$. The score can be extended to consider other edit operations, with computation through dynamic programming.

Episode detection through partial sequences. On the beginning of the Fugue #2, the consecutively repeating patterns are as follows:

- the second episode fits perfectly into an sequence: $I(\text{soprano}, 58, 7, 2) = 6$, $I(\text{alto}, 76, 7, 2) = 6$ and $I(\text{tenor}, 21, 16, 2) = 15$.
- the first episode has two complete occurrences, as $I(\text{soprano}, 21, 5, 2) = 4$ and $I(\text{alto}, 41, 6, 2) = 5$. There is no complete third occurrence, as the lengths do not match for one voice: $I(\text{alto}, 41, 6, 3) = -\infty$.

The complete algorithm computes $I(x, e, p, r)$ for every voice x, every note x_e starting right after a quarter beat, several periods (1 quarter, and 1/2, 1 and 2 measures) and for $r \in \{2, 3, 4\}$ occurrences. The algorithm reports an episode every time that at least two different voices contain a consecutively repeating pattern after the same onset (with $\tau(p) = 0.5 \times p$). Overlapping episodes with the same period are merged into an unique episode. The result on the Fugue #2 is depicted at the bottom of Figure 3.

For testing purposes, we used a naive implementation running in $O(n^2)$ worst time, where n is the total number of notes in the fugue. Even if similarities between occurrences in a sequence could be computed with tools in existing frameworks (such as the `simil` tool in the Humdrum toolkit [11, 19]), we coded our own implementation to be able to handle some specificities (specific length matching, partial detection in two voices).

4. RESULTS AND DISCUSSION

Results can be asserted in two different ways:

- We can count the sequences that are located completely or partially in episodes of the ground truth;
- More precisely, we can look at the total length of detected sequences, and compare it to the total length of all ground truth episodes, computing a ratio called *length sensibility*. This sensibility can be seen as a *coverage of episodes by harmonic sequences*: it will not reach 100%, as some episodes do not have sequences, and as the sequences may not be spanning all the episodes. We also compute a *length specificity* as the ratio between the lengths of true positive measures and of detected measures.

The result on Fugue #2 is shown at the bottom of Figure 3. Here 5 episodes out of 6 are detected with partial harmonic sequences. This is musically relevant, since the last episode (measures 29-31) is a cadential end, with a last exposition on the subject on a bass pedal. The ground truth has 14 1/2 measures of episodes. The algorithm outputs 10 measures (length sensibility of 68%) and falsely marks one half measure (2 quarters) as an episode (length specificity of 96%).

The false negatives are: 1 measure at the codetta (due to the shift between the two voices, only 2 occurrences are detected), 2 measures and a half at measure 24 (including a change of voices, see below), and all the 3 measures of the last episode (discussed above). The only false positive is the end of the second episode, which is extended 2 quarters below the next subject occurrence at measure 11, the soprano and the bass voices continuing the sequence (see Figure 4, last measure).

The complete test contains 21 fugues of the first book of Bach's *Well-Tempered Clavier* (fugues #1, #4 and #9 not showing significant episodic material). We started from `.krn` Humdrum files [11], available for academic purposes at `http://kern.humdrum.org/`. The output of the algorithm on all these 21 fugues is available at `http://www.lifl.fr/~giraud/fugues`. We checked all detected episodes, and Table 1 summarizes the results. On the 1098 measures of this test set, the algorithm labels about 20% of the measures as episodes, and finally identifies 43% of all episodes. A subjective quality assessment on the predictions, looking on the detailed output of each run, gives a *good* mark on 6 fugues, and a *correct* mark for 5 out of the 21 fugues.

False positives. There are very few false positives: less than 5% of the partial harmonic sequences overlap with subject and counter-subject occurrences. As for the measure 11 in Fugue #2, this is often because the texture of the episode fades into the next section.

False negatives. The length sensibility, that is the coverage of the episodes (in the ground truth) by the prediction of harmonic sequences is, in average, only 36%. These false negatives can be explained by several facts:

- As mentioned above, the sequences often not cover all the episodes. Moreover, there are some episodes with no harmonic sequence: It is often the case for the last episode, which is thus almost always missed by the proposed method;
- There are some episodes with *changes of voices* (Figure 5), in which the consecutive occurrences of a pattern are not in a same voice;
- Finally, the algorithm fails to detect some partial harmonic sequences that are too much altered to be recognized with the current threshold, or too short to be discovered.

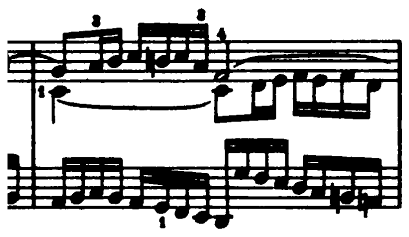


Figure 5. Partial sequence with a change of voices: the pattern is heard at the soprano, then, transposed, at the alto (measure 24 of Fugue #2).

5. CONCLUSIONS

We proposed an algorithm retrieving some episodes in the first book of Bach's *Well-Tempered Clavier*, starting from a symbolic score which is already track-separated. To our knowledge, this is the first MIR study on episodes in fugues. The algorithm, relying only on partial harmonic sequences detection, gives very few false positives, and already gives good results on some fugues. Enabling voice changes inside harmonic sequences should naturally detect more episodes, but may produce more false positives.

Many improvements are possible to have a better analysis of episodes. Detection of other positive features of the episodes (such as cadential passages) or, most of all, of some negative features (subject and counter-subject occurrences) could probably lead to a complete fugue analysis pipeline with better results.

The algorithm could also be tested on other corpus of fugues. As an example, the web page `http://www.lifl.fr/~giraud/fugues` shows the output of the proposed algorithm in the fugue of Mozart's *Adagio and Fugue in C minor*, K 546. Finally, partial or full harmonic sequence detection could be used to help the analysis of other genres.

Acknowledgements. The authors would like to thank the anonymous reviewers for their valuable comments.

6. REFERENCES

[1] Marcel Bitsch. *Précis d'harmonie tonale*. Alphonse Leduc, 1957.

[2] Lisa Browles. Creating a tool to analyse contrapuntal music. Bachelor Dissertation, Univ. of Bristol, 2005.

[3] Siglind Bruhn. *J. S. Bach's Well-Tempered Clavier. In-depth Analysis and Interpretation*. 1993. ISBN 962-580-017-4, 962-580-018-2, 962-580-019-0, 962-580-020-4. Available online at `http://www-personal.umich.edu/~siglind/text.htm`.

[4] Siglind Bruhn. *J. S. Bachs Wohltemperiertes Klavier, Analyse und Gestaltung*. Edition Gorz, 2006. ISBN 3-938095-05-9.

[5] Emilios Cambouropoulos. A general pitch interval representation: Theory and applications. *Journal of New Music Research*, 25(3):231–251, 1996.

					ground truth		found by the proposed algorithm					
#	BWV	tonality	voices	ℓ	nb	ℓ	nb	ℓ(TP)	ℓ(FP)	sens	spec	quality
2	847	C minor	3	31m	6	14m2q	5	10m	2q	68%	96%	+
3	848	C# major	3	55m	7	31m2q	4	17m	1m	54%	94%	+
5	850	D major	4	27m	5	16m+1q	4	7m	–	41%	100%	=
6	851	D minor	3	44m	8	12m	3	7m	1m+1q	57%	84%	=
7	852	Eb major	3	37m	9	22m+2q	7	16m+1q	–	72%	100%	+
8	853	D# minor	3	87m	10	22m+2q	1	3m+1q	2q	15%	100%	–
10	855	E minor	2	42m	4	25m	4	18m+2q	1m	75%	95%	+
11	856	F major	3	72m	6	31m+1q	4	10m	1m	31%	87%	=
12	857	F minor	4	58m	8	26m	5	12m+3q	7m	48%	65%	–
13	858	F# major	3	35m	6	17m+2q	4	7m+1q	2m+2q	42%	75%	=
14	859	F# minor	4	40m	5	10m	4	5m+3q	3m	56%	66%	=
15	860	G major	3	86m	9	36m	6	11m	15m+1q	30%	42%	–
16	861	G minor	4	34m	6	12m+1q	2	3m+1q	1m+2q	27%	71%	–
17	862	Ab major	4	35m	7	24m+2q	4	5m+1q	1q	21%	100%	–
18	863	G# minor	4	41m	6	17m	4	9m+3q	1m	57%	92%	+
19	864	A major	3	54m	7	27m	1	2m+3q	–	10%	100%	–
20	865	A minor	4	87m	16	19m	3	1m+2q	10m+2q	9%	14%	–
21	866	Bb major	3	48m	3	14m+1q	3	12m	1q	83%	96%	+
22	867	Bb minor	5	75m	6	34m+2q	1	2m+1q	2m	6%	60%	–
23	868	B major	4	34m	7	10m+1q	0		–	0%		–
24	869	B minor	4	76m	12	38m+1q	3	7m+2q	3m	20%	76%	–
				1098m	153	457m+20q	72	164m+24q	48m+14q			+ : 6
												= : 5
												– : 10

Table 1. Detection of episodes in 21 fugues of the first book of Bach's *Well-Tempered Clavier*. The ground truth is the analysis of [4]. All lengths (ℓ) are given in number of measures (m) and quarters (q). The sensibility (sens.) and specificity (spec.) are computed on the lengths (see Section 4). The last column, "quality", is a subjective assessment on the output of the algorithm: find almost all episodes with $\geq$ 50% length sensibility, almost no false positives with $\geq$ 90% length specificity (*good*, +), find most of the episodes, few false positives (*correct*, =), miss many episodes or produces many false positives (*bad*, –).

[6] Raphaël Clifford and Costas S. Iliopoulos. Approximate string matching for music analysis. *Soft. Comput.*, 8(9):597–603, 2004.

[7] T. Crawford, C. Iliopoulos, and R. Raman. String matching techniques for musical similarity and melodic recognition. *Computing in Musicology*, 11:71–100, 1998.

[8] Mathieu Giraud, Richard Groult, and Florence Levé. Subject and counter-subject detection for fugue analysis. In *Computer Music Modeling and Retrieval (CMMR 2012)*, 2012.

[9] J. Hakenberg. The Pirate Fugues. `http://www.hakenberg.de/music/music.htm`.

[10] J. L. Hsu, C. C. Liu, and A. Chen. Efficient repeating pattern finding in music databases. In *Int. Conference on Information and Knowledge Management (CIKM 1998)*, 1998.

[11] David Huron. Music information processing using the humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):11–26, 2002.

[12] Ioannis Karydis, Alexandros Nanopoulos, and Yannis Manolopoulos. Finding maximum-length repeating patterns in music databases. *Multimedia Tools Appl.*, 32:49–71, 2007.

[13] Kent Kennan. *Counterpoint*. Prentice Hall, 4th ed., 1999.

[14] Kjell Lemström and Pauli Laine. Musical information retrieval using musical parameters. In *Int. Computer Music Conference (ICMC '98)*, pages 341–348, 1998.

[15] Chih-Chin Liu, Jia-Lien Hsu, and Arbee L.P. Chen. Efficient theme and non-trivial repeating pattern discovering in music databases. In *Int. Conf. on Data Engineering (ICDE 99)*, pages 14–21, 1999.

[16] Colin Meek and William P Birmingham. Automatic thematic extractor. *Journal of Intelligent Information Systems*, 21(1):9–33, 2003.

[17] Marcel Mongeau and David Sankoff. Comparaison of musical sequences. *Computer and the Humanities*, 24:161–175, 1990.

[18] Hugo Norden. *Foundation Studies in Fugue*. Crescendo Publishing, 1977.

[19] Keith S. Orpen and David Huron. Measurement of similarity in music: A quantitative approach for non-parametric representations. *Computers in Music Research*, 4:1–44, 1992.

[20] Lloyd Smith and Richard Medina. Discovering themes by exact pattern matching. In *Int. Symp. for Music Information Retrieval (ISMIR 2001)*, pages 31–32, 2001.

[21] Pei-Hsuan Weng and Arbee L. P. Chen. Automatic musical form analysis. In *Int. Conference on Digital Archive Technologies (ICDAT 2005)*, 2005.

INFERRING CHORD SEQUENCE MEANINGS VIA LYRICS: PROCESS AND EVALUATION

Tom O'Hara[1], **Nico Schüler**[2], **Yijuan Lu**[1], and **Dan E. Tamir**[1]
[1]Dept. of Computer Science and [2]School of Music, Texas State University, San Marcos, TX 78666
{tomohara, nico.schuler, yl12, dan.tamir}@txstate.edu

ABSTRACT

We improve upon our simple approach for learning the "associational meaning" of chord sequences from lyrics based on contingency statistics induced over a set of lyrics with chord annotations. Specifically, we refine this process by using word alignment tools developed for statistical machine translation, and we also use a much larger set of chord annotations. In addition, objective evaluation measures are included. Thus, this work validates a novel application of lexicon induction techniques over parallel corpora to a domain outside of natural language learning. To confirm the associations commonly attributed to major versus minor chords (i.e., happy and sad, respectively), we compare the inferred word associations against synonyms reflecting this dichotomy. To evaluate meanings associated with chord sequences, we check how often tagged chords occur in songs labeled with the same overall meaning.

1. INTRODUCTION

Chords are the foundation of western music, providing the harmony for music and also influencing the melody (given close relation to musical keys). Chords are not simply three or more notes simultaneously played but also involve precise relationships among the notes. For example, the notes in a major chord consist of the root (lowest frequency), a note a third above the root (i.e., two whole steps), and a note a fifth above the root (e.g., three whole steps and a half). An example would be the $CMaj$ chord, which consists of the notes C, E and G. Likewise, chord sequences generally have precise definitions. For example, the popular *12-Bar Blues Progression* commonly uses the following scheme: ⟨I, I, I, I, IV, IV, I, I, V, IV, I, I⟩, where Roman numerals refer to chord intervals [16]. In the key of C, this would be as follows: ⟨C, C, C, C, F, F, C, C, G, F, C, C⟩. Given such precise relationships to musical intervals, meanings typically attached to chord sequences are unlikely to be completely arbitrary. This paper demonstrates how to learn the *associational meaning* [8] of chord sequences (e.g., in terms of word associations).

Parallel text corpora were developed primarily to serve multilingual populations but have proved invaluable for inducing lexicons for machine translation [2, 5]. Similarly, a type of resource intended for musicians can be exploited to associate meaning with music. Guitarists learning new songs often rely upon tablature notation ("tabs") provided by others to show the finger placement for a song measure by measure. Tabs often include lyrics, enabling note sequences to be associated with words. They also might indicate chords as an aid to learning the sequence (as is often done in scores for folk songs). In some cases, the chord annotations for lyrics are sufficient for playing certain songs (e.g., accompaniment by guitar strumming).

There are several web sites with large collections of tabs and chord annotations for songs (e.g., about 250,000 via www.chordie.com). These build upon earlier Usenet-based guitar forums (e.g., alt.guitar.tab). Such repositories provide a practical means to implement unsupervised learning of the meaning of chord sequences from lyrics. As these resources are willingly maintained by thousands of guitarists and other musicians, a system based on them can be readily kept current. This paper investigates how to utilized such resources for associating meaning with chords.

A motivation for this work comes from the context of songwriting. Given lyrics one has written, the challenge is to come up with the structure of the accompaniment, such as chord sequences that might be strummed and/or a series of notes to be played at various points of the song. Although the main consideration is in composing music that sounds good when played, it is often desirable for the music to convey a mood that complements the lyrics. The approach used here could be used to suggest chord sequences that might convey moods suitable for a particular set of lyrics. It could also aid in the reverse direction to aid songwriters who proceed from melody to lyrics, but this would require elaborate natural language generation support [7] to produce coherent lyrics. This is a follow-up to our previous work [15], which presents a simple approach for learning the meaning of chord sequences from associated lyrics. There, co-occurrence statistics are maintained over chord sequences and meaning tokens to determine significant associations. To improve the associations between chords and lyrics, we use tools developed for machine translation, which rely upon word alignments discovered in parallel corpora. This is not to suggest that learning meaning from chords is simply a matter of "translating" chord sequences into text. Our hypothesis is simply that word associations over a large collection of lyrics with chord annotations provide an effective basis for chord meanings. Given the relatively small number of chords used in practice compared to words, this is a *many-to-few* type of association (i.e., course-grained). Text categorization can be used to produce more constrained associations, as done in our previous work [15], which is more suitable for music recommendation.

We first discuss related work (§2). Subsequent sections provide details on the methodology (§3), an overview of the data (§4), and experimentation results (§5). We conclude with a summary and directions for future work (§6).

2. BACKGROUND

There has been a variety of work in music information retrieval on learning the meaning of music. Most approaches have used supervised classification in which user tags serve as ground truth for machine learning algorithms. A few have inferred the labels based on existing resources. The approaches differ mainly on the types of features used. Whitman and Ellis [20] combine audio features based on signal processing with features based on significant terms extracted from reviews for the album in question, thus an unsupervised approach relying only upon metadata about songs (e.g., author and title). Turnbull et al. [19] use similar types of audio features, but they incorporate tagged data describing the song in terms of genre, instrumentality, mood, and other attributes. Hu et al. [6] combine word-level lyrics and audio features, using tags derived from social media, filtered based on degree of affect, and then revised by humans (i.e., partly supervised). McKay et al. [11] combine class-level lyric features (e.g., part of speech frequencies and readability level) with ones extracted from user tags from social media, specifically via Last.fm. [1] They also include features for general term co-occurrence via web searches for the task of genre classification.

There has been other recent work in analyzing symbolic chord annotations. Macrae and Dixon [9] extract online chord annotations and show how they can be ranked according to sequence similarity to help filter bad annotations. McVicar et al. [13] use chord sequences from online sources to augment the task of chord recognition from audio via Hidden Markov Models (HMM's). Barthet et al. [1] extract chord annotations to augment a guitar tutor program (e.g., to illustrate chord fingering).

Lastly, there are a few approaches addressing the relations between lyrics and audio, rather than using them as separate features. Torres et al. [18] use a correlation-based approach referred to as *Canonical Correlation Analysis* (CCA) to associate lyrics with audio features. Under the CCA methodology, songs are represented in two feature spaces: a semantic annotation feature space and an audio feature space. For each space, the CCA identifies a one dimensional projection that maximizes the correlation between the projected data. The identified projections are used to construct and refine a musically meaningful vocabulary applied to assigning meaning to music. In addition, they present an approach to infer the projections under the assumption that the vector spaces are sparse. More recently, McVicar et al. [12] apply CCA to assess the correlation between lyrics and audio features as a part of an unsupervised system for quantifying mood. The system exploits a special dictionary on affect, specifically with ratings for valence (e.g., 'pleased' vs. 'frustrated') and arousal (e.g., 'excited' vs. 'sleepy'). Both approaches deal with meaning at the song level, but we address the issue of assigning meaning to smaller units. Furthermore, rather than audio features, we assign meanings to musical units more commonly used in music theory (e.g., chord progressions), making the results more accessible to musicians.

[1] See http://www.last.fm.

Overall process

1. Obtain large collection of lyrics with chord annotations
2. Extract lyrics proper with annotations from dataset
3. Convert into tab-delimited chord annotation data format
4. Determine best chord-word associations

Simple approach

4a. Fill contingency table: chord(s)/word co-occurrences
4b. Determine significant chord(s)/word associations

Preferred approach

4a. Invoke GIZA to produce chord(s)/word alignments
4b. Filter extraneous alignments

Figure 1. **Process in learning meanings for chord sequences.** The meanings are via individual words; and, $chord(s)$ is a single chord or a four-chord sequence.

Parallel corpora are vital for machine translation. Gale and Church [5] show how translation lexicons can be induced via co-occurrence statistics over contingency tables derived from such corpora. Parallel corpora have also been exploited to develop statistical machine translation systems, following pioneering work by IBM [2]. This incorporates sophisticated statistical models to account not only for co-occurrence, but also word order and degree to which alignment with multiple words are allowed (i.e., "fertility", which can account for phrasal alignments). Och and Ney [14] show that these models outperform other approaches for alignment (using GIZA, their implementation of them).

3. METHODOLOGY

Figure 1 lists the steps involved in the overall process for learning the meaning of chord sequences. First, a website for guitar instruction is downloaded to obtain a large sample of lyrics with chord annotations. The resulting data then is passed through a filter to remove extraneous text associated with the lyrics (e.g., transcriber notes). Next, the data is converted into a tabular format reflecting the chord/lyrics correspondences.

There are two approaches for obtaining the chord/word

Alternating lines:

```
C                            F
They're gonna put me in the movies
C                                     G
They're gonna make a big star out of me
      C
We'll make a film about a man that's sad
        F
    and lonely
    G7                       C
And all I have to do is act naturally
```

In-line chords:

```
[C] They're gonna put me in the [F] movies
[C] They're gonna make a big star out of [G]
    me
We'll [C] make a film about a man that's sad
    and [F] lonely
And [G7] all I have to do is act
    [C] naturally
```

Figure 2. **Chord annotation sample.** Lyrics are from "Act Naturally" by Johnny Russell, with chord annotations for the song as recorded by Buck Owens.

associations. In the simple approach, the data is converted into contingency tables from which co-occurrence statistics [10] are computed (e.g., Dice and mutual information). In the preferred approach (i.e., current NLP "best practice"), the data is formatted as a parallel corpus file and fed into a statistical word alignment system, such as GIZA. Afterwards, extraneous alignments are filtered.

3.1 Lyric Chord Annotation Data

The most critical resource required is a large set of lyrics with chord annotations. These annotations are often specified with alternative lines for chords and for the lyrics. They can also be specified with chords in-line with the lyrics. Figure 2 shows some examples of both formats. The popular website Chordie is used to obtain the data. [2] The website is crawled, and all the songs in the *chord.pere* directory are extracted (other directories are for user songbooks, etc.). There are over 65,000 files, but preprocessing complications reduces this to about 10,000 usable songs. After all processing, over 2 million distinct chord annotations are obtained. The chord annotation data is used as is (e.g., without normalization into key of C). We are working on transposing into the key of C, but we have run into key detection issues with the standard approach using key profiles [17]: presumably, that relies upon support from notes in the melody (omitted from chord annotations).

After the chord-annotated lyrics are downloaded, postprocessing is needed to ensure that user commentary and other additional material are not included. This is based on a series of regular expressions. [3] The lyrics are all converted into a tabular format that more directly reflects the line-level alignment of chords and the corresponding text. Specifically, this uses a tab-separated format with the current chord name along with words from the lyrics for which the chord applies. There will be a separate line for each chord change in the song. Figure 3 illustrates this format. This shows that special tokens are also included to indicate the end of the line and verse.

```
C  They're gonna put me in the
F  movies <l>
C  They're gonna make a big star out of
G  me <l> We'll
C  make a film about a man that's sad and
F  lonely <l> And
G7 all I have to do is act
C  naturally <l> <v>
```

Figure 3. **Sample chord annotations extracted from lyrics.** Each chord instance in figure 2 has a separate line.

Contingency Table Cells

X\Y	+	-
+	XY	$X\neg Y$
-	$\neg XY$	$\neg X \neg Y$

G versus 'film'

	+	-
+	14	231,223
-	85	1,557,047

Table 1. **Contingency tables.** The left shows the general case, and the right shows the data for chord G and 'film'.

3.2 Chord Sequence Token Co-occurrence

As mentioned above, the simple approach to deriving word associations is based on co-occurrence statistics. Several metrics have been proposed to measure this [4]; for example, *Chi Square* analysis determines the extent to which co-occurrence counts differs from that due to chance (e.g., difference of joint probability from the product of the marginals).

Given the tabular representation of the chord annotations with lyrics words, the next stage is to compute the co-occurrence statistics. This first tabulates the contingency table entry for each pair of chord and target token, as illustrated in table 1. (Alternatively, chord sequences can be of length four, as discussed later. These are tabulated using a sliding window over the chord annotations, as in n-gram analysis.) This table shows that the chord G co-occurred with the word 'film' 14 times, out of the 231,237 total instances for G. The word itself had 99 occurrences, and there were 1,557,047 instances where neither the word 'film' nor the chord G occurred. Next, a variety of co-occurrence metrics are derived using these tabulations, including Dice, Jaccard, mutual information, Chi square, and G^2 log likelihood [4, 10]. These are defined as shown in figure 4.

3.3 Alignment via GIZA

Using the IBM models [2] for word alignment has been shown to outperform simple co-occurrence metrics [14]. For this, we use the GIZA toolkit (specifically *GIZA++* version 2). Given its development for machine translation,

[2] See www.chordie.com; this was crawled in September of 2011.

[3] The Perl code for reproducing the experiments is available at www.cs.txstate.edu/%7Eto17/chord-meaning-from-lyrics.

$$Dice(X,Y) = \frac{2 \times P(X=1,Y=1)}{P(X=1)+P(Y=1)} \qquad Jaccard(X,Y) = \frac{f(X=1,Y=1)}{f(X=1,Y=1)+f(X=1,Y=0)+f(X=0,Y=1)}$$

$$MI(X,Y) = log_2 \frac{P(X=1,Y=1)}{P(X=1) \times P(Y=1)} \qquad AvgMI(X,Y) = \sum_x \sum_y \frac{P(X=x,Y=y) \times log_2(P(X=x,Y=y))}{P(X=x) \times P(Y=y)}$$

$$\chi^2(X,Y) = \sum_{i,j} \frac{P(obs[ij]-exp[ij])^2}{exp[ij]} \qquad G^2(X,Y) = 2 * \sum_{i,j} exp[ij] \times log(\frac{obs[ij]}{exp[ij]})$$

$Dice(G, film) = 0.000121$; $Jaccard(G, film) = 0.000061$; $MI(G, film) = 0.129199$
$AvgMI(G, film) = 0.0000001$; $X^2(G, film) = 0.129045$; $G^2(G, film) = 0.125770$

Figure 4. **Common co-occurrence metrics.** Using the counts shown in table 1, these statistics can be directly computed, resulting in the values shown for the chord G and word 'film'.

```
C F  They're gonna put me in the movies<l>
C G  They're gonna make a big star ... me<l>
C F  We'll make a film about a man that's \\
        sad and lonely<l>
F G7 C And all I have ... act naturally<l>
```

Figure 5. **Alternative chord annotations extracted from lyrics.** Chords for same verse line in figure 3 are together.

GIZA requires the specification of the source and target languages. Most work in statistical MT treats English as the source language and another language like French as the target. For the experiments discussed here, the chords are treated as the source and the target the words (mostly English). In our case, running the tool with the reverse direction produces negligible differences. In addition, GIZA normally includes a preprocessing stage that groups tokens in classes based on similar usages. However, that stage is omitted here because there is no context with which to determine the classes.

IBM Model 1, the simplest one in GIZA, follows: [2]

$$P_\theta(t, a|s) = P_\theta(l|s)P_\theta(a|l, s)P_\theta(t|a, l, s)$$

where s is the source language, t is the target language, l is target sentence length, a is the alignment, and θ are the overall parameters. The alignments are hidden and estimated via an HMM.

Prior to using GIZA, each column is put into separate files. Then, the toolkit preprocessing utilities convert them into a combined sentence file. (To avoid problems, lines that are too long or that contain garbage are discarded using the toolkit's utility to clean the input files.) GIZA only relies upon line correspondence in the two files when establishing alignments. Figure 5 shows how the input might be formatted. In addition, an optional step is used to group chords on the same line into sequences of four chords (e.g., $C\_D\_C\_D$), which are treated as individual tokens in the alignment.

4. OVERVIEW OF DATA

Before discussing the experiments, we present characterizations of the data involved. Naturally, lyrics are different from general English. Table 2 illustrates some differences in relative frequency for the top words. Comparing the two word frequency listings, we can see some peculiarities with respect to lyrics, such as the most common word being 'I' rather than 'the' and that 'you' moves up to the top 5. The word 'love' moves up in rank dramatically (271 to 27), and the word 'your' moves up a bit as well (from 69 to 14).

General		Lyrics		General		Lyrics	
Word	Freq	Word	Freq	Word	Freq	Word	Freq
the	.057	*i*	.033	with	.007	is	.005
and	.028	the	.028	as	.006	all	.005
of	.027	a	.028	at	.005	for	.005
to	.026	*you*	.024	this	.005	we	.005
a	.023	and	.018	they	.005	can	.004
in	.019	to	.017	be	.005	but	.004
that	.013	in	.011	are	.005	so	.004
i	.011	it	.010	have	.005	don	.004
it	.010	me	.010	we	.005	re	.004
is	.010	my	.009	but	.005	ll	.004
for	.009	of	.008	his	.005	d	.004
you	.008	on	.007	from	.004	*love*	.004
was	.008	that	.007	not	.004	no	.004
he	.007	your	.006	n't	.004	she	.004
on	.007	be	.005				

Table 2. **Top words in corpus.** General word frequencies based on Corpus of Contemporary American English [3], and word frequencies for lyrics based on Chordie.

Frequency information for common chords and for chord sequences is shown in table 3. This illustrates that the major chords dominate the others, accounting for 64% of total occurrences. The B chord is an oddball, occurring less frequently than both of the minor chords Am and Bm, as well as being just a little more frequent than its minor. Note that the top of the sequence listing is skewed towards major chords; minor chords do occur in about half of the sequence types.

5. EXPERIMENTS

Two separate groups of experiments are performed. We first present an evaluation of the meanings attached to individual chords, using the common happy-versus-sad attribution regarding major versus minor chords. We also evaluate arbitrary chord sequences, using external annotations for songs meanings. External song-level annotations

Single		Sequence		Single		Sequence	
Ch.	Freq	Seq.	Freq	Ch.	Freq	Seq.	Freq
G	.154	CGCG	.005	G7	.010	CFCF	.003
C	.124	GCGC	.005	D7	.008	DCGD	.003
D	.124	EEEE	.004	A7	.008	GCGD	.002
A	.094	DGDG	.004	E7	.007	DAGD	.002
E	.068	GDGD	.003	Gm	.006	GCDG	.002
F	.061	GDCG	.003	Eb	.006	AGDA	.002
Am	.053	EAEA	.003	Em7	.006	AEDA	.002
Em	.047	DADA	.003	Am7	.005	DGCG	.002
B	.026	ADAD	.003	Cm	.005	GDAG	.002
Bm	.022	AEAE	.003	B7	.005	CGDG	.002
Dm	.019	CGDC	.003	C7	.005	DAED	.002
Bb	.015	FCFC	.003	Cadd9	.004	GDEmC	.002

Table 3. **Chord frequency.** This shows the frequency of chords and sequences (i.e., 4-grams) in Chordie.

happy: happy, blessed, blissful, bright, golden, halcyon, prosperous, laughing, riant, cheerful, contented, content, glad, elated, euphoric, felicitous, joyful, joyous, felicitous, fortunate, glad, willing, well, chosen, felicitous

sad:, sad, bittersweet, doleful, mournful, heavyhearted, melancholy, melancholic, pensive, wistful, tragic, tragical, tragicomic, tragicomical, sorrowful, deplorable, distressing, lamentable, pitiful, sorry, bad

Figure 6. **Synonyms for happy & sad.** Via WordNet 2.1.

are used in order to keep the evaluation objective, as there is no available resource with segment-level annotations.

5.1 Results for individual chords

The first evaluation covers the meaning attached to individual chords, such as that $Cmaj$ is 'bright' whereas $Cmin$ is 'somber'). To confirm the typical associations attributed to major versus minor chords (i.e., happy and sad, respectively), we compare the inferred word associations with synonyms reflecting this dichotomy. Figure 6 shows the synonyms for 'happy' and 'sad' from WordNet.[4] The idea is to check the most common chord associated with each of these synonym sets, seeing how often a major chord is chosen for a *happy* word versus a minor chord for a *sad* word.

Specifically, we tabulate the average metric assigned to true and false positives for major versus minor chords. Figure 7 summarizes the result. For major chords, synonyms for 'happy' are assigned an average score of 81.1 (using X^2), whereas synonyms for 'sad' are assigned an average score of 39.2. Likewise, for minor chords, synonyms for 'sad' have an average score of 77.7, compared to 62.1 for 'happy'. As a baseline, a random value was used in place of the co-occurrence metric. As shown in the figure, there are much fewer true positives for the major chords (e.g., average scores for good versus bad nearly the same).

[4] See http://wordnet.princeton.edu.

Total
186 cases with score 12541.236 (avg 67.426)
Major
good: 81 with 6573.275 (avg **81.152**) (A,contented)
bad: 35 with 1326.313 (avg 37.895) (C,wistful)
baseline: average scores *52.4* and 51.4, respectively
Minor
good: 19 with 1476.133 (avg **77.691**) (Bm,tragic)
bad: 51 with 3165.515 (avg 62.069) (Am,bright)
baseline: average scores *32.9* and 43.7, respectively

Figure 7. **Evaluation of individual chord meanings.** This tests how well the metric decides whether synonyms for 'happy' ('sad') should go with a major (minor) chord.

5.2 Results for chord sequences

To evaluate the performance in learning chord sequence meaning, we compare the output against the Mood Tag Dataset (MTD) prepared by Hu et al. [6].[5] Table 4 lists the meaning categories used in the MTD, along with the words used to define the categories. For example, category $G11$ is for sincerity and is defined in terms of 'earnest' and 'heartfelt'. This data set only provides song-level annotations, so we count how often the inferred chord sequence meanings match the song-level meanings for all the songs incorporating the chord sequence. For example, if a particular song contains 10 distinct chord sequences, and if six of the sequences were labeled with the meaning category corresponding to the song annotation, then the score for the song would be 0.6. As the MTD categories are defined in terms of words, we check for word overlap from the top words associated with a chord sequence with those from the meaning category. Although a lenient measure, the word-chord alignment process being evaluated has the handicap of dealing with over 10,000 meaning categories (i.e., all lyric words).

To test against the MTD, we just need the chord annotations for each of the songs covered. The annotations are for specific combinations of artist and album, so the songs are downloaded individually via the web interface to ensure the right version is used (if available). Out of 3,470 songs that are annotated, only 2,160 chord annotation files were obtained. Songs can be labeled with more than one category. If so, when verifying whether a chord is a match, we check the associated word for membership in any of the lists. The results are promising when using GIZA for the alignment using special tokens for chord sequences. The resulting alignment shows high precision, specifically at 89.5% (1,779 chord sequences out of 1,987). However, this comes at the expense of recall, with no suggestions for many of the chord sequences. In comparison, using average mutual information yields about 70,000 more taggings, but the precision drops to 20%. The baseline for this is 25.9%, which is the relative frequency for the most common category ($G12$).

[5] This dataset was used in MIREX-2011. See www.music-ir.org/mirex/wiki/2011:Audio_Tag_Classification.

Label	Freq	Examples
G12	.259	calm, comfort, quiet, ... tranquility
G15	.182	sad, sadness, unhappy, ..., sad song
G5	.115	happy, happiness, ..., mood: happy
G32	.095	romantic, romantic music
G2	.084	upbeat, gleeful, ...
G16	.073	depressed, blue, dark, ... gloomy
G28	.039	anger, angry, choleric, ...
G17	.028	grief, heartbreak, ... sorrowful
G14	.022	dreamy
G6	.022	cheerful, cheer up, ... sunny
G8	.018	brooding, contemplative, ... wistful
G29	.018	aggression, aggressive
G25	.012	angst, anxiety, ... nervous
G9	.009	confident, encouraging, ... optimistic
G7	.007	desire, hope, hopeful, ...
G11	.006	earnest, heartfelt
G31	.006	pessimism, cynical, pessimistic, ...
G1	.005	excitement, exciting, exhilarating, ...

Table 4. **Mood Tag Dataset.** Categories for MTD along with sample words used to define them. $Freq$ gives relative frequency, out of 6,490 total assignments.

6. CONCLUSION

This paper has demonstrated how to learn the meaning of chord sequences from lyrics annotated with chords. Two separate approaches have been illustrated. The simple approach uses co-occurrence statistics derived from contingency tables. The preferred approach uses word alignment tools designed for statistical machine translation.

For future work, we will look into additional aspects of music as features for modeling meaning (e.g., tempo and note sequences). In addition, as this approach could be used to suggest chord sequences that convey moods suitable for a particular set of lyrics, future work will investigate its use as a songwriting aid; in fact, this was the original motivation for the research.

By using resources intended for guitarists, the current work is more suitable for popular music than other types (e.g., classical). A long-term research goal is to develop a framework for learning similar associations from scores that include lyrics (e.g., operas). Other long-term aspects to be addressed include getting access to more data and integrating audio analysis into the process. In principle, voice recognition over lyrics could ameliorate sparse data problem, provided that the natural noise in songs can be sufficiently filtered.

Acknowledgments Hupahu Ballard helped validate our experiments. Scott Heftler provided useful tips on the processing. Hu Xiao provided access to the MTD. Douglas Eck suggested the extension using speech recognition over audio.

7. REFERENCES

[1] M. Barthet, A. Anglade, G. Fazekas, S. Kolozali, and R. Macrae. Music recommendation for music learning: Hotttabs, a multimedia guitar tutor. In *Proc. Workshop on Music Recommendation and Discovery (WOMRAD-2011)*, pages 7–13, 2011.

[2] P. F. Brown, V. J. Pietra, S. A. D. Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–311, 1993.

[3] M. Davies. The 385+ million word corpus of contemporary American English (1990-2008+): Design, architecture, and linguistic insights. *International Journal of Corpus Linguistics*, 14:159–90, 2009.

[4] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[5] W. A. Gale and K. W. Church. Identifying word correspondences in parallel texts. In *Fourth DARPA Workshop on Speech and Natural Language*, pages 152–157, 1991.

[6] X. Hu, J. S. Downie, and A. F. Ehman. Lyric text mining in music mood classification. In *Proc. ISMIR*, pages 411–6, 2009.

[7] D. Jurafsky and J. Martin. *Speech and Language Processing*. Prentice Hall, Upper Saddle River, New Jersey, 2000.

[8] G. Leech. *Semantics*. Middlesex, Penguin Books, 1974.

[9] R. Macrae and S. Dixon. Guitar tab mining, analysis and ranking. In *Proc. ISMIR, Miami, Florida.*, 2011.

[10] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts, 1999.

[11] C. McKay, J. A. Burgoyne, et al. Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features. In *Proc. ISMIR*, 2010.

[12] M. McVicar, T. Freeman, and T. D. Bie. Mining the correlation between lyrical and audio features and the emergence of mood. In *Proceedings ISMIR*, 2011.

[13] M. McVicar, Y. Ni, R. Santos-Rodriguez, and T. D. Bie. Leveraging noisy online databases for use in chord recognition. In *Proc. 12th International Society on Music Information Retrieval (ISMIR)*, 2011.

[14] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[15] T. O'Hara. Inferring meaning of chord sequences from lyrics. In *Proc. Workshop on Music Recommendation and Discovery (WOMRAD-2011)*, pages 40–43, 2011.

[16] C. Schmidt-Jones and R. Jones, editors. *Understanding Basic Music Theory*. Connexions, 2007. http://cnx.org/content/col10363/latest.

[17] D. Temperley. A Bayesian approach to key-finding. In C. Anagnostopoulou, M. Ferrand, and A. Smaill, editors, *Music and Artificial Intelligence*, pages 195–206. Berlin, Springer-Verlag, 2002.

[18] D. Torres, D. Turnbull, L. Barrington, and G. Lanckriet. Identifying words that are musically meaningful. In *Proc. ISMIR*, September 2007.

[19] D. Turnbull, L. Barrington, et al. Semantic annotation and retrieval of music and sound effects. *IEEE TASLP*, 16 (2), 2008.

[20] B. Whitman and D. Ellis. Automatic record reviews. In *Proc. ISMIR*, 2004.

FACILITATING COMPREHENSIVE BENCHMARKING EXPERIMENTS ON THE MILLION SONG DATASET

Alexander Schindler, Rudolf Mayer, and Andreas Rauber

Institute of Software Technology and Interactive Systems, Vienna University of Technology
{schindler,mayer,rauber}@ifs.tuwien.ac.at

ABSTRACT

The Million Song Dataset (MSD), a collection of one million music pieces, enables a new era of research of Music Information Retrieval methods for large-scale applications. It comes as a collection of meta-data such as the song names, artists and albums, together with a set of features extracted with the The Echo Nest services, such as loudness, tempo, and MFCC-like features.

There is, however, no easily obtainable download for the audio files. Furthermore, labels for supervised machine learning tasks are missing. Researchers thus are currently restricted on working solely with these features provided, limiting the usefulness of MSD. We therefore present in this paper a more comprehensive set of data based on the MSD, allowing its broader use as benchmark collection. Specifically, we provide a wide and growing collection of other well-known features in the MIR domain, as well as ground truth data with a set of recommended training/test splits.

We obtained these features from audio samples provided by 7digital.com, and metadata from the All Music Guide. While copyright prevents re-distribution of the audio snippets per se, the features as well as metadata are publicly available on our website for benchmarking evaluations. In this paper we describe the pre-processing and cleansing steps applied, as well as feature sets and tools made available, together with first baseline classification results.

1. INTRODUCTION

Music Information Retrieval (MIR) research has historically struggled with issues of publicly available benchmark datasets that would allow for evaluation and comparison of methods and algorithms on the same data base. Most of these issues stem from the commercial interest in music by record labels, and therefore imposed rigid copyright issues, that prevent researchers from sharing their music collections with others. Subsequently, only a limited number of data sets has risen to a pseudo benchmark level, i.e. where most of the researchers in the field have access to the same collection.

Another reason identified as a major challenge for providing access to research data in general is the lack of esteem and valuation of these kind of activities. While preparing, maintaining and providing access to massive data collections requires significant investments in terms of system maintenance and data (pre-)processing, it is considered administrative rather than research work (in spite of several even research-affine challenges emerging during such activities), and thus does not gain acceptance in classical research-oriented publication venues. Such lack of career rewards is one of the many factors, next to legal limitations and lack of expertise, limiting sharing of research data [7]. Several initiatives have been started in the Research Infrastructures area to mitigate this problem and foster collaborative research. These areas span across virtually all topical areas, from Astronomy, via meteorology, chemistry to humanities [1] .

A recent effort in the MIR domain has lead to the compilation of the Million Song Dataset [3] (MSD). It provides a database of meta-data for a collection of one million songs, such as the song name, artists and album. In addition, a number of descriptive features extracted with the services from The Echo Nest[2] are provided. These features include tempo, loudness, timings of fade-in and fade-out, and MFCC-like features for a number of segments. Moreover, a range of other meta-data has been published recently, such as song lyrics (for a subset of the collection), or tags associated to the songs from Last.fm[3] .

The MSD enables researchers to test algorithms on a large-scale collection, thus allowing to test them on more real-world like environments. However, there are no easily obtainable audio files available for this dataset, and therefore, researchers are practically restricted to benchmarking on algorithms that work on top of features, such as recommendation of classification, but can not easily develop new or test existing feature sets on this dataset. The availability of just one feature set also does not allow an evaluation across multiple feature sets. As previous studies showed, however, there is no single best feature set, but their performance depends very much on the dataset and the task.

We therefore aim to alleviate these restrictions by pro-

[1] http://ec.europa.eu/research/infrastructures/
[2] http://the.echonest.com
[3] http://www.last.fm

viding a range of features extracted for the Million Song Dataset, such as MFCCs and a set of low-level features extracted with the jAudio feature extraction software as well as the Marsyas framework [13], and the Rhythm Patterns and derived feature sets [9]. To this end, we first obtained audio samples for the MSD by using the content provider 7digital.

A second shortcoming of the MSD is that it does not, at the moment, contain a mapping to categorisation such as genres. Thus, experimental evaluations such as musical genre classification, a popular task in MIR research, are not possible. We therefore further propose a partitioning of the dataset into a set of genres obtained from the All Music Guide [4]. Specifically, we created a partitioning on two levels of detail, with 13 top-level-genres and 25 sub-genres, and propose a number of splits for training and test sets, with different filters, allowing several tasks for evaluation.

Both the feature sets and the partitioning into genres are available from our website [5]. The features are stored in the WEKA Attribute-Relation File Format (ARFF) [14], with one attribute being the unique identifier of the song in the MSD. We further provide a set of scripts to match the features with the genre mapping so that they can be used for classification experiments.

The remainder of this paper is structured as follows. Section 2 introduces the dataset and the properties of the audio samples, while Section 3 describes the sets of features extracted from them. Section 4 gives details on the genre assignment obtained, and in Section 5, we describe benchmark partitions and how we aim to facilitate exchange between researchers. Finally, we provide conclusions in Section 6

2. THE DATASET

The Million Song Dataset contains in the meta-data a unique identifier for an audio sample at the content provider 7digital [6]. For some songs no sample could be downloaded, as the identifier was unknown to 7digital. We thus obtained a total of 994,960 audio samples, i.e. a coverage of 99.5% of the dataset; the list of missing audio samples is available on the website. This points to an important issue related to the use of external on-line resources for scientific experimentation. Especially when the provider is not genuinely interested in the actual research performed, there is little motivation to maintain the data accessible in unmodified manners, and is thus susceptible to changes and removal. Thus, maintaining a copy of a fixed set of data is essential in benchmarking to allow the evaluation of newly developed feature sets, and for acoustic evaluation of the results.

In total, the audio files account for approximately 625 gigabyte of data. The audio samples do not adhere to a common encoding quality scheme, i.e. they differ in length and quality provided. Figure 1 shows a plot of the sample lengths; please note that the scale is logarithmic. It can

[4] http://allmusic.com
[5] http://www.ifs.tuwien.ac.at/mir/msd/
[6] http://www.7digital.com

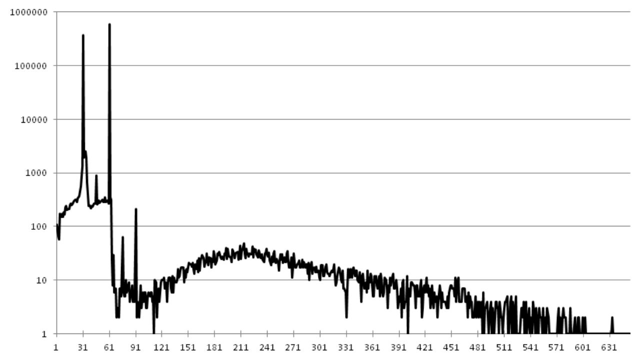


Figure 1: Distribution of sample length

be observed that there are two peaks at sample lengths of 30 and 60 seconds with 366,130 and 596,630 samples, respectively, for a total of 96,76% of all the samples. These shorter snippets normally contain a section in the middle of the song. Many other well-known collections in the MIR domain as well contain only 30 second snippets, and feature extraction algorithms normally deal with this.

Table 1: Audio properties of 7Digital Samples

Samplerate		
22	768,710	77,26%
44	226,169	22,73%
other	81	0,01%
Bitrate		
128	646,120	64,94%
64	343,344	34,51%
other (VBR)	5,494	0,55%
Channels		
Mono	6,342	0.64%
Stereo	150,779	15.15%
Joint stereo / dual channel	837,839	84.21%

Table 1 gives an overview on the audio quality of the samples. The majority, more than three quarters, of the audio snippets have a sample rate of 22khz, the rest has a sample rate of 44khz (with the exception of 81 songs, of which the majority have 24 and 16khz). Regarding the bitrate, approximately two-thirds of the songs are encoded with 128kbit, the majority of the rest with 64kbit; only about half a percent of the songs come with higher (192 or 320kbps) or variable bitrates (anywhere between 32 and 275kpbs). Almost all samples are provided in some form of stereo encoding (stereo, joint stereo or dual channels) – only 0.6% of them have only one channel. As these characteristics, specifically the sample rate, may have significant impact on the performance of the data analysis algorithms, we must consider these for stratification purpose when designing the benchmark splits.

3. FEATURE SETS

We extracted a wide range of audio features from the samples provided, namely features provided by the jAudio feature extraction software (which is a part of the jMIR pack-

Table 2: Overview on features extracted from the MSD samples. *Dim.* denotes the dimensionality, *Deriv.* derivatives computed from the base features

#	Feature Set	Extractor	Dim	Deriv.
1	MFCCs [12]	MARSAYS	52	
2	Chroma [6]	MARSAYS	48	
3	timbral [13]	MARSAYS	124	
4	MFCCs [12]	jAudio	26	156
5	Low-level spectral features [11] (Spectral Centroid, Spectral Rolloff Point, Spectral Flux, Compactness, and Spectral Variability, Root Mean Square, Zero Crossings, and Fraction of Low Energy Windows)	jAudio	16	96
6	Method of Moments [11]	jAudio	10	60
7	Area Method of Moments [11]	jAudio	20	120
8	Linear Predictive Coding [11]	jAudio	20	120
9	Rhythm Patterns [9]	rp_extract	1440	
10	Statistical Spectrum Descriptors [9]	rp_extract	168	
11	Rhythm Histograms [9]	rp_extract	60	
12	Modulation Frequency Variance Descriptor [10]	rp_extract	420	
13	Temporal Statistical Spectrum Descriptors [10]	rp_extract	1176	
14	Temporal Rhythm Histograms [10]	rp_extract	420	

age [11]), the MARSYAS feature extractor [13], and the Rhythm Patterns family of feature sets [9]. An overview on these features is given in Table 2.

The jAudio software provides a range of 28 features associated with both the frequency and time domains. It includes several intermediate-level musical features, mainly related to rhythm, as well as lower-level signal processing-oriented features. It also provides an implementation of MFCC features [12], using 13 coefficients. jAudio computes in general mean and standard deviations over the sequence of frames, and provides for most measures also derivatives, i.e. additional statistical moments over the basic measures. For the extraction, we utilised jAudio as bundled in the jMIR 2.4 release [7].

A very popular audio feature extraction system is MARSAYS, one of the first comprehensive software packages to be available to MIR researchers. A very popular set from this audio extractor is the so-called "timbral" set, which is composed of 13 MFCC coefficients, and the twelve chroma features and the average and minimum chroma value, and the four low-level features zero crossings, and rolloff, flux and centroid of the spectrum. For these 31 values, four statistical moments are computed, resulting in a 124 dimensional vector. For the extraction, we utilised MARSYAS version 0.4.5 [8].

The Rhythm Patterns and related features sets are extracted from a spectral representation, partitioned into segments of 6 sec. Features are extracted segment-wise, and then aggregated for a piece of music computing the median (Rhythm Patterns, Rhythm Histograms) or mean (Statistical Spectrum Descriptors, Modulation Frequency Variance Descriptor) from features of multiple segments. For the extraction, we employed the Matlab-based implementation, version 0.6411 [9].

It is intentional that we provide two different versions of the MFCCs features, as this will allow for interesting insights in how these implementations differ on various MIR tasks.

3.1 Publication of Feature Sets

All features described above are available on our website for download, encoded in the WEKA Attribute-Relation File Format (ARFF) [14]. The features are available under the Creative Commons Attribution-NonCommercial-ShareAlike 2.0 Generic License [10].

To allow high flexibility when using them, we provide one ARFF file for each type of features; these can then be combined in any particular way when performing experimental evaluations. A set of scripts is provided as well on the website for this. In total, the feature files account for approximately 40 gigabyte of uncompressed text files. The feature files contain the numeric values for each feature, and additionally the unique identifier assigned in the MSD. This way, it is possible to generate various feature files with different ground truth assignments; we again provide scripts for this. The proposed assignment into genres for genre classification tasks is described in Section 4.

Further feature sets to be extracted and provided will include e.g. MIRtoolbox [8] or M2k [5].

4. ALLMUSIC DATASETS

The All Music Guide (AMG) [4] was initiated by an archivist in 1991 and emerged 1995 from its book form into a database which can be accessed through the popular commercial Web page allmusic.com. The Web page offers a wide range of music information, including album reviews, artist biographies, discographies well as classification of

[7] available from `http://jmir.sourceforge.net/`

[8] available from `http://sourceforge.net/projects/marsyas/`

[9] available from `http://www.ifs.tuwien.ac.at/mir/downloads.html`

[10] `http://creativecommons.org/licenses/by-nc-sa/2.0/`

albums according to genres, styles, moods and themes. Information is provided and curated by experts.

Genre information is very coarse, provided as a single tag for each album. Further the two main categories Pop and Rock are combined into a single genre 'Pop/Rock'. Additionally to genre labels, style tags are provided allowing for a more specific classification. Usually multiple style tags are applied for each album, but unfortunately no weighting scheme can be identified and in many cases only one tag is provided. Style tags also tend to be even more generic than genre labels. Especially non-American music is frequently tagged with labels describing country or region as well as the language of the performing artist. Instrumentation, situational descriptions (e.g. *Christmas*, *Halloween*, *Holiday*, etc.) as well as confessional or gender attributes (e.g. *Christian*, *Jewish*, *Female*, etc.) are also provided. Unfortunately these meta-descriptive attributes are not used as isolated synonyms, but are concatenated with conventional style information (e.g. *Japanese Rock*, *Christian Punk*, *Classic Female Blues*, etc.).

Allmusic.com assembles styles to meta-styles which can be interpreted as sub genres used to diversify the major genre labels. Meta-styles are not distinctive and are used overlapping in many meta-styles (e.g. *Indie Electronic* is contained in the meta-styles *Indie Rock*, *Indie Pop* and *Alternative/Indie Rock*).

4.1 Data Collection

Data was collected automatically from Allmusic.com using direct string matching to query for artist-release combinations. From the resulting Album Web page genre and style tags were collected.

We were able to collect 21 genre labels for 62,257 albums which initially provided genre tags for 433,714 tracks. Style tags were extracted attributing only 42,970 albums resulting in 307,790 labeled tracks. An average of 3.25 tags out of a total of 905 styles were applied to each album, but 5,742 releases were only tagged with a single style label. The most popular genre with 32,696 tagged albums, was *Pop/Rock* - this is 10% more as the sum of all other genres. Referring to tracks the difference rises to 30%. Further, the granularity of Rock is very scarce, including Heavy Metal, Punk, etc. A similar predominating position of this genre as well as was also reported by [2]. The most popular style tag is *Alternative/Indie Rock* applied to 12,739 albums, which is more than twice as much as the second popular style *Alternative Pop/Rock*. About 120 tags describe the country of the performing artist or the language of the interpretation - the most common among them is *Italian Music* which has been applied to 610 albums.

4.2 Allmusic Genre Dataset

The Allmusic Genre Dataset is provided as an unoptimized expert annotated ground truth dataset for music genre classification. We provide two partitions of this set. The *MSD Allmusic Genre Dataset (MAGD)* assembles all collected genres including generic and small classes.

Table 3: MSD Allmusic Genre Dataset (MAGD) - upper part represents the MSD Allmusic Top Genre Dataset (Top-MAGD)

Genre Name	Number of Songs
Pop/Rock	238,786
Electronic	41,075
Rap	20,939
Jazz	17,836
Latin	17,590
R&B	14,335
International	14,242
Country	11,772
Reggae	6,946
Blues	6,836
Vocal	6,195
Folk	5,865
New Age	4,010
Religious	8814
Comedy/Spoken	2067
Stage	1614
Easy Listening	1545
Avant-Garde	1014
Classical	556
Childrens	477
Holiday	200
Total	422,714

The second partition - *MSD Allmusic Top Genre Dataset (Top-MAGD)* - consists of 13 genres - the 10 major genres of Allmusic.com (Pop/Rock, Jazz, R&B, Rap, Country, Blues, Electronic, Latin, Reggae, International) including the three additional genres Vocal, Folk, New Age (see Table 3). Generic genres as well as classes with less than 1% of the number of tracks of the biggest class Pop/Rock are removed. Due to the low number of tracks, the Classical genre is also removed from the Top Genre dataset.

4.3 Allmusic Style Dataset

The Allmusic Style Dataset attempts to more distinctively separate the collected data into different sub-genres, alleviating predominating classes. For the compilation of the dataset genre labels were omitted and solely style tags were used. In a first step metastyle description as presented on the Allmusic.com Web site were used to map multiple style tags to a single genre name - in this case we used the metastyle name. This simple aggregation approach generated a total of 210 genre labels many of them highly generic or hierarchical specializing (e.g. *Electric Blues* and *Electric Chicago Blues*. The *MSD Allmuisc Metastyle Dataset - Multiclass (MAMD)* was derived from these 210 resulting metaclasses. Each track was matched to one or more metaclasses according to its style tags. In a second step we removed from the initial set of 905 style tags all confessional, situational and language specific entries. Regional tags were discarded if they do not refer to a specific traditional cultural music style (e.g. *African Folk*). Popular music attributed with regional information was dis-

Table 4: The MSD Allmusic Style Dataset (MASD)

Genre Name	**Number of Songs**
Big Band	3,115
Blues Contemporary	6,874
Country Traditional	11,164
Dance	15,114
Electronica	10,987
Experimental	12,139
Folk International	9,849
Gospel	6,974
Grunge Emo	6,256
Hip Hop Rap	16,100
Jazz Classic	10,024
Metal Alternative	14,009
Metal Death	9,851
Metal Heavy	10,784
Pop Contemporary	13,624
Pop Indie	18,138
Pop Latin	7,699
Punk	9,610
Reggae	5,232
RnB Soul	6,238
Rock Alternative	12,717
Rock College	16,575
Rock Contemporary	16,530
Rock Hard	13,276
Rock Neo Psychedelia	11,057
Total	273,936

carded due to extensive genre overlaps (e.g. *Italian Pop* ranges from Hip-Hop to Hard-Rock). Finally, we successively merged these genres into general descriptive classes until we finalized the dataset into the *MSD Allmusic Style Dataset (MASD)* presented in Table 5. For completeness we also provide the *MSD Allmuisc Style Dataset - Multiclass (Multi-MASD)*. This set contains the pure track-style mapping as collected from Allmusic.com.

5. BENCHMARK PARTITIONS

Influenced by the tremendous experience in the text classification domain, specifically with the landmark Reuters-21578 corpus, we provide a number of benchmark partitions that researcher can use in their future studies, in order to facilitate repeatability of experiments with the MSD beyond x-fold cross validation. We also encourage and provide a platform for exchange of results obtained and new partitions created via our website.

We provide the following categories of splits:

- Splits with all the tow ground truth assignments into genre and style classes, described in Section 4.
- Splits with just the majority classes from these two ground truth assignments.
- Splits considering the sample rate of the files, i.e. only the 22khz samples, only the 44khz samples, and a set with all audio files.

Table 5: Classification results on MSD Allmusic Guide Style dataset (MASD), 66% training set split

Dataset	NB	SVM	k-NN	DT	RF
MFCC (4)	**15.04**	20.61	*24.13*	14.21	18.90
Spectral (5)	*14.03*	17.91	13.84	12.81	17.21
Spectral Derivates (5)	11.69	*21.98*	16.14	*14.09*	*19.03*
MethodOfMoments (6)	13.26	16.42	12.77	11.57	14.80
LPC (8)	13.41	17.92	15.94	11.97	16.19
SSD (10)	13.76	**27.41**	**27.07**	**15.06**	**20.06**
RH (11)	12.38	17.23	12.46	10.30	13.41

In particular, we provide the following size partitions:

- "Traditional" splits into training and test sets, with 90%, 80%, 66% and 50% size of the training set, applying stratification of the sampling to ensure having the same percentage of training data per class, which is important for minority classes.
- A split with a fixed number of training samples, equally sized for each class, with 2,000 and 1,000 samples per class for the genre and style data sets, respectively. This excludes minority classes with less than the required number of samples.

Finally, we apply stratification on other criteria than just the ground truth class, namely:

- Splits into training and test sets with an artist filter, i.e. avoiding to have the same artist in both the training and test set; both stratified and non-stratified sets are provided
- As above, but with an album filter, i.e. no songs from the same album appear in both training and test set, to account for more immediate production effects
- As above, but with a time filter, i.e. for each genre using the earlier songs in the training set, and the later releases in the test set.

Full details on the results for predictions for the different tasks outlined above are available on our website. In this paper, we discuss the results of a musical genre classification experiment on the MSD Allmusic Guide Style Dataset (MASD) with a frequently-used 2/3 training and 1/3 test set split.

Table 5 shows classification accuracies obtained with five different classifiers using the WEKA Machine Learning Toolkit [14], version 3.6.6. Specifically, we employed Naïve Bayes, Support Vector Machines (polynomial kernel with exponent 1), k-nearest Neighbours with k=1, a J48 Decision Tree, and Random Forests, with the default settings. Due to space limitations, we selected the most interesting of the feature sets. The number in parentheses after the feature set name corresponds to the number given in Table 2. Bold print indicates the best, italics the second best result per feature set (column-wise).

For this classification task, we have 25 categories, for which the biggest "Pop Indie" accounts for 6.60% of the songs, which is thus the lowest baseline for our classifiers. It can be noted from the results that the jMIR MFFC features provide the best results on the Naïve Bayes classifier, followed by the jMIR low-level spectral features. However, all results on this classifier are just roughly twice as good as the baseline identified above, and low in absolute terms. Better results have been achieved with Support Vector Machines and k-NN classifiers, on both the Statistical Spectrum Descriptors achieve more than 27% accuracy. Also on the other two classifiers, Random Forests and Decision Trees, the SSD feature set is the best, followed by either the derivatives of the jMIR spectral features, or the jMIR MFFC implementation.

6. CONCLUSION AND FUTURE WORK

Benchmarking is an important aspect in experimental sciences – results reported by individual research groups need to be comparable. Important aspects of these are common platforms to exchange these results, and datasets that can be easily shared among researchers, together with a set of defined tasks. The MIR community has traditionally suffered from only few (and small) data collections being available, also complicated by stringent copyright laws on music. Recently, the publication of the Million Song Dataset has aimed at alleviate these issues. The dataset comes with associated metadata and a basic set of features extracted from the audio. Other modalities such as lyrics have subsequently been provided for (parts of the) collection.

To increase the usefulness of the dataset, we presented a wide range of other features extracted from the audio signals, and enabled musical genre classification tasks by providing a ground-truth annotation to a significant part of the dataset. To foster exchange between different researchers, we defined a number of tasks by providing standardised splits between training and test data.

Our goal is to create a collaborative research environment for sharing data and adding new features (by inviting other researchers to submit their algorithms), also for other data sets besides the MSD. We will extend the features provided also by features for each short segment of the audio analysed, similar to the Echonest features currently available for the MSD, which will allow for time-based analysis over a song. The platform shall also allow sharing of results. This is an important aspect in experimental research, as researchers normally know well how to tune their own algorithms and to optimise parameters to achieve better results – but when they utilise other algorithms for comparison, we often simply apply the default parameter settings, which does not create a realistic baseline Such a collaborative platform will allow fairer comparisons, relieving researchers from the need to run all permutations of feature extractions and settings, and will enable moving towards evaluation platforms as described in [1].

7. REFERENCES

[1] Timothy G. Armstrong, Alistair Moffat, William Webber, and Justin Zobel. Improvements that don't add up: ad-hoc retrieval results since 1998. In *Proc. of the ACM Conf. on Information and Knowledge Management*, CIKM '09, New York, USA, 2009.

[2] James Bergstra, Re Lacoste, and Douglas Eck. Predicting genre labels for artists using freedb. In *Proc. of the Int. Conf. on Music Information Retrieval*, 2006.

[3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proc. of the Int. Conf. on Music Information Retrieval*, 2011.

[4] Dave Datta. Managing metadata. In *Proc. of the Int. Conf. on Music Information Retrieval*, Paris, France, October 2002.

[5] J. Stephen Downie, Joe Futrelle, and David K. Tcheng. The international music information retrieval systems evaluation laboratory: Governance, access and security. In *Proc. of the Int. Conf. on Music Information Retrieval*, Barcelona, Spain, October 10-14 2004.

[6] M. Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Trans. on Audio, Speech & Language Processing*, 14(5):1783–1794, 2006.

[7] Michel Jubb, Alma Swan, and Sheridan Brown. To share or not to share. Publication and Quality Assurance of Research Data Outputs. Technical report, Research Information Network, 2008.

[8] Olivier Lartillot, Petri Toiviainen, and Tuomas Eerola. A matlab toolbox for music information retrieval. pages 261–268. 2008.

[9] Thomas Lidy and Andreas Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proc. of the Int. Conf. on Music Information Retrieval*, pages 34–41, London, UK, September 11-15 2005.

[10] Thomas Lidy, Carlos N. Silla, Olmo Cornelis, Fabien Gouyon, Andreas Rauber, Celso A. A. Kaestner, and Alessandro L. Koerich. On the suitability of state-of-the-art music information retrieval methods for analyzing, categorizing, structuring and accessing non-western and ethnic music collections. *Signal Processing*, 90(4):1032–1048, 2010.

[11] Cory Mckay. *Automatic Music Classification with jMIR*. PhD thesis, McGill University, Canada, 2010.

[12] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[13] George Tzanetakis and Perry Cook. Marsyas: a framework for audio analysis. *Organized Sound*, 4(3):169–175, December 1999.

[14] Ian H. Witten, Eibe Frank, Len Trigg, Mark Hall, Geoffrey Holmes, and Sally Jo Cunningham. Weka: Practical Machine Learning Tools and Techniques with Java Implementations, 1999.

DECODING TEMPO AND TIMING VARIATIONS IN MUSIC RECORDINGS FROM BEAT ANNOTATIONS

Andrew Robertson
School of Electronic Engineering and Computer Science
andrew.robertson@eecs.qmul.ac.uk

ABSTRACT

This paper addresses the problem of determining tempo and timing data from a list of beat annotations. Whilst an approximation to the tempo can be calculated from the inter-beat interval, the annotations also include timing variations due to expressively timed events, phase shifts and errors in the annotation times. These deviations tend to propagate into the tempo graph and so tempo analysis methods tend to average over recent inter-beat intervals. However, whilst this minimises the effect such timing deviations have on the local tempo estimate, it also obscures the expressive timing devices used by the performer. Here we propose a more formal method for calculation of the optimal tempo path through use of an appropriate cost function that incorporates tempo change, phase shift and expressive timing.

1. INTRODUCTION

Musicologists are interested in how individual performers convey musical expression, which can manifest itself through control of dynamics, instrumental timbre and through tempo and timing variation. Honing [9] describes performed rhythm as consisting of three aspects: the rhythmic pattern, the tempo or speed of the performed pattern, and expressive timing deviations. Whilst the tempo can be understood as the rate at which beats occur, the onset time of a note is also dependent upon deviation from strict metrical time. Indeed, deviation from the score is a crucial aspect of musical performance and these variations have been found to be systematic [15]. Vercoe [16] characterises the relationship between score and performance "as if the musical score acts as a carrier signal for other things we prefer to process".

Gouyon and Dixon [8] present the difficulty of analysing performance data in that "the two dimensions of tempo and timing are projected onto the single axis of time". At the extreme, any tempo change can be represented as a sequence of timing changes and vice-versa. One simple way to represent tempo is to use the instantaneous inter-beat interval, but in doing so all expressive timing information has also been included. Desain and Honing [6] criticise the use of such "tempo curves" as meaningful representations of timing, arguing that expressive features, such as rubato, do not scale linearly with tempo, and that timing must be understood in relation to musical structure and phrasing. Whilst a simple moving average can help to smooth this estimate, these other timing deviations remain hidden within the tempo data and there is no explanation how these two aspects of timing might relate to each other.

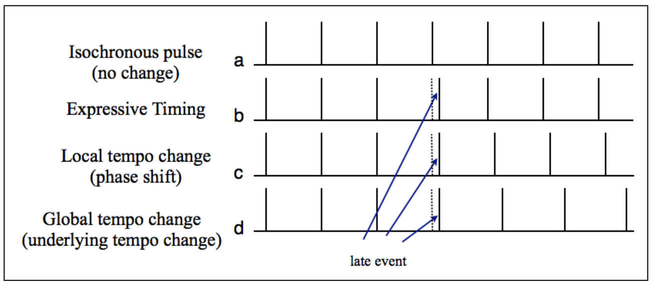

Isochronous pulse (no change)
a
Expressive Timing
b
Local tempo change (phase shift)
c
Global tempo change (underlying tempo change)
d
late event

Figure 1. Four time lines illustrating the difference between the different timing variations (after Gouyon and Dixon [8]).

Despite these difficulties, it is still possible to attribute values to the tempo as it changes over throughout a piece, albeit with some inherent uncertainty. In rock and pop music, where the tempo is often approximately steady, beat trackers are successfully used to track tempo changes and when evaluated do so relatively well when compared with human listeners performing the same task [13]. The Performance Worm [7] provides real-time visualisation of tempo and dynamics by clustering inter-onset intervals. For scored music, Müller et al. [14] generate a tempo graph by aligning a "neutral" MIDI file, in which the tempo is constant, to the audio recording through the matching of chroma-onset features [14]. The tempo graph is calculated by using using windowing techniques to compute the average tempo in each local region.

1.1 Timing variations

Our model makes use of a framework provided by Gouyon and Dixon [8], who enumerate three types of timing variation: expressively timed events, local tempo variation or phase shift, and global tempo variation or tempo change. Figure 1 shows their illustration of each of these types for a single late event where the preceding events are a series of regularly spaced events. It should be noted that at the point in time where the event happens, it is unknown which type of timing variation has occurred.

In the case of expressively timed events, the event happens early or (in this case) late, but subsequent events are unaffected with respect to timing. The displacement occurs merely for the expressively timed event, but the underlying sequence is constantly spaced. In a local tempo change (or phase shift), there is a displacement for both the event and all subsequent events. So whilst the time between events, the underlying tempo, remains constant, the phase shift represents a variation in the interval. The global tempo change (or tempo variation) occurs when there is a change to the interval duration which continues in all subsequent intervals. This would be heard as a slowing down or speeding up of the events.

2. METHODOLOGY

Intuition might suggest that once the beat times in a recording are annotated that the instantaneous tempo is thereby known directly. We can calculate the tempo at annotation i at time t_i from the beat period $t_i - t_{i-1}$. If these annotation times are in milliseconds, the tempo in beats per minute (BPM) is $60000/(t_i - t_{i-1})$. However, in practice, such a tempo graph is often jagged and then requires smoothing to extract what is taken to be the underlying tempo. The reason for this is the conflation of tempo and timing (phase) variations, described above. Thus a local tempo change or phase shift will be represented by first a global tempo change in one direction and then a reverse change in the other. The smoothing process discards information about expressively timed events and phase shifts, as there is no explicit interpretation of the annotations in terms of potential timing variations.

Here, we propose a formal solution to this problem, which calculates the optimal timing variations according to a set of associated cost functions designed to penalise tempo change, expressive timing and phase shifts. By calculating the accumulated cost across a multitude of temporal locations (phases) and tempi, we can use the well-known dynamic programming technique to then trace the solution with least cost back through the song.

2.1 Input: Annotated beat times

For simplicity, we describe here how the method works for annotations at the beat level. However, the input can also be annotations at the note level, in which case the input contains both the event time and the quantised event location in beats and bars.

We shall assume that exact beat annotations exist for the audio recording. These take the form of a list of beat times, in seconds, and may have been generated either algorithmically or by hand. One program allowing the creation of annotated audio data is Sonic Visualiser [2] [1], designed to provide visualisation of audio analysis features using the VAMP plugin format. One such plugin is a beat tracking algorithm, based on work by Davies and Plumbley [5], which automatically labels the beats. It is also possible to manipulate these annotations, so that in cases where the beat is not exactly correct, it may be pushed earlier or later. Sonic Visualiser also supports the creation of hand annotated audio data by tapping the keyboard or use of a MIDI interface. This data may then be exported as a text file.

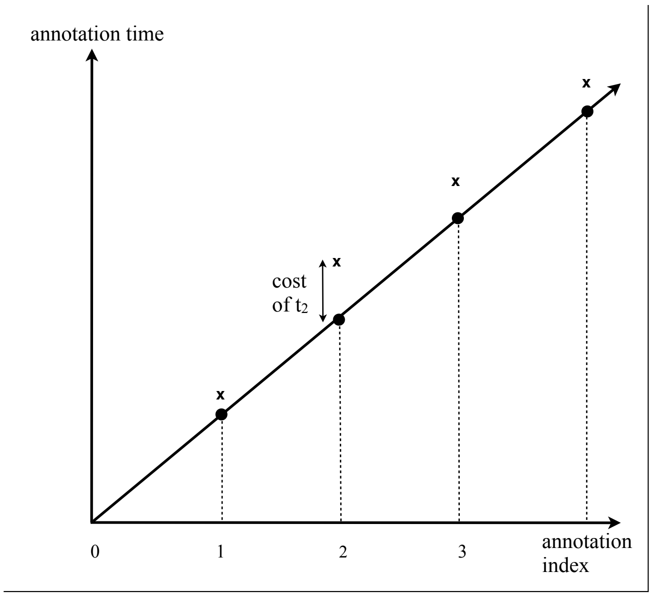


Figure 2. Illustration of the costs incurred by a sequence of isochronous pulses (circles) relative to the sequences of annotated beats $\{x_1, x_2, ...\}$. The cost for the pulse at annotation time t_2 is illustrated as the distance between the pulse and the annotation time.

The analysis proceeds on the assumption that the annotations indicate where the event actually occurred. The act of tapping along by hand, and use of the algorithmic beat tracker described above, will actually smooth the data by placing the beat towards the general trend rather than where each note onset happens. Thus if one wants to extract information about precise timing variations via this method, it would be advisable to then edit by hand so that the beat annotations are as close as possible to where the onsets occur in the audio. In the literature, a corresponding difference can also be found between predictive beat trackers which place beats causally and therefore smooth the output, and descriptive ones which place the beat after analysing the whole file and provide the ground truth of where the beat occurred [8].

2.2 Timing Transition costs

Given a set of annotated beat times as input, $\{t_0, t_1, t_2, ..\}$, we evaluate a total cost for each possible path of tempo and timing variations.

Let us define a beat path as consisting of a sequence of event times $\{\theta_0, \theta_1, \theta_2, ...\}$, each with an associated beat period of τ_i ms. These event times define the underlying beat and involve transitions in tempo and phase which incur costs. We can express each point on the path as a pair consisting of a phase location (the time of the event) and a tempo (as a beat period). Thus the point on the path corresponding to annotated time t_i is (θ_i, τ_i). Now we shall define the cost for this path and for the possible timing variations within it.

Firstly, the annotated beat time t_i might be expressively timed relative to this beat path, and the cost incurred is $|\theta_i - t_i|$, equivalent to the time difference in ms. In Figure 2, we see the cost of an annotated path relative to a series of isochronous pulses. The cost is simply the sum of the error between the two.

Secondly, the path may involve a phase shift or local tempo change. A tempo and phase pair $(\theta_{i-1}, \tau_{i-1})$ at annotation time t_{i-1} naturally implies that the next point on the path at annotation time t_i will be at $(\theta_{i-1}+\tau_{i-1}, \tau_{i-1})$. However, there may occur a phase shift of x ms, so that the next phase θ_i is in fact $\theta_{i-1}+\tau_{i-1}+x$. Then an additional cost is incurred of αx, where α is a parameter set by hand.

Thirdly, the path may involve a tempo change. Suppose we change from tempo of period τ_{i-1} to a tempo of $\tau_{i-1} + x$, making this the next tempo τ_i, we incur a cost of βx. The predicted point for such a transition would also have a phase location of $\theta_{i-1} + \tau_{i-1} + x$, although in this case due to the change in tempo.

To reflect the fact that we wish to penalise phase shifts and tempo changes, we set α and β by hand to values greater than 1. We have chosen α to be 1.4, and β to be 1.8 in practice although there is no definitive 'correct' value.

2.3 Updating the cost matrix

Let us define the cost matrix Γ_i to be all possible pairs of tempo and phase values, each with an associated cost. For each point (θ, τ) in Γ_i, we must consider all the possible transitions from points in Γ_{i-1}.

Supposing there was no change in tempo or phase, then a point (θ, τ) in Γ_{i-1} naturally suggests the next beat location at time t_i to be $\theta + \tau$ with the beat period remaining τ ms. We will employ dynamic programming to choose the minimum cost so far incurred on a path to (θ, τ) in Γ_i. This is done by working out the respective costs for all phase shifts and tempo changes to our new point and then choosing the minimum.

Observe that the point (θ, τ) in Γ_i can be reached from $(\theta - x - y, \tau - y)$ in Γ_{i-1} by a tempo transition of y ms (from $\tau - y$ to τ) and a subsequent phase shift of x ms, from the predicted event time of $\theta - x$ to θ. These incur costs of βy and αx respectively. We also need to compute the additional cost for the point, which is given by $|\theta - t_i|$, the discrepancy between the location of the beat and the annotation time. Then our full update equation is

$$\Gamma_i(\theta,\tau) = min_{x,y}\{\Gamma_{i-1}(\theta-x-y,\tau-y)+\alpha x+\beta y\}+|\theta-t_i|. \tag{1}$$

2.4 Backwards Path calculation

Having calculated the cost matrix Γ_i for each annotated beat times, t_i, we find the minimum point in the final matrix and the corresponding backwards path. Thus, we choose

$$(\theta_N, \tau_N) = min_{\theta,\tau}\Gamma_N(\theta,\tau) \tag{2}$$

Then we iterate back to find each previous point in the matrix that was chosen by Equation 1. This gives the complete path through the annotated beat times with the lowest cost for our parameters α and β. This path can be seen as the optimal explanation of the sequence of annotated beat times as a combination of tempo changes, phase shifts and expressively timed events.

2.5 Computational considerations

For our tempo analysis to be reasonably quick, we made use of some simplifications to reduce the computation time. By considering only those phase locations within occur within a fixed range either side of the beat annotation, we can discard points in the cost matrix which would almost certainly never occur. Similarly the tempo range was determined to be a fixed range either side of the interval between the two most recent annotations. These two ranges can be set by hand, depending on the nature of the piece.

Also, our data has a fixed temporal resolution. By choosing integers to represent note onset times, we have thereby chosen to use a precision of 1 ms for the resolution of both phase and beat period. However, by changing this to 2 ms or higher, the computation time can be reduced to a few seconds for a whole song without any significant degradation of the output.

3. PERFORMANCE ANALYSIS

The resulting tempo path is significantly more helpful when seeking to understand the global tempo changes in a performance than simply plotting the inter-beat intervals. We visualise the data using a standard tempo curve which plots the graph of tempo, or beat period, against the beat annotation index, i.e. plotting τ_i. Expressive timing information can be shown by placing a dot above or below this point (i, τ_i), whereby if the beat annotation occurs x ms after the location of the path point, then the dot is x ms above the tempo curve. In the figures below, for simplicity of presentation we have translated the beat period into the more commonly found representation as BPM. Whilst this thereby omits specific units for the expressive timing and phase shift information, we consider the benefits in understanding the tempo information to make this worthwhile.

3.1 The Beatles Dataset

An example of this can be seen in Figure 3. The input data was used was ground-truth annotations to the Beatles' song 'Taxman' from the album 'Revolver' [12]. The annotations were created in a semi-automatic manner, via the use of a beat tracking algorithm and then corrected by hand [4]. The fact that an algorithm was used does mean that some smoothing has taken place, however, our proposed decoding process still provides timing data that offers insightful information for musicologists.

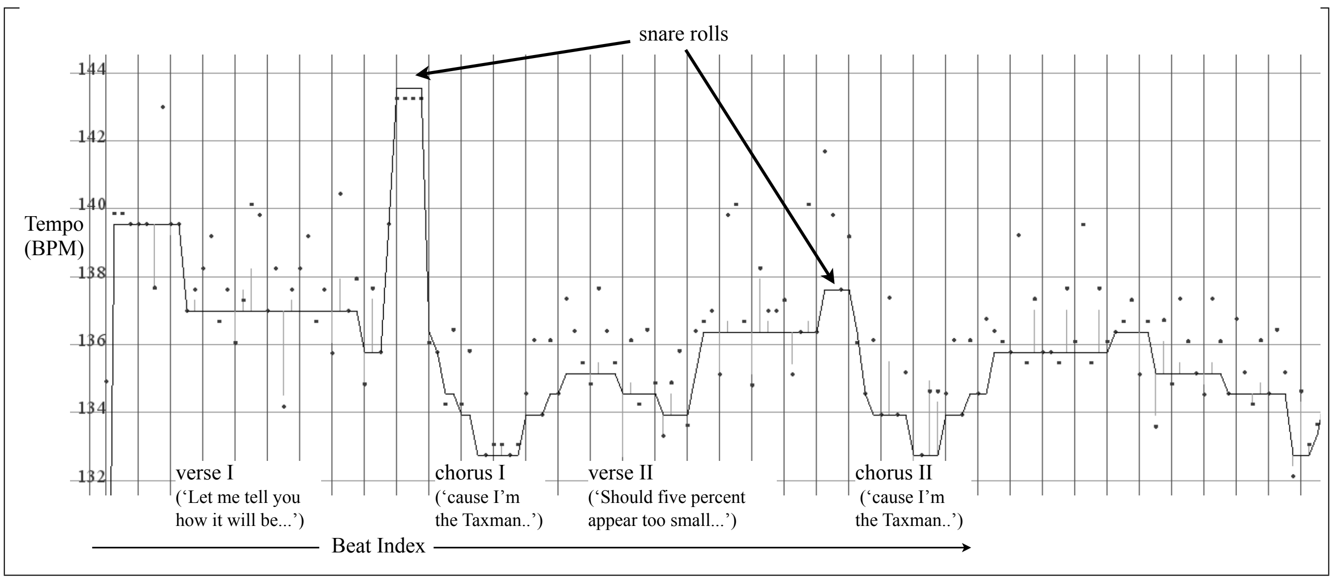


Figure 3. Tempo graph for The Beatles' 'Taxman'.

This analysis of the song indicates considerable complexity in Ringo Starr's time-keeping. He is both sensitive and in control of small fluctuations in tempo that generate a 'feel' to the different sections of the song. The decoded timing data displays clear small rises in tempo during the snare rolls that precede both the first and second choruses. There are then clear drops in tempo for both choruses of approximately 2 BPM, and this remains the case for later choruses beyond the scope of the Figure.

One can also observe a general trend in the expressive timing such that the timing of the second and fourth beats of the bar appears to be marginally later than the '1' and the '3'. On these beats, the song tends to feature the snare backbeat, as is common in rock music [10], and a regular guitar motif consisting of a staccato chord. Calculating the mean over the whole song confirms this, with the mean offsets being 0.25, 3.86, 0.82 and 2.18 ms for the respective beats. Drummers consider that placing the snare hit on '2' and '4' fractionally later, results in a more relaxed feel [11]. Such analysis supports the hypothesis that trends in microtime deviation lend a particular 'feel' to a song.

3.2 Beethoven's Moonlight Sonata

Chew [3] presents a detailed analysis of the timing variations in three different performances of Beethoven's Piano Sonata No. 14 in C Sharp Minor, Op. 27 No. 2: I. Adagio sostenuto, known as the 'Moonlight Sonata'. These recordings, by Daniel Barenboim [1], Artur Schnabel [2] and Maurizio Pollini [3], were initially used in an invited lecture by Jeanne Bamberger. This piece consists of repeated groups of four triplets in the right hand, with a movement between different chords. Such a repetitive structure suits it for revealing the tendencies of different performers with respect to their tempo and microtime variations. We have used of the same hand-annotated data, created using Sonic Visualiser.

In creating the tempo graphs, Chew comments on the necessity of smoothing to make the data understandable by the human eye, whilst warning that over-smoothing can result in important details being obscured. By use of the proposed method, we obtain smooth tempo graphs, but also preserve information about expressive timing and phase shifts.

The tempo graph for Pollini's performance is shown in Figure 4. Chew notes how the local minima of the tempo graph all occur on the bar boundaries. Bamberger contrasts this with the rendition by Schnabel, explaining that whereas other performers appear to 'stop' with each bass note at the beginning of the bar, Schnabel progresses through until the end of the first complete phrase after four bars "as if in one long breath". The extracted tempo and timing information for Schnabel's performance can be seen in Figure 5. Later in the piece, we can recognise a similar pattern to that exhibited by Pollini, whereby there is a slowing at the beginning of each bar.

This example can also serve to demonstrate some advantages of our proposed decoding method. The resulting tempo graph has less of the jagged edges that are still found in Chew's smoothed tempo graph. This is due to the projection of other timing data, expressive timing and phase shifts, onto the tempo curve. Instead, these quantities are made explicit and removed from the tempo curve, and thereby allowing us to calculate data relating to the phrasing of the notes. In this piece, we can observe that the third triplet eighth note exhibits a tendency to be marginally earlier than the first two notes of the bar. This would indicate that it thus begins earlier and is held fractionally longer. We have calculated the average deviation for each note and these results are presented in Table 1.

[1] On Beethoven: Moonlight, Pathtique and Appassionata Sonata CD Hamburg, Germany: Deutsche Grammophon GmbH.

[2] On Artur Schnabel CD United Kingdom: EMI Records Ltd.

[3] On Beethoven Piano Sonatas: Moonlight and Pastorale CD, Hamburg, Germany: Deutsche Grammophon GmbH.

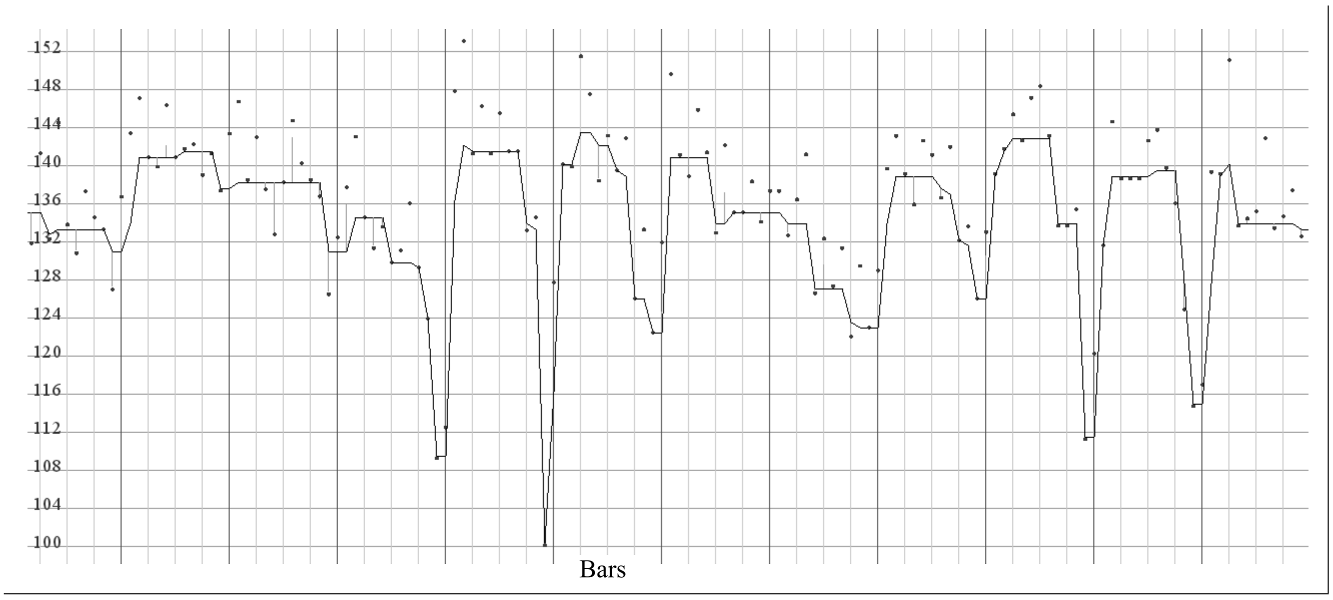


Figure 4. Tempo graph for Pollini's recording of Beethoven's Moonlight Sonata. The lighter vertical lines indicate crotchet boundaries and the darker vertical lines indicate bar boundaries. The expressive timing information is represented by the dots and phase shifts are represented by lines. Where the dot is above the tempo graph, the event is late relative to the time predicted by the underlying tempo; where the dot is below the event is early. Similarly a phase shift later is represented by a vertical line upwards from the tempo graph and a phase shift early by a line below the tempo graph. The tempo is indicated by BPM values to the left in 8 BPM intervals. The expressive timing and phase shift quantities are such that the

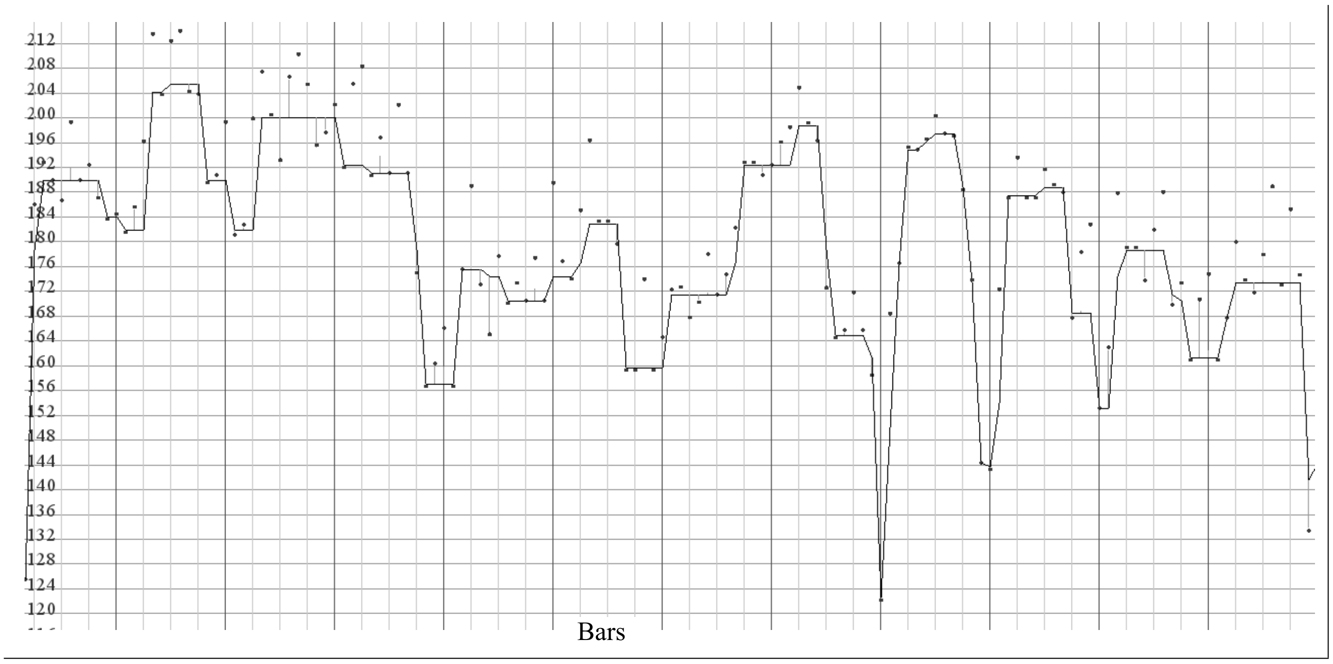


Figure 5. Tempo graph for Schnabel's recording of Beethoven's Moonlight Sonata. Again the lighter vertical lines indicate crotchet boundaries and the darker vertical lines indicate bar boundaries. the end of the first phrase is after four bars.

4. IMPLEMENTATION

The program was written in C++, using openFrameworks to provide visualisation using openGL libraries. The code is freely available for download at the Sound Software website[4], thereby allowing other researchers to import annotations. Both the resulting timing information and a file of the processed beat location times can then be exported as text files. Sonic Visualiser supports the loading of the processed annotations, which can then be sonified. In informal

[4] https://code.soundsoftware.ac.uk/projects/performance-timing-analyser

	Triplet note index		
Performer	1	2	3
Barenboim	7.27	2.56	0.83
Pollini	6.48	5.11	2.50
Schnabel	5.20	4.45	0.76

Table 1. Average deviation by triplet eighth note position in ms for the three performances.

tests, our processing algorithm appeared to have smoothed the data well, elimating timing errors whilst preserving the timing variations we are interested in.

5. CONCLUSION

In this paper, we present a new method for extracting the optimal tempo and timing path from a list of onset annotations. The output contains both tempo and expressive timing information for the optimal path according to our cost parameters. Such information enables a detailed musicological analysis of how performance timing data relates to musical structure. We have investigated how such data might be used in a classical case with the study of three performances of Beethoven's Moonlight Sonata, and in the rock and pop case through studying the timing of songs by The Beatles.

In future, we seek to extend our application of this method to the analysis of other annotated audio and develop a better understanding of how musicians make use of tempo and timing variations in expressive performance. We envisage that such work might also lead to improvements in the expressivity of computer-generated parts.

6. ACKNOWLEDGEMENTS

Thanks to Elaine Chew for making available the annotations for the Beethoven piano sonata recordings. Thanks to the EPSRC and the Royal Academy of Engineering for supporting this research.

7. REFERENCES

[1] C. Cannam, C. Landone, and M. Sandler. Sonic Visualiser: An open source application for viewing, analysing, and annotating music audio files. In *Proceedings of the ACM Multimedia 2010 International Conference, Firenze, Italy, October 2010.*, pages 1467–1468, 2010.

[2] Chris Cannam, Chris Landone, Mark B. Sandler, and J.P. Bello. The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR-06)*, 2006.

[3] Elaine Chew. About time: Strategies of performance revealed in graphs. *Visions of Research in Music Education*, 20, 2012.

[4] M. E. P. Davies, N. Degara, and M. D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. technical report c4dm-tr-09-06. Technical report, Queen Mary University of London, Centre for Digital Music., 2009.

[5] M. E. P. Davies and M. D. Plumbley. Context-dependent beat tracking of musical audio. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1009–1020, 2007.

[6] Peter Desain and Henkjan Honing. Tempo curves considered harmful: A critical review of the representation of timing in computer music. In *Proceedings of International Computer Music Conference*, pages 143–149, 1991.

[7] Simon Dixon, Werner Goebl, and Gerhard Widmer. The Performance Worm: Real time visualisation based on langner's represen- tation. In *Proceedings of the International Computer Music Conference*, 2002.

[8] Fabien Gouyon and Simon Dixon. A review of automatic rhythm description systems. *Computer Music Journal*, 29(1):34–54, 2005.

[9] Henkjan Honing. From time to time: The representation of timing and tempo. *Computer Music Journal*, 25(3):50–61, 2002.

[10] Tommy Igoe. In the Pocket. Essential Grooves. Part 2. Funk. *Modern Drummer*, July 2006.

[11] Vijay Iyer. *Microstructures of Feel, Macrostructures of Sound: Embodied Cognition in West African and African-American Musics.* PhD thesis, University of California, Berkeley, 1998.

[12] 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 Conference on Music Information Retrieval (ISMIR 2009)*, 2009.

[13] 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*, 36(1):1–16, 2007.

[14] Meinard Müller, Verena Konz, Andi Scharfstein, Sebastian Ewert, and Michael Clausen. Toward automated extraction of tempo parameters from expressive music recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Kobe, Japan.*, 2009.

[15] Bruno H. Repp. Patterns of expressive timing in performances of a beethoven minuet by nineteen famous pianists. *Psychology of Music*, 22:157–167, 1995.

[16] Barry Vercoe and Miller Puckette. Synthetic Rehearsal, training the Synthetic Performer. In *Proceedings of the International Computer Music Conference (ICMC 1985)*, pages 275–278, 1985.

UNSUPERVISED CHORD-SEQUENCE GENERATION FROM AN AUDIO EXAMPLE

Katerina Kosta [1,2], **Marco Marchini** [2], **Hendrik Purwins** [2,3]
[1] Centre for Digital Music, Queen Mary, University of London, Mile End Road, London E1 4NS, UK
[2] Music Technology Group, Universitat Pompeu Fabra, 08018 Barcelona, Spain
[3] Neurotechnology Group, Berlin Institute of Technology, 10587 Berlin, Germany
marco.marchini@upf.edu, katkost@gmail.com, hpurwins@gmail.com

ABSTRACT

A system is presented that generates a sound sequence from an original audio chord sequence, having the following characteristics: The generation can be arbitrarily long, preserves certain musical characteristics of the original and has a reasonable degree of interestingness. The procedure comprises the following steps: 1) chord segmentation by onset detection, 2) representation as Constant Q Profiles, 3) multi-level clustering, 4) cluster level selection, 5) metrical analysis, 6) building of a suffix tree, 7) generation heuristics. The system can be seen as a computational model of the cognition of harmony consisting of an unsupervised formation of harmonic categories (via multi-level clustering) and a sequence learning module (via suffix trees) which in turn controls the harmonic categorization in a top-down manner (via a measure of regularity). In the final synthesis, the system recombines the audio material derived from the sample itself and it is able to learn various harmonic styles. The system is applied to various musical styles and is then evaluated subjectively by musicians and non-musicians, showing that it is capable of producing sequences that maintain certain musical characteristics of the original.

1. INTRODUCTION

To what extent can a mathematical structure tell an emotional story? Can a system based on a probabilistic concept serve the purpose of composition? Iannis Xenakis discussed the role of causality in music in his book “Formalized Music, Thought and Mathematics in Composition”, where it is mentioned that a fertile transformation based on the emergence of statistical theories in physics played a crucial role in music construction and composition [20].

Statistical musical sequence generation dates back to Mozart’s “Musikalisches Würfelspiel” (1787) [8], and more recently to “The Continuator” by F. Pachet [14], D. Conklin’s work [3], the “Audio oracle” by S. Dubnov et al. [6] and the “Rhythm Continuator” by M. Marchini and H. Purwins (2010) [13]. The latter system [13] learns the structure of an audio recording of a rhythmical percussion fragment in an unsupervised manner and synthesizes musical variations from it. In the current paper this method is applied to chord sequences. It is related to work such as a harmonisation system described in [1] which, using Hidden Markov Models, it composes new harmonisations learned from a set of Bach chorals.

The results help to understand harmony as an emergent cognitive process and our system can be seen as a music cognition model of harmony. “*Expectation* plays an important role in various aspects of music cognition” [18]. In particular, this holds true for harmony.

2. CHORD GROUPING

Harmony is a unique feature distinguishing Western music from most other predominantly monophonic music traditions. Different theories account for the phenomenon of harmony, mapping chords e.g. to three main harmonic functions, seven scale degrees, or even finer subdivisions of chord groups, such as separating triads from seventh or ninth chords. The aim of this paper is to suggest an unsupervised model that lets such harmonic categories emerge from samples of a particular music style and model their statistical dependencies.

As Piston remarks in [15] (p. 31), “each scale degree has its part in the scheme of tonality, its tonal function”. Function theory by Riemann concerns the *meanings* of the chords which progressions link. The term “function” can be used in a stronger sense as well, for specifying a chord progression [10]. A problem arises from the fact that scale degrees cannot be mapped to the tonal functions in a unique way [4] [16] (p. 51-55). In our framework, the function of a chord emerges from its cluster and its statistical dependency on the other chord clusters.

It is considered that the tonic (I), dominant (V) and subdominant (IV) triads constitute the *tonal degrees* since “they are the mainstay of the tonality” and that the last two give an impression of “balanced support of the tonic” [15]. This hierarchy of harmonic stability has been supported by psychological studies as well. One approach involves collecting ratings of how one chord follows from another. As it is mentioned in [11], Krumhansl, Bharucha, and Kessler used such judgments to perform multidimensional scaling and hierarchical clustering techniques [9]. The psychological distances between chords reflected both key membership and stability within the key; "chords belonging to different keys grouped together with the most stable chords in each key (I, V, and IV) forming an even smaller cluster. Such rating methods also suggest that the harmonic stability of each chord in a pair affects its perceived relationship to the other, and this depends upon the stability of the second chord in particular" [9].

3. METHODOLOGY

The goal of this system is the analysis of a chord sequence given as audio input, with the aim of generating arbitrarily long, musically meaningful and interesting sound sequences maintaining the characteristics of the input sample.

From audio guitar and piano chord sequences, we detect onsets, key and tempo, and group the chords, applying agglomerative clustering. Then, Variable Length Markov Chains (VLMCs) are used as a sequence model. In Figure 1 the general architecture is presented.

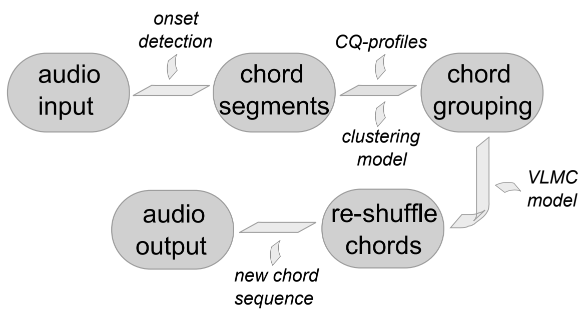


Figure 1. General system architecture.

3.1 Onset Detection

In order to segment the audio into a sequence of chords we employed an onset detection algorithm. Different approaches have been considered since a simplified onset detection method based only on the energy envelope would not be sufficient. After trying a bunch of available algorithms from the literature we found that the *complexdomain* from Aubio [21] was suited for our propose.

A crucial parameter of this algorithm is the *sensitivity* which required an ad hoc tuning. We selected a piano performance of Bach's choral "An Wasserflussen Babylon (Vergl. Nr. 209) in G major - from here on referred as "test -Bach choral" - as a ground truth test set for onset detection. Although with an optimal sensitivity we were still obtaining an incorrect merge of two consecutive segments in the 5.88% of the cases out of a total of 68 segments considered. In Figure 2, the first five segments that were obtained for the test-Bach choral are presented. An example of incorrect merge is shown on the 5th segment, the two consecutive chords of which get still gathered together, as their common notes are still resonating during the passing.

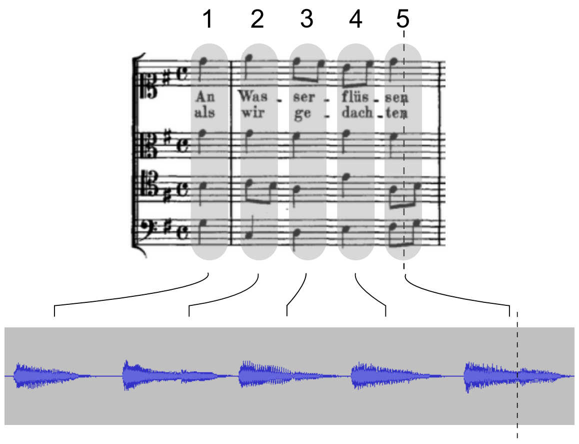


Figure 2. The first 5 segments of the test - Bach choral using Aubio [21] for onset detection. The fifth excerpt should be splitted into two parts -vertical black line- since two different kind of chords are identified and could be used separately.

3.2 Constant Q Profiles and Sound Clustering

From the audio input we extract chroma information based on Constant Q (CQ) profiles, which are 12 - dimensional vectors, each component referring to a pitch class. The idea is that every profile should reflect the tonal hierarchy that is characteristic for its key [2].

The calculation of the CQ profiles is based on the CQ transform; as decribed by Schorkhuber and Klapuri in [19], "it refers to a time-frequency representation where the frequency bins are geometrically spaced and the Q factors which are ratios of the center frequencies to bandwidths, of all bins are equal". This is the main difference between the CQ transform and Fourier transform. In our implementation we have used 36 bins per octave, the square root of a Blackman-Harris window and a hop size equal to 50% of the window size. The CQ profiles are closely related to the probe tone ratings by Krumhansl [17]. Also the system employs a method described by Dixon in [5] for tempo estimation.

In the clustering part, as each event is characterized by a 12-dimensional vector, they can thus be seen as points in a 12-dimensional space in which a metric is induced by the Euclidean distance. The single linkage algorithm has been used to discover event clusters in this space. As defined in [13], this algorithm recursively performs clustering in a bottom-up manner. Points are grouped into clusters. Then clusters are merged with additional points and clusters are merged with clusters into super clusters. The distance between two clusters is defined as the shortest distance between two points, each in a different cluster, yielding a binary tree representation of the point similarities. The leaf nodes correspond to single events. Each node of the tree occurs at a certain height - level, representing the distance between the two child-nodes (cf. [7] p. 517-557 for details).

Then the *regularity* concept described in [13] is computed for each sequence of each clustering level. Firstly, we compute the histogram of the time differences (CIOIH) between all possible combinations of two onsets. What we obtain is a sort of harmonic series of peaks that are more or less prominent according to the self-similarity of the sequence on different scales. Secondly, we compute the autocorrelation $\mathrm{ac}(t)$ (where t is the time in seconds) of the CIOIH which, in case of a regular sequence, has peaks at multiples of its tempo. Let t_{usp} be the positive time value corresponding to its upper side peak. Given the sequence of m onsets $x = (x_1, \ldots, x_m)$ we define the *regularity* of the sequence of onsets x to be:

$$\mathrm{Regularity}(x) = \frac{\mathrm{ac}(t_{usp})}{\frac{1}{t_{usp}} \int_0^{t_{usp}} \mathrm{ac}(t)dt} \log(m)$$

This regularity is then used to select the most regular level for tempo detection and a small amount of representative levels for the VLMC generation.

In Figure 3, there is a tree representation of the clustering results for the audio test - Bach choral. The system has selected 10 clustering levels, and the cluster hierarchy for the levels 1 - 6 is presented. We have only considered the clusters with more than one element.

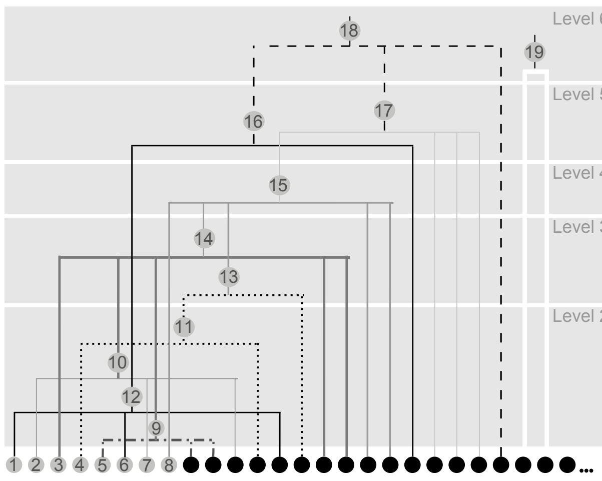


Figure 3. Base line: the clusters generated at Level 1 as circles; the black ones contain one single element.

In Table 1, the clustering results on levels 1 - 4 of the analyzed Bach choral are shown in more detail. It is noticable that we get a *rich* group, containing a large amount of G Major dominant chords.

3.3 Statistical Model for Sequence Generating

Having the segments of the input sound categorized properly, the next step is to re-generate them in a different order than the original one, taking into account that they are not independent and identically distributed, but dependent on the previous segments. For implementing this idea it would be impractical to consider a general dependence of future observations on all previous observations because the complexity of such a model would grow without limit as the number of observations increases. This leads us to consider Markov models in which we assume that future predictions are independent of all but the most recent observations.

A VLMC of order p is a Markov chain of order p, with the additional attractive structure that its memory depends on a variable number of lagged values [12]. This can be evaluated on our system as follows; Let's assume that we have, as an input, two sequences of events - elements of a categorical space having length $\ell = 4$. Be (A,B,C,A) and (B,C,C,D), which are parsed from right to left. As seen in [14], context trees are created where a list of continuations encountered in the corpus are attached to each tree node.

Cluster	# of Elements	Recognition
Level 1:		
cl. 1	3	2 G I, 1 G V
cl. 2	3	1 G I, 1 a V, 1 d IV
cl. 3	2	2 G IV
cl. 4	2	1 G I, 1 G V
cl. 5	10	5 G V, 1 a I, 1 d I, 1 d VI, 1 d V, 1 G I
cl. 6	2	1 G IV, 1 a I
cl. 7	4	1 G II, 1 a V, 1 a I, 1 d V
cl. 8	2	2 G V
Level 2:		
cl. 9	(cl.5)+2	6 G V, 1 a I, 2 d I, 1 d VI, 1 d V, 1 G I
cl. 10	(cl.2+cl.7)+1	1 G I, 2 a V, 1 d IV, 1 G II, 1 a I, 1 d V
cl. 11	(cl.4)+1	2 G I, 1 G V
cl. 12	(cl.1+cl.6)+1	2 G I, 1 G V, 1 G IV, 1 a I
Level 3:		
cl. 13	(cl.11)+1	3 G I, 1 G V
cl. 14	(cl.3+cl.9+cl.10) +2	2 G I, 1 G II, 2 G IV, 6 G V, 2 a I, 2 a V, 2 d I, 1 d IV, 2 d V, 1 d VI
Level 4:		
cl. 15	(cl.8+cl.13+cl.14) + 2	9 G V, 5 G I, 1 G II, 2 G IV, 2 a I, 2 a V, 3 d I, 1 d IV, 2 d V, 2 d VI

Table 1. the clustering results on levels 1 - 4 of the analyzed Bach choral. At the first column, we define each cluster by a number and at the second column we present the number of elements inside that cluster. At the third column we *recognize* these elements and label them based on our score' s harmonic analysis for each one separately (for example: "2 G I" means "2 of the elements are the root of G major" and "5 a V" means "5 of the elements are the dominant of A minor").

The "continuations" are integer numbers which denote the index of continuation item in the input sequence. In Figure 4, the procedure of the context tree creation based on sequences (A,B,C,A) and (B,C,C,D) is shown, where the index numbers show with which element one can proceed.

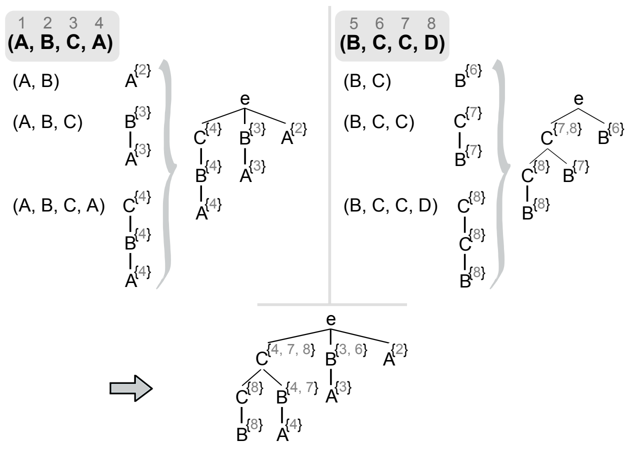


Figure 4. Top left and right: Context trees built from the analysis of the sequences (A B C A) and (B C C D) respectively. Bottom: Merge of the context trees above.

Exploring the final graph in Figure 4, where the trees above are merged, we have all the possible sequence situations, following each path that is created from bottom to up and considering the index number of the first element. For example, if we want to find which is the next element of the sequence (A,B,C), we follow this specific path from the bottom of the tree and then we see the index number of the first element, A, so we take the element with this index number, which is A and the sequence now becomes (A,B,C,A). For "e" (the empty context) we consider a random selection of any event. Also the length ℓ can be variable.

For the generation we use the suffix trees for all previously selected levels. If we fix a particular level, the continuation indices are drawn according to a posterior probability distribution determined by the longest context found. Depending on the sequence, it could be better to do predictions based either on a coarse or a fine level. In order to increase recombination of blocks and still provide good continuation we employ the heuristics detailed in Section 3.1. in [13] taking into account multiple levels for the prediction.

4. EVALUATION

Five audio inputs have been selected to evaluate the method: a guitar chord sequence based on the song "If I fell in love with you" by the Beatles, a Bach choral played on the piano, part of the "Funeral March" by Chopin, a guitar flamenco excerpt and a piano chord sequence by a non-musician (Examples No.1-5).

The next step was to create generations, using these five different piano and guitar audio inputs followed each one by generations of one minute duration. All the audio examples, some meta data, as well as the generations, and the results of the evaluation are available on the web site [22]. There are two carefully selected generations presented per piece, except for Example No. 5, where there is only one. The following characteristics of the system are assessed: the selected clustering level, the similarity between the input sample and the generation, and how many times an event is followed by another event in the generation that is not the event's successor in the original (i.e. how many "jumps" the generated sound contains).

Since the opinion of a musician rather than an objective measure is a more suitable evaluation measure for the aesthetic value of a generated music sample, a questionnaire for each input and its generations was created and given to five musicians [1] and five non-musicians at ages between 22 and 28. They had to listen to and rate each audio (from 1- "not at all" to 5- "very much") for their familiarity with the piece and the interestingness of the piece. In addition, the subject had to select the most interesting 10-second parts of it and they had to determine a similarity value comparing two audio examples. Original and the generations were presented without indicating which was which. For Examples 2 and 3 (Bach and Chopin) another question was added, asking to rate how clear the structure of the piece is.

Through the results of this experiment (details in Table 2), we can highlight that only 3% of the responses found the generation example as *not similar* to the original input. Also through the Examples 1, 4 and 5 we notice that 20% of the responses found the generation example more *interesting* than the original and 26% of the responses found the generation example less *interesting*, although the range from the rate of the original one is not big.

In general the cumulative results for the similarity module show small differences between musician's and non-musician's replies. Another measure of comparison between these groups is their response concerning the 10 most interesting seconds; ten groups of overlapping seconds have emerged and seven of these groups were indicated by both musicians and non-musicians.

The comments made by the subjects gave us additional insight into the behaviour of the system. Metrical phase errors have been spotted in the generations of Example No. 4, resulting in rhythmic pattern discontinuities. Some of the musician subjects considered these sections as *"confusing"* and some others as *"intriguing expertise"*. Another important issue is the quality of the generation, in terms of its harmonic structure. A representative comment on Example No.5 is: *"In the second audio (i.e. the Original) I could hear more harmonically false sequences"*.

5. DISCUSSION AND CONCLUSION

The system generates harmonic chord sequences from a given example, combining machine learning and signal processing techniques. As the questionnaire results highlight, the generation is similar to the original sample, maintain-

[1] They are defined as individuals, having at least five years of music theory studies and instrument playing experience.

ing key features of the latter, with a relatively high degree of interestingness.

An important extension of this work would incorporate and learn structural constraints as closing formulae and musical form. Other future work comprises an in-depth comparison of the chord taxonomies generated by the system and taxonomies suggested by various music theorists, e.g. Riemann, Rameau, or the theory of jazz harmony and possibly the experimental verification of such harmonic categories in the brain, e.g. in an EEG experiment.

However, for an automatic music generation system, there remains still a long way to go in order to comply with the idea of music as Jani Christou puts it: ”The function of music is to create soul, by creating conditions for myth, the root of all soul”.

6. ACKNOWLEDGEMENT

The work of the second author (M. M.) was supported in part by the European project ”Social Interaction and Entrainment using Music PeRformance Experimentation” (SIEMPRE, Ref. No. 250026). The work of the third author (H. P.) was supported in part by the German Bundesministerium für Forschung und Technologie (BMBF), Grant No. Fkz $01GQ0850$.

7. REFERENCES

[1] Moray Allan and Christopher K. I. Williams, Harmonising Chorales by Probabilistic Inference, Advances in Neural Information Processing Systems 17, 2005.

[2] J. C. Brown, M. S. Puckette. An efficient algorithm for the calculation of a constant Q transform.,J. Acoust. Soc. Am., 92(5):2698-2701, 1992.

[3] D. Conklin, Music Generation from Statistical Models, Proceedings AISB, p. 30-35, 2003.

[4] C. Dahlhaus, Untersuchungen ber die Entstehung der harmonischen Tonalitt, volume 2 of Saarbrcker Studien zur Musikwissenschaft. Brenreiter-Verlag, Kassel, 1967

[5] S. Dixon, Automatic extraction of tempo and beat from expressive performances, Journal of New Music Research, 30(1):39-58, 2001.

[6] S. Dubnov, G. Assayag, A. Cont, Audio Oracle: A New Algorithm for Fast Learning of Audio Structures, in Proceedings of ICMC, 2007.

[7] R. O. Duda, Peter E. Hartl, D. G. Stork, Pattern Classification (2nd edition), 2001.

[8] K. Jones, Dicing with Mozart: Just a whimsical musicians in the 18th, NewScientist, Physics & Math, 1991.

[9] T. Justus, J. Bharucha, Stevens' Handbook of Experimental Psychology, Volume 1: Sensation and Perception, Third Edition, pp. 453-492, New York: Wiley, 2002.

[10] D. Kopp, On the Function of Function, Music Theory Online, Society for Music Theory, Volume 1, Number 3, May 1995, ISSN: 1067-3040, 1995.

[11] Krumhansl, C.L., Bharucha, J.J., Kessler, E.J. Perceived harmonic structure of chords in three related musical keys. Journal of Experimental Psychology: Human Perception and Performance, vol. 8, pp. 24-36, 1982.

[12] M. Machler, P. Buhlmann, Variable Length Markov Chains: Methodology, Computing and Software, ETH, Research Report No. 104, March 2002.

[13] M. Marchini, H. Purwins, Unsupervised Generation of Percussion Sound Sequences from a Sound Example, MSc thesis, UPF, 2010.

[14] F. Pachet, The continuator: Musical interaction with style, in Proceedings of ICMC (ICMA ed.), pp. 211-218, September 2002.

[15] W. Piston, Harmony, Victor Gollancz Ltd, London, 1959.

[16] H. Purwins, Profiles of Pitch Classes Circularity of Relative Pitch and Key - Experiments, Models, Computational Music Analysis, and Perspectives, Ph.D. Thesis, Berlin Institute of Technology, 2005.

[17] H. Purwins, B. Blankertz, K. Obermayer, A New Method for Tracking Modulations in Tonal Music in Audio Data Format, Proceedings of the IJCNN. vol.6, pp. 270-275, 2000

[18] H. Purwins, M. Grachten, P. Herrera, A. Hazan, R. Marxer, and X. Serra, Computational models of music perception and cognition II: Domain-specic music processing, Physics of Life Reviews, vol. 5, pp. 169-182, 2008.

[19] C. Schorkhuber, A. Klapuri, Constant-Q transform toolbox for music processing, in: 7th Sound and Music Computing Conference, July 2010.

[20] Iannis Xenakis, Formalized Music: Thought and Mathematics in Composition, Bloomington: Indiana University Press, 1971.

[21] http://aubio.org, April 2012.

[22] http://soundcloud.com/chordsequencegenerator , April 2012.

Example 1	Musicians		Non-musicians	
	Familiarity	Interesting	Familiarity	Interesting
Original	2,1,3,1,4	2,2,4 (22-30s),4 (11-16s),3	2,2,3,2,4	3,4(38-42s),4 (22-32s),2,3
Generation 1	2,1,3,1,3	2,2,3,5 (4-12s),2	2,3,2,2,2	4 (1-11s),3,3,2,3
Generation 2	5,1,3,1,2	2,2,3,2,3	3,3,4,2,4	2,5 (48-58s),4 (40-50s),2, 4 (45-55s)

Similarity	Org.-Gen.1	Org.-Gen.2	Org.-Gen.1	Org.-Gen.2
Not similar				
Somewhat similar	++	++	++	+++
Very similar	+++	+++	+++	++

Example 2	Musicians		Non-musicians	
	Familiarity		Familiarity	
Original	4,4,4,5,4		3,3,4,2,5	
	Clearness	Interesting	Clearness	Interesting
Generation 1	4,5,5,3,2	3,5 (30-40s),4 (30-40s),2,1	4,5,4,4,4	4 (1-11s),5 (1-11s), 4 (45-55s),4 (30-40s),3
Generation 2	5,4,3,2,3	1,4 (23-32s),3,3,2	4,4,3,3,3	2,3,4,2,4

Similarity	Org.-Gen.1	Org.-Gen.2	Org.-Gen.1	Org.-Gen.2
Not similar		+	+	
Somewhat similar	+	+	++	+++
Very similar	++++	+++	++	++

Example 3	Musicians		Non-musicians	
	Familiarity		Familiarity	
Original	5,5,4,5,5		4,5,5,5,5	
	Clearness	Interesting	Clearness	Interesting
Generation 1	5,5,5,3,2	5 (0-10s),5 (43-53s),3,3,1	5,5,3,4,3	5 (33-43s),5 (43-48s),3,3 ,5 (30-40s)
Generation 2	5,4,4,2,4	5 (34-44s),4 (43-51s),4,3,2	4,5,3,5,4	5 (17-24s),5 (34-44s),4 (45-52s), 4 (20-30s),4 (40-50s)

Similarity	Org.-Gen.1	Org.-Gen.2	Org.-Gen.1	Org.-Gen.2
Not similar				
Somewhat similar	++	++++		++
Very similar	+++	+	+++++	+++

Example 4	Musicians		Non-musicians	
	Familiarity	Interesting	Familiarity	Interesting
Original	1,2,1,5,2	4 (0-10s),2,4 (34-38s), 4 (28-38s),4 (10-20s)	3,2,3,2,4	4 (1-8s),3,3,1,3
Generation 1	1,2,1,5,2	4 (0-10s),2,3,4 (8-14s), 4 (9-13s)	4,1,4,2,5	3,3,3,1,4 (10-20s)
Generation 2	1,2,1,5,2	1,2,5 (7-15s),3,3	2,1,3,2,5	3,3 (32-42s),4 (45-55s),1,3

Similarity	Org.-Gen.1	Org.-Gen.2	Org.-Gen.1	Org.-Gen.2
Not similar				
Somewhat similar		+++	+	++++
Very similar	+++++	++	++++	+

Example 5	Musicians		Non-musicians	
	Familiarity	Interesting	Familiarity	Interesting
Original	1,2,1,3,4	1,2,2,2,4 (20-30s)	1,1,3,3,3	2,2,2,2,3
Generation	1,2,1,4,4	1,2,3,3,4 (11-16s)	1,1,2,3,2	2,2,3,2,3

Similarity	Org.-Gen.		Org.-Gen.	
Not similar			+	
Somewhat similar	+++++		+++	
Very similar			+	

Table 2. We present the questionnaire responses for Examples 1 - 5; the ratings (from 1 to 5) that both musicians and non musicians have given for each audio thus the rate for similarity comparing specific audio couples are shown. At the *interesting* part, there is a potential mention of the most interesting 10 seconds, in case the response in that section was 4 or 5.

LISTENING LEVEL CHANGES MUSIC SIMILARITY

Michael J. Terrell György Fazekas Andrew J. R. Simpson Jordan Smith Simon Dixon
Centre for Digital Music, Queen Mary University of London
Mile End Road, London, E1 4NS, UK
michael.terrell@eecs.qmul.ac.uk, gyorgy.fazekas@eecs.qmul.ac.uk
andy.simpson@eecs.qmul.ac.uk, jordan.smith@eecs.qmul.ac.uk
simon.dixon@eecs.qmul.ac.uk

ABSTRACT

We examine the effect of listening level, i.e. the absolute sound pressure level at which sounds are reproduced, on music similarity, and in particular, on playlist generation. Current methods commonly use similarity metrics based on Mel-frequency cepstral coefficients (MFCCs), which are derived from the objective frequency spectrum of a sound. We follow this approach, but use the level-dependent auditory spectrum, evaluated using the loudness models of Glasberg and Moore, at three listening levels, to produce auditory spectrum cepstral coefficients (ASCCs). The ASCCs are used to generate sets of playlists at each listening level, using a typical method, and these playlists were found to differ greatly. From this we conclude that music recommendation systems could be made more perceptually relevant if listening level information were included. We discuss the findings in relation to other fields within MIR where inclusion of listening level might also be of benefit.

1. INTRODUCTION

The auditory system can be thought of, in signal processing terms, as a level-dependent filter bank, where each component is known as an auditory filter [15]. Incoming sound is first processed by the frequency and direction dependent filter of the pinna (outer ear), before passing through the ear canal, which acts as a narrowband resonant amplifier. The acoustic pressure at the ear-drum is mechanically transmitted, via the amplifying stage of the middle-ear ossicles, to the fluid of the cochlea (inner ear) via the oval window [19]. Due to continuous variation in mass and stiffness along the basilar membrane, the cochlea provides a tonotopic representation (arranged in order of frequency) of sound energy spectrum that is broadly consistent with Fourier analysis.

Within the cochlea, inner hair cells are tonotopically arranged along the basilar membrane. The inner hair cells are innervated with neurons that provide the firing-rate coded signal that is sent to the brain via the auditory nerve. The inner hair cells are accompanied by respective outer hair cells. Pressure gradients in the cochlear fluid cause the inner hair cells at any given location to be deflected in a shearing motion which results from place-frequency dependent resonance of the basilar membrane. At the same time, the motile outer hair cells act in phase-locked synchrony to amplify the excitation. This system is known as the cochlear amplifier.

Each inner hair cell is innervated with a population of neurons that code the local signal in terms of the rate-level function (the function that relates the rate of neuron firing to the perceived intensity level). The stochastic firing rate-level function of a neuron, or a population of neurons, can be thought of as having three distinct stages: spontaneous firing, threshold, and saturation. Below threshold, the neuron fires randomly at a low rate. Between threshold and saturation, the function is close to linear and provides a good coding of level. Above saturation point, increase in level does not result in a proportional increase in firing rate. Thus, with increase in sound pressure level, an increasing area of inner hair cells on the basilar membrane are excited beyond neural threshold. Within the context of the excitation pattern model described above, this is known as spread of excitation.

The action of the cochlear amplifier gives rise to strongly level-dependent tuning of the auditory filter. At low levels, the phase-locked action of outer hair cells provides tonotopically localised amplification, which results in a narrow auditory filter. At high sound pressure levels, the cochlear amplifier is not able to contribute amplification, due to mechanical limits, and so the auditory filter becomes broader with increase in level.

The parameters of the human auditory filter have been determined using psychophysical methods [17] and are represented in terms of equivalent rectangular bandwidth (ERB). Within the music information retrieval (MIR) community, the auditory filters are typically more broadly represented in terms of the approximately analogous Mel frequency scale [21]. The Mel scale is defined in terms of equal pitch distance. Both scales produce a "non-linear mapping" of the frequency domain.

Mel-frequency cepstrum coefficients (MFCC), derived using the discrete cosine transform, have been used for speech recognition [9], music modelling [13] and music similarity [12]. The ERB scale has been used to improve speech feature extraction [20]. Other related work [9] used a gammatone auditory filter-bank [8] (derived from non-human physiology) in the place of ERBs. The resulting coefficients were referred to as EFCCs.

Thus far, although the MFCCs and EFCCs applied to MIR problems have made some attempt to address the question of perception in terms of frequency warping, no attempt has been made to demonstrate the major level-dependent effects of cochlear processing: (i) absolute threshold, (ii) spread of excitation, (iii) compression, and (iv) masking. In other words, the major parameter of listening level has not been investigated.

At present, MIR is usually based on recordings, which listeners can reproduce at any listening level. Whilst we acknowledge the immediate practical difficulty that this imposes, we believe it is important to determine whether the effects of listening level may be significant. In this article, we use a psychoacoustic model to produce level dependent spectrograms, which incorporate the effects of (i) absolute threshold, (ii) spread of excitation and (iii) compression, and which can be used to evaluate level dependent similarity metrics. The similarity ratings are compared for each listening level to determine whether specific applications of MIR, such as music playlist generation, may be listening level dependent. This article also serves to begin a more general discussion as to the relevance and importance of listening level for other areas within MIR.

2. MODELLING

The loudness models [7, 16] provide a means to predict time and level dependent excitation patterns for time-varying acoustic stimuli. The outer and middle ear stages are modelled as a single FIR filter. Next, a bank of parallel filters is used to calculate spectral magnitude over specific frequency bands. The resulting excitation pattern is then transformed into instantaneous specific loudness (ISL) according to a compressive nonlinearity designed to model the action of the cochlea. The instantaneous specific loudness is essentially a level dependent spectrogram with the frequency axis in the ERB scale. We refer to it as an auditory spectrogram.

We collected a random subset of 500 recordings from the Magnatagatune data set [10]. Magnatagatune is a collection of over 56,000, 30-second music clips from the Magnatune catalogue, with matching tags collected from Law's TagATune game. Our subset of 500 clips has approximately the same proportion of genres as the full data set, including roughly 22% Classical, 17% each of Pop/Rock, Electronic and "Ethnic" or World music, and the rest from assorted genres. The clips, all 44.1kHz, 32kbps mono mp3 files, were obtained using the "Source Only" version of the Magnatagatune data set.

Using the auditory model, auditory spectrograms were estimated at three listening levels for each recording. A 20 ms normalised Hanning window was used with a 50% overlap. The frequency axis was split into ERB bands, which gave 53 discrete frequency bins. The listening levels were characterised by peak sound pressure levels of 40, 80 and 120 dB SPL. The input to the loudness model is a waveform in Pascals (Pa), where a pressure of 1 Pa corresponds to 94 dB SPL. Therefore, in order to convert a normalised digital recording (peak amplitude is 1), s_d, into a pressure signal s_p with a peak level of X SPL, we use,

$$s_p = 10^{\frac{(X-94)}{20}} s_d. \tag{1}$$

Figure 1 shows the auditory spectrograms for a randomly selected recording played at each listening level. At 40 dB SPL it becomes relatively narrow-band due to the high and low frequency energy falling below the absolute thresholds of audibility. At 80 dB the majority of the energy is above absolute threshold and the auditory spectrogram is similar to the objective spectrogram. At 120 dB SPL the spread of excitation causes smearing of the energy across the frequency range, and the recording becomes relatively broadband.

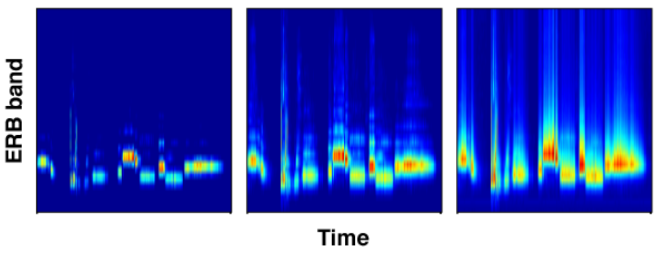


Figure 1. The auditory spectrograms of a randomly selected recodring with peak play-back intensity levels from left to right of: 40, 80 and 120 dB SPL respectively.

3. ANALYSIS

An acoustic model of musical timbre is often a core component of content-based MIR systems. It is fundamental in tasks such as content-based music recommendation [13], playlist generation [18], genre classification [23] and instrument recognition [5]. In our experiments, we choose to follow a deliberately simple, yet widely adopted method of modelling the overall timbre of a recording first by extracting frame-wise cepstral coefficients, and then modelling the overall timbre distribution by fitting a single Gaussian to the resulting coefficient vectors [13]. In order to be able to take the effect of listening level into account, we use a set of auditory spectra cepstral coefficients (termed AS-CCs), computed from auditory spectra, calculated using the method outlined in Section 2.

Similarly to MFCCs, the computation of this feature is derived from the computation of the real Cepstrum shown in Equation 2, where $X(\omega)$ represents the Fourier transform of the analysed signal. The cepstrum separates the slowly varying components of a signal from superimposed higher frequency and noise like components. It can be viewed as a rearranged spectrum, such that relatively few coefficients are sufficient to characterise the spectral envelope; however, the higher the number of coefficients, the more spectral detail is retained.

$$c(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \log |X(\omega)| \, e^{j\omega n} d\omega \tag{2}$$

In many applications, including speech recognition and audio similarity analysis, it has become common to characterise short audio segments using a set of cepstral coefficients, such that non-linear frequency warping is used to emphasise perceptually relevant frequencies corresponding to auditory bands. Mel-scaling is the most widely adopted method for this purpose.

Our feature extraction follows a common procedure of computing MFCCs [4]; however instead of using Mel-scaled magnitude spectra, we use auditory spectra estimated at three different listening levels. The auditory spectrograms are logarithmically compressed and then decorrelated using the Discrete Cosine Transform (DCT) given in Equation 3.

$$C(n) = \sum_{i=1}^{M} X(i) \cos \frac{n(i-0.5)\pi}{M}, \text{with } n = 1, 2..., J, \tag{3}$$

where M is the number of auditory filters, J is the number of ASCCs (typically $J < M$), and $X(i)$ is the log-magnitude output of the i-th filter. These coefficients are then modelled using a single Gaussian characterising the distribution of ASCCs over a song in our collection.

This method makes several simplifying assumptions. For one, it ignores musical structure, and also the fact that the distribution of timbre features is not necessarily Gaussian. A solution to these problems may be the use of Gaussian mixture models (GMM) or a sequence of Gaussians fitted on coherent segments, for instance, a single Gaussian representing each bar or each structural segment of the music, for modelling a track. However, approaches to estimate similarity between these models such as Monte Carlo sampling are computationally expensive. Detailed discussions on timbre models and the effects of the above assumptions can be found, for instance, in [1], [2] and [3]. Besides modelling recordings using a single Gaussian, a further simplifying assumption is introduced by using Gaussians with diagonal covariance. Although modelling timbre using a single Gaussian is a very simple approach, it was shown in [14] that it can perform comparably to mixture models when computing similarity between recorded audio tracks. It was also shown to be effective and computationally efficient for finding similar songs in personal music collections in [11]. An important advantage of using this model is that the similarity between two tracks can be estimated using closed form expressions, such as the Jensen-Shannon (JS) or Kullback-Leibler (KL) divergences. Here, we use the symmetrised KL divergence given in Equation 4, where p and q are Gaussian distributions, with μ mean and Σ covariance, and d is the dimensionality of the feature vectors.

$$\begin{aligned} KL_s(p\|q) &= 2KL(p\|q) + 2KL(q\|p) \\ &= tr(\Sigma_q^{-1}\Sigma_p + \Sigma_p^{-1}\Sigma_q) \\ &\quad +(\mu_p - \mu_q)^T(\Sigma_q^{-1} + \Sigma_p^{-1})(\mu_p - \mu_q) \\ &\quad -2d \end{aligned} \tag{4}$$

Using this simple model, we calculate symmetric distance matrices holding pair-wise KL-divergences (similarity estimates) between all recordings in our collection. For each distance matrix computation, different sets of ASCCs are used that are calculated from the auditory spectra estimated for different listening levels. The distance matrices are then individually analysed using the methods described in Section 4.1 and 4.2, and the results produced at three different levels are compared.

4. RESULTS

The data set is analysed as per Section 3 to produce a KL divergence rating per pair of recordings at each listening level. To illustrate the approach, 25 tracks from the set (n=500) were selected at random and KL divergence matrices computed at each listening level (40, 80, 120 dB SPL). Figure 2 shows the matrices. Blue indicates low values (similar) and red indicates high values (dissimilar). Figure 3 shows a box-plot of the matrix data. Figs. 2 and 3 clearly illustrate that the similarity ratings are strongly dependent on the listening level. At low level, the set shows a high mean similarity with relatively small variance. At high level the mean similarity is lower and the variance is larger. At the medium level the variance lies between the low and high listening levels.

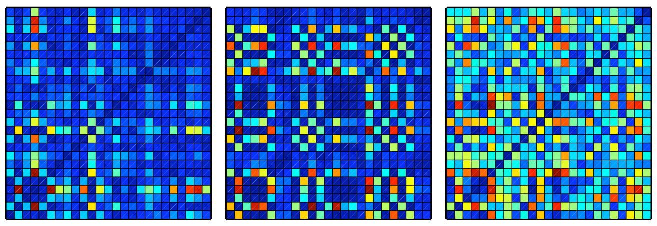


Figure 2. The normalised KL divergence matrices for a subset of recordings with peak intensity levels from left to right of: 40, 80 and 120 dB SPL respectively. Blue indicates low values (similar) and red indicates high values (dissimilar).

Whilst Fig. 2 shows that the similarity ratings are dependent upon the listening level, it is important to determine whether these differences are significant in MIR applications. The application we choose to study is music recommendation. Music recommendation tools generate playlists based on similarity ratings, typically derived from MFCCs. We compared the similarity data across the three intensity levels in two ways: (i) by comparing the ordering of distances within triples, and (ii) by comparing the members of playlists with different seed recordings, and with different playlist sizes.

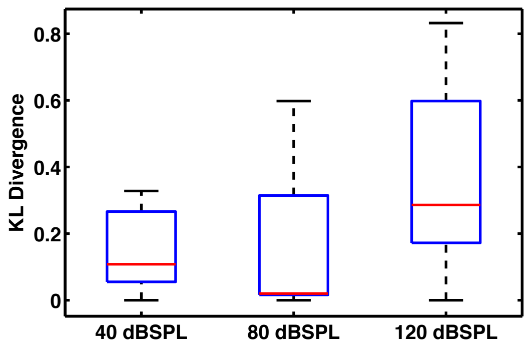


Figure 3. Boxplots of the KL divergence matrices (Fig. 2) at each listening level. Low values correspond to similar recordings and high values to dissimilar recordings.

4.1 Triple analysis

We analysed all subsets of 3 recordings from the dataset and the pair of recordings with minimum distance (in terms of the KL divergence feature space), was identified. The data was compared across listening level, and changes in the closest identified pairs were recorded. For example, if a given triplet (I,J,K) showed that at 40 dB SPL recordings I and J were closest together, but that at 80 dB SPL I and K were closest together, this was recorded as a change. The percentage changes were calculated across all triples and are shown in Table 1. We see around a 30% change in the ordering of triples. This suggests that MIR applications that use similarity metrics, such as playlist generation, will be affected by listening level.

% Change in Triplet Order		
40 vs 80	40 vs 120	80 vs 120
32	29	27

Table 1. The percentage change in the closest identified pair within each set of triples. The column headers refer to the listening levels between which the comparisons were made, i.e. 40 vs 80 relates to comparison of triplet data from the 40 dB SPL and 80 dB SPL sets.

4.2 Playlist generation

Playlists were generated by assigning a seed song, and then identifying the $(n-1)$ closest songs in the similarity space, where n is the size of the playlist. The playlists were compared across listening levels. For example, if a five song playlist is generated for seed song A, where identified songs are (T,U,S,X) at 40 dB SPL, but at 80 dB SPL are (W,T,U,S), the percentage change would be 25%. We do not consider a playlist to have changed if the order of the chosen songs is different.

The mean and 95% confidence intervals are calculated for playlist changes across all seed songs. The mean data are shown in Table 2 using the first 20 ASCCs. Playlist change data using first 12, 20 and 29 ASCCs are plotted in Figure 4. The changes range from 80% for small playlists, to 50% for large playlists.

In order to verify the significance of these changes, an equivalent process is followed but comparisons are made between playlists generated using different numbers of ASCCs at each listening level. These data are shown in Figure 5. The changes range from 50% for small playlists, to 10% for large playlists. For a 10 song playlists, the average change (in the songs added) is: 62% caused by listening level (Fig. 4), and 22% caused by the number of ASCCs used (Fig. 5).

N. Songs	Mean % Change in Playlist Members		
	40 vs 80	40 vs 120	80 vs 120
1	74	67	80
2	69	64	78
3	68	62	76
4	66	59	75
5	66	58	75
6	65	57	74
7	64	56	73
8	63	55	73
9	62	54	72
10	61	53	71
11	61	52	70
12	60	52	70
13	60	51	69
14	59	51	68
15	58	50	68
16	58	49	67
17	58	49	67
18	57	48	66
19	57	47	66
20	57	47	65
21	57	46	65
22	56	46	65
23	56	46	64
24	55	45	64

Table 2. The percentage change in the recommended playlists using the first 20 ASCCs. The column headers refer to: the length of playlist (excluding seed song), $(n-1)$, and the listening levels between which the comparisons were made, i.e. 40 vs 80 relates to comparison of playlists from the 40 dB SPL and 80 dB SPL sets.

5. DISCUSSION

We have demonstrated that the effect of listening level is larger than that of variation of the number of ASCCs used in the playlist generation. The large percentage change in playlist members shown for the comparison between 40-80 dB SPL is perhaps most relevant to the typical MIR end user - such variation in listening levels may be typical in the home (e.g., for radio broadcast). The equally large percentage change shown in the results for the highest sound pressure level (120 dB SPL) may be relevant for the live

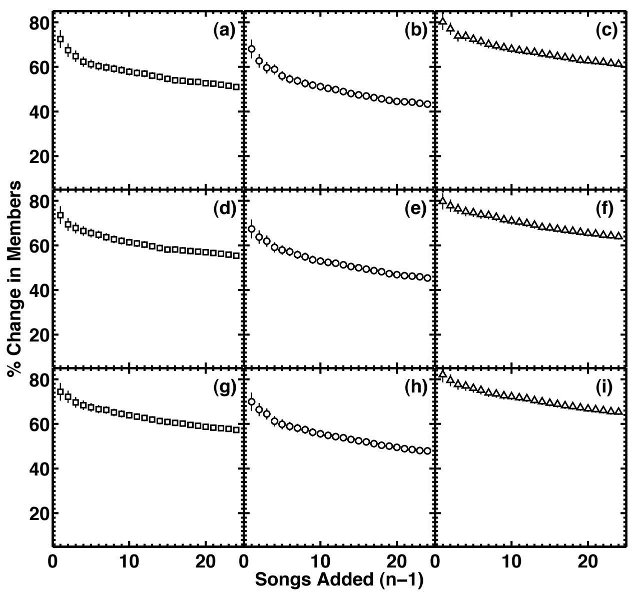


Figure 4. The percentage change in playlist members with listening level as a function of the length of playlist (excluding seed song). The data shown are the mean and 95% confidence intervals across all seed songs. The square, circle and triangle markers show comparisons between: 40 to 80, 40 to 120 and 80 to 120 dB SPL respectively. Figs. (a) to (c) show comparisons using the first 12 ASCCs, (d) to (f) use the first 20, and (g)-(i) use the first 29.

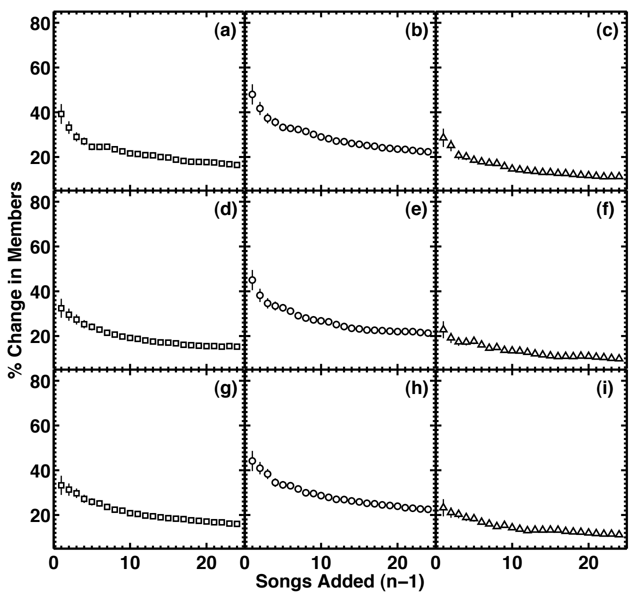


Figure 5. The percentage change in playlist members with the number of ASCCs used as a function of the length of playlist (excluding seed song). The data shown are the mean and 95% confidence intervals across all seed songs. The square, circle and triangle markers show comparisons between: 12 to 20, 12 to 29, and 20 to 29 ASCCs respectively. Figs. (a) to (c) show comparisons at 40 dB SPL, (d) to (f) at 80 dB SPL and (g)-(i) at 120 dB SPL.

sound (or disc jockey) context, where sound levels tend to be higher.

Another conclusion that may be drawn from the analysis is that low listening levels may be considered to produce a homogenization effect by limiting bandwidth (due to absolute thresholds). A similar effect is seen at high levels, where saturation and upward spread of excitation limit the effective number of independent ASCCs. It is conceivable that, given a larger set from which playlist members are drawn, the trends shown in Figs. 4 and 5 would resolve to a more signal or method dependent function, for example, it may be shown that the effect of listening level is more significant on certain genre. Future work should include modelling with larger sets of data.

Although demonstrated here using a music similarity study, the effect of listening level on auditory spectra may have wide ranging implications for MIR theory and practice in general, and initiating this debate was a primary aim of this article. It seems unlikely that changes in listening level will manifest changes in MIR properties relating to musical score (e.g., notation) or structure (e.g., segmentation). However, where MIR methods rely on spectrum (e.g., timbre) some effects of listening level may be expected. For example, speech (or even speaker) recognition in a high noise environment might be enhanced by the proper masking (noise suppression) effects of loud speech in the auditory model. In a more general sense, loudness itself may be a useful perceptual feature for MIR problems. For example, in the creation of a playlist, using a similar procedure to that described in the present paper, loudness and loudness dynamic range may be used to produce a sequence of songs which is tailored for smooth loudness transitions between tracks, and for similar loudness dynamics. Furthermore, incorporation of complete psychoacoustic listening conditions within listening tests designed to validate such perceptual similarity metrics may lead to more meaningful ground truth data.

6. CONCLUSIONS

In this paper we have presented a computational analysis of the effect of listening level on a perceptual music spectrum similarity metric. The similarity matrices and statistical data have shown that the metric is strongly level dependent. The playlist data shows similarly striking effects of listening level. Some general discussion has been given on the immediate implications of the use of listening-level dependent auditory models in MIR and loudness itself has been suggested as possible future similarity feature. The results of this study suggest that more complete data about sound [22] and about music production [6] may be useful to future context specific MIR applications.

7. REFERENCES

[1] J. J. Aucouturier. *Ten experiments on the modelling of polyphonic timbre*. PhD thesis, University of Paris, 2006.

[2] M. Casey, and M. Slaney. The importance of sequences

in musical similarity." *Proc. IEEE Int. Conf. ASSP*, 2006.

[3] M. A. Casey, R. Veltcamp, 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. B. Davis, and P. Mermelstein. "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences." *Proc. IEEE Trans. on Acoustic, Speech and Signal Processing*, Vol. 28, No. 4, pp. 357–366, 1980.

[5] A. Eronen, and A. Klapuri. "Musical instrument recognition using cepstral coefficients and temporal features." *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 753–756, 2000.

[6] G. Fazekas, and M. Sandler. "The Studio Ontology Framework." *Proc. 12th Int. Soc. for Music Information Retrieval, USA.*, 2011.

[7] B. R Glasberg, and B. C. J. Moore. "A model of loudness applicable to time-varying sounds." *J. Audio Eng. Soc*, Vol. 50, pp. 331–342, 2002.

[8] P. I. M Johannesma. "The pre-response stimulus ensemble of neurons in the cochlear nucleus." *Symp. Hear. Theory*, pp. 58–69, 1972.

[9] K. Metha. "Robust front-end and back-end processing for feature extraction for hinhi speech recognition." *Proc. IEEE Int. Conf. ICCIC*, 2010.

[10] E. Law, and L. von Ahn. "Input-agreement: a new mechanism for collecting data using human computation games." *Proc. 27th Int. Conf. on Human Factors in computing systems*, pp. 1197–1206, 2009.

[11] M. Levy, and M. Sandler. "Lightweight measures for timbral similarity of musical audio." *Proc. 1st ACM Workshop on Audio and Music Computing Multimedia*. 2006.

[12] B. Logan, and A. Salomon. "A music similarity function based on signal analysis." *Proc. IEEE Int. Conf. Multimedia and Expo*, pp. 745–748, 2001.

[13] B. Logan. "Mel frequency cepstral coefficients for music modeling." *Proc. Int. Symp. on Music Information Retrieval*, 2000.

[14] M. I. Mandel, G. E. Poliner, and D. P. W. Ellis. "Support vector machine active learning for music retrieval." *Proc. Int. Conf. on Music Information Retrieval*, 2005.

[15] B. C. J. Moore. *An Introduction to the Physiology of Hearing*. Academic Press, 1997.

[16] B. C. J. Moore, B. R. Glasberg, and T. Baer. "A model for the prediction of thresholds, loudness, and partial loudness." *J. Audio Eng. Soc*, Vol. 45, pp. 224–240, 1997.

[17] B. C. J. Moore ,and B. R. Glasberg. "Suggested formulae for calculating auditory-filter bandwidths and excitation patterns." *The Journal of the Acoustical Society of America*, Vol. 74, No. 3. pp. 750–753, 1983.

[18] E. Pampalk. *Computational models of music similarity and their application to music information retrieval*. PhD thesis.

[19] J O. Pickles. *An Introduction to the Physiology of Hearing*. 2008.

[20] M. D. Skowronski, and J. G. Harris. "Improving the filter bank of a classic speech feature extraction algorithm." *Proc. Int. Symp. Circuits and Systems*, Vol. 4, pp. 281–284, 2003.

[21] S. S. Stevens, J. Volkmann, and E. B. Newman. "A scale for the measurement of the psychological magnitude pitch." *The Journal of the Acoustical Society of America*, Vol. 8, No. 3, pp. 185–190, 1937.

[22] M. J. Terrell, A. J. R. Simpson, and M. Sandler. "Sounds not signals: A perceptual audio format." *Eng. Brief, AES 132nd Int. Convention*, 2012.

[23] G. Tzanetakis and P. Cook. "Musical genre classification of audio signals." *IEEE Trans. Speech Audio Processing*, Vol. 10, pp. 293–301, 2002.

PITCH CONTENT VISUALIZATION TOOLS FOR MUSIC PERFORMANCE ANALYSIS

Luis Jure[1] Ernesto López[2] Martín Rocamora[12] Pablo Cancela[2] Haldo Sponton[2] Ignacio Irigaray[2]
[1]School of Music and [2]Faculty of Engineering, Universidad de la República, Uruguay
`lj@eumus.edu.uy` {`elopez,rocamora,pcancela,haldos,irigiaray`}`@fing.edu.uy`

ABSTRACT

This work deals with pitch content visualization tools for the analysis of music performance from audio recordings. An existing computational method for the representation of pitch contours is briefly reviewed. Its application to music analysis is exemplified with two pieces of non-notated music: a field recording of a folkloric form of polyphonic singing and a commercial recording by a noted blues musician. Both examples have vocal parts exhibiting complex pitch evolution, difficult to analyze and notate with precision using Western common music notation. By using novel time-frequency analysis techniques that improve the location of the components of a harmonic sound, the melodic content representation implemented here allows a detailed study of aspects related to pitch intonation and tuning. This in turn permits an objective measurement of essential musical characteristics that are difficult or impossible to properly evaluate by subjective perception alone, and which are often not accounted for in traditional musicological analysis. Two software tools are released that allow the practical use of the described methods.

1. INTRODUCTION

Most of the established techniques for musical analysis do not work directly on the acoustic signal, but on some kind of symbolic representation of it [1]. This representation reduces the continuous and complex sound flow into a set of discrete events, usually determined by their most salient parameters, such as temporal location, duration and pitch. Applications of spectrographic analysis of sound to the development of new techniques of musical analysis began to be explored systematically with the work by Robert Cogan [3]. Using time-frequency representations of the audio signal, Cogan proposes an analytical method applicable to both structural and local aspects of a musical piece, that exemplifies analyzing music from very varied corpus. Recently, techniques based on sonographic representation have been applied extensively to the analysis of electroacoustic music [8]. These tools are also being applied to notated music or music from traditions not based on scores, to discuss aspects of music not represented in symbolic notation by the analysis of recordings. This may include both components that depend on the performance [7] (such as temporal and tuning micro-deviations), or the precise determination of the tuning system of a certain music [6].

Different software tools for computer-aided analysis, visualization and annotation of recorded music have been developed, for instance Sonic Visualiser.[1] They typically include traditional time-frequency representations and digital signal processing tools intended for music information retrieval, such as onsets or pitch detection. Some mid-level representations are also available, i.e., signal transformations that tend to emphasize higher semantics than the energy in the time-frequency plane [4]. Those mid-level representations are usually devised to facilitate the subsequent feature extraction and processing of an automatic algorithm. However, as suggested in [5], they can also be used by humans to study performance nuances such as pitch modulations or expressive timing.

In this article, examples are given of the type of analysis that can be done with an implementation of the pitch salience representation proposed in [2] by an end-user with a musicological background (first author). In addition, two graphical software tools are released that allow the practical use of the described methods by the research community. The representation proposed, called F0gram, is based on the Fan Chirp Transform (FChT) [13] and seeks two main goals: firstly, the precise time-frequency location of the components of a complex sound, using recent analysis techniques that overcome the limitations of the classical tools; secondly, the automatic grouping of all the components that are part of the spectrum of a single harmonic source, highlighting the fundamental frequency, f_0. This makes it possible to obtain an accurate graphical representation of the temporal evolution of the melodic content of a music recording, that allows the detailed study of performance aspects related to pitch intonation and timing (e.g. tuning system, vibrato, glissando, pitch slides).

The remaining of the document is organized as follows. Sections 2 and 3 briefly describe the time-frequency analysis and the pitch salience computation respectively. Examples of performance music analysis using the released tools are provided in section 4. The paper ends with some discussion on this work and ideas for future research.

[1] `http://www.sonicvisualiser.org/`

2. TIME-FREQUENCY ANALYSIS

Music audio signals often exhibit ample frequency modulation, such as the typical rapid pitch fluctuations of the singing voice. Precisely representing such modulations is a challenging problem in signal processing. It is reasonable to look for a signal analysis technique that concentrates the energy of each component in the time-frequency plane as much as possible. In this way, the representation of the temporal evolution of the spectrum is improved and the interference between sound sources is minimized, simplifying the task of higher level algorithms for estimation, detection and classification.

The standard method for time-frequency analysis is the Short Time Fourier Transform (STFT), which provides constant resolution in the time-frequency plane. A typical alternative for multi-resolution analysis is the Constant Q Transform (CQT). Both representations produce a Cartesian tiling of the time-frequency plane, as depicted in Figure 1. This may be inappropriate for non-stationary signals, for instance a frequency modulated sinusoid, namely a chirp. The virtue of the FChT is that it offers optimal resolution simultaneously for all the partials of a harmonic linear chirp, i.e. harmonically related chirps of linear frequency modulation. This is well suited for music analysis since many sounds have a harmonic structure and their frequency modulation can be approximated as linear within short time intervals.

The FChT can be formulated as [2],

$$X(f,\alpha) \triangleq \int_{-\infty}^{\infty} x(t)\, \phi'_\alpha(t)\, e^{-j2\pi f \phi_\alpha(t)} dt, \qquad (1)$$

where $\phi_\alpha(t) = (1 + \frac{1}{2}\,\alpha\, t)\, t$, is a time warping function. The parameter α, called the chirp rate, is the variation rate of the instantaneous frequency of the analysis chirp. Notice that by the variable change $\tau = \phi_\alpha(t)$, the formulation can be regarded as the Fourier Transform of a time warped version of the signal $x(t)$, which enables an efficient implementation based on the FFT. If a harmonic chirp is analyzed and the correct α value is selected for the transform, the warping yields sinusoids of constant frequency so the spectral representation is a set of very narrow peaks.

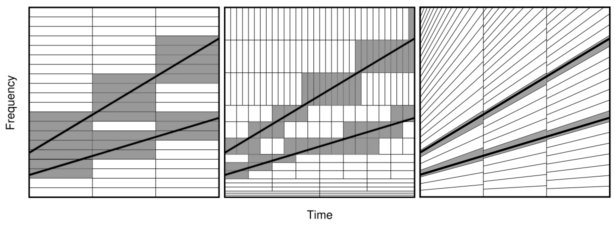


Figure 1. Time-frequency tiling sketch for the STFT, the Short Time CQT and the Short Time FChT and the resulting resolution for a two-component harmonic linear chirp.

A time-frequency representation can be built by computing the FChT for consecutive short time signal frames, namely a Short Time FChT (STFChT). This requires the determination of the optimal α value for each signal frame. For polyphonic music analysis there is no single optimal α value, so the approach followed in [2] is to compute several FChT instances with different α values. This yields a multidimensional representation made up of various time-frequency planes. The selection of the α values that produce the better representation of each sound present is performed by means of pitch salience. A comparison of the STFT and the STFChT applied to a polyphonic music audio clip is provided in Figure 2.

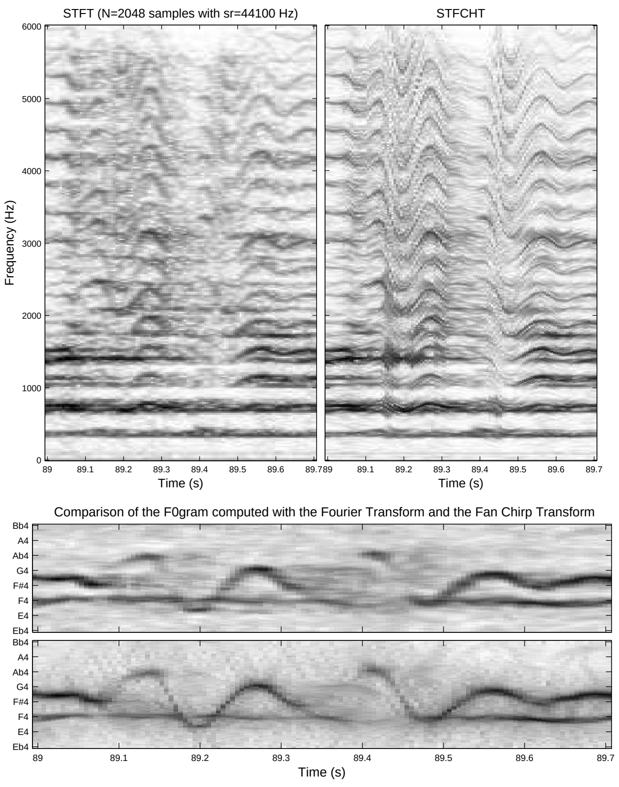


Figure 2. Above: Comparison of the STFT and SFChT for an excerpt from the example of section 4.1. The chirp rate of the most prominent sound source is selected for each frame. Note the improved representation obtained for this source while the rest is blurred. Below: F0grams obtained from the DFT and FChT. Rapid pitch fluctuations are better represented in the latter.

3. PITCH SALIENCE REPRESENTATION

A representation intended for visualizing the pitch content of polyphonic music signals should provide an indication of prominence or salience for all possible pitch values within the range of interest. A common approach for pitch salience calculation is to define a fundamental frequency grid, and compute for each frequency value a weighted sum of the partial amplitudes in a whitened spectrum [5,13]. A method of this kind was used in [2] and is briefly described in the following.

Given the FChT of a frame $X(f,\alpha)$, salience of fundamental frequency f_0 is obtained by summing the log-spectrum at the positions of the corresponding harmonics,

$$\rho(f_0, \alpha) = \frac{1}{n_H} \sum_{i=1}^{n_H} \log |X(if_0, \alpha)|, \qquad (2)$$

where n_H is the number of harmonics located up to a certain maximum analysis frequency. This is computed for each signal frame in a certain range of f_0 values.

Some postprocessing steps are carried out in order to attenuate spurious peaks at multiples and submultiples of the true pitches, and to balance different fundamental frequency regions [2]. Finally, for each f0 in the grid, the highest salience value is selected among the different available α values. In this way, a representation that shows the evolution of the pitch of the harmonic sounds in the audio signal is obtained, namely an F0gram. Examples of the resulting representation are depicted in Figure 3 for two short audio clips. The F0gram produces a fairly precise pitch evolution representation, contrast balanced and without spurious noticeable peaks when no harmonic sound is present. Note that simultaneous sources can be correctly represented, even in the case that they coincide in time and frequency if their pitch change rate is different. Figure 2 shows a comparison of the F0gram obtained from the DFT and the FChT. The improvement in time-frequency localization provides a more accurate representation of pitch.

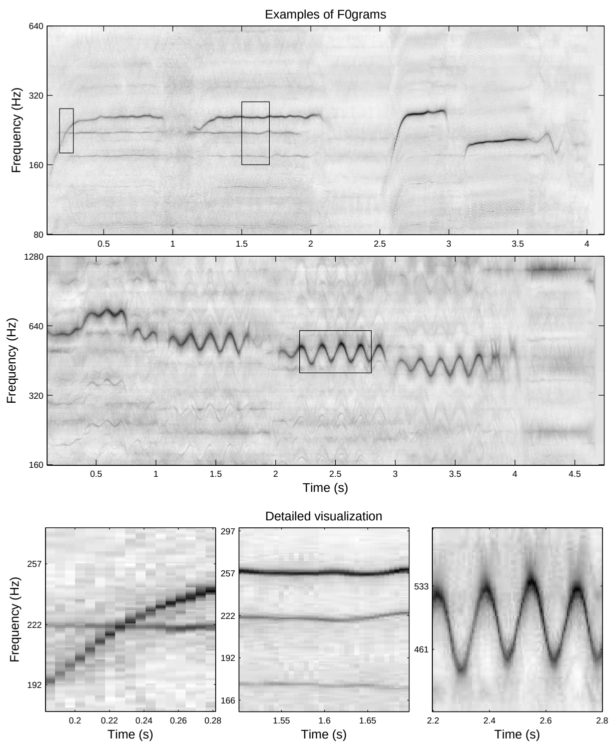


Figure 3. Above: F0gram examples for audio excerpts of *pop1.wav* and *opera_fem4.wav* from the MIREX melody extraction test set. Below: Detailed visualization. Crossing pitch contours are well resolved, and simultaneous sources and rapid pitch fluctuations are precisely represented.

4. CASE STUDIES

In order to exemplify the application of these techniques to musicological analysis, we have selected two pieces of non-notated music, both of them with vocal parts exhibiting complex pitch evolution, very difficult or downright impossible to notate with precision using Western common music notation: a field recording of a folkloric female vocal trio from west-central Bulgaria, and a commercial recording by noted blues singer and guitarist Muddy Waters.

4.1 Diaphonic chant of the Shope country

Throughout the world, folkloric forms of polyphonic singing are relatively scarce, one of the most notable exceptions being the diaphonic singing of the Shope region in west-central Bulgaria. A closely related form can be found in the Pirin region in the south-west of the country, and extending into the Republic of Macedonia.

As a general rule, these polyphonic songs are performed by female singers, and the sound itself of the voices is usually enough to impress listeners not familiar with this idiom. But the treatment of pitch in these two-part songs, or *dvuglas*, also has some unique characteristics.

In a typical setting, the melody part is sung by one singer, and the second part by two or sometimes more. The upper part has several classified melodic gestures, one of the most characteristic being a sort of—usually fast—glottal trill called *tresene*. Another characteristic gesture is the *izvikvane*, a form of ending the phrases with a fast upward leap on the vowel sound *"eee"*. The second part is more static, and has been described as a "drone" or pedal. It usually stays on the tonic of the mode, with occasional deviations to the sub-tonic when the melody descends to the tonic. Both parts join, however, to perform the *izvikvane* together. Apart from some fast swoops, the melody part moves within a very limited range, especially in the Shope region. This results in a preponderance of narrow intervals between the voices [9, 10].

For our case study, a commercially available field recording of a folkloric group from the Shope region was used [12]. The recording is identified, without further information, as a "Harvest Song" performed by a female vocal trio from the village of Zheleznitsa. The recording date can be placed around 1980. The song consists of 9 short phrases of similar duration (ca. 10∼12 s), structured as three variations of a group of three distinct phrases.

Figure 4 shows an F0gram of the third of these phrases, exhibiting all the characteristics described above: the second part begins in the sub-tonic and soon moves to the tonic for the rest of the phrase, while the first part moves both above and below the tonic, singing a more embellished melody that includes faster and slower *tresene*. The cadential *izvikvane* covers a narrow octave before descending back to the tonic area, and sounds like a unison of the three voices, in accordance with the prevailing description of *izvikvane*. The F0gram allows us to appreciate, however, that there is actually a slight separation of the voices, very difficult to perceive by listening alone.

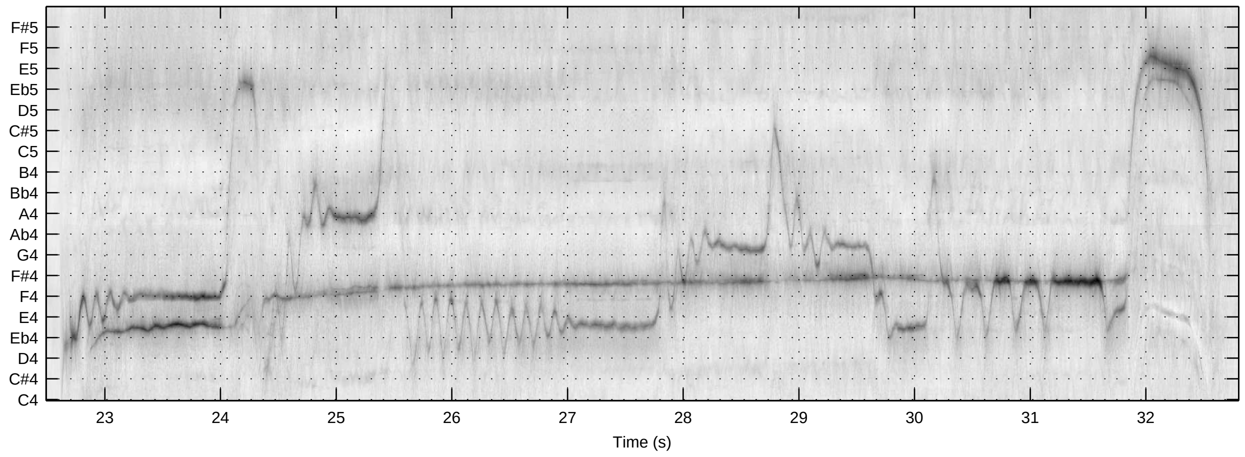


Figure 4. One phrase of the Harvest Song, showing characteristic traits of Shope diaphonic singing: melodic ornamentation in the first part (including *tresene* and a final *izvikvane*), and a pedal on the "tonic" in the second part, with a slow glissando.

An analysis of the simultaneities confirms that narrow intervals prevail, and a variety of intervals can be found between the unison and the major third. The F0gram obtained from the FChT permits a precise measurement of these type of intervals, as can be appreciated in Figure 2. Of special interest was the location of the sub-tonic, and the interval most frequently found lies half-way between one and two semitones below the tonic (sec. 23–24). This same kind of "second" can often be found above the tonic, in the upper part (sec. 28–29). The speed and range of the *tresene* can also be assessed with good precision. Typical rates are around 8~9 Hz (sec. 26–27), but slower rates can also be found (sec. 30–31). The width is variable, extending through intervals of up to three semitones (sec. 26).

So far, the analysis of the F0gram confirms—and permits a better measurement of—the characteristics described. Observing the second part with more detail, however, its behavior should be striking: instead of remaining on a fixed note, as its supposed character of "drone" would suggest, it performs a slow upward glissando, covering roughly the equivalent of a semitone (from F to F♯) during approximately 7 seconds. This displacement of the tonic not only occurs in various degrees in all the phrases throughout the song, but it also covers different pitch areas in each one (e.g., between E and F, or F♯ and G), resulting in a sort of "roving" tonic. Field recordings from different villages in the Shope region were analysed, and a similar behaviour was found in most of them, with glissandi of the "pedal" notes typically spanning between 50 cents and a semitone within a phrase. In the course of the song, the intonation of the local "tonics" can vary as much as three semitones.

The implications are two-fold and of paramount importance: unlike a typical drone or pedal point, essentially static, this second part has a dynamic character, and this kind of slowly ascending movement imposes on the polyphony a very particular tension and expressiveness. Additionally, the fact that the "tonic" varies between phrases, turns somewhat fuzzy the idea itself of modal tonic.

This phenomenon is not mentioned in the consulted bibliography and is not represented in the available transcriptions, although it was found in various degrees in several recordings analysed, suggesting that these traits conform a characteristic feature of the Shope musical idiom and should be considered an essential component of the powerful expressiveness of this particular form of folkloric polyphonic singing. These analysis techniques should be applied to a wider corpus to properly assess the importance of this performance practice.

4.2 Muddy Waters - Long Distance Call

The Blues is a genre of popular music deeply rooted in the African-American folksong tradition of the rural South of the United States, and as such it shows several traits that differ considerably from those found in the traditional European musical system. The most characteristic of these traits are the so-called *"blue notes"*, the precise definition of which has been elusive and even somewhat controversial. A simplistic but widely circulating definition reduces them to the use of the minor third and minor seventh degrees (sometimes also the diminished fifth) in a major-key context, for example, E♭ and B♭ in C major. Actually, this performance practice is much more complex, and entails two related but distinct aspects: the use of pitches that lie outside the standard Western tuning system, and continuous variations of pitch within certain tonal regions. Rather than fixed tones in a discrete scale system, blue notes would be flexible areas in the pitch space. For the analysis of the behaviour of these pitch complexes in actual performance, we chose a recording by Muddy Waters, one of the most important blues musicians of all time, regarded as an unsurpassed performer both as a guitarist *and* as a singer.

On January 23, 1951 he recorded his own composition "Long distance call" for Chess Records. He sings and plays electric guitar, and is accompanied by Marion "Little Walter" Jacobs on harmonica and Willie Dixon on double bass. The song is a standard 12-bar, three-line stanza blues, where the second line in each verse repeats the first, and the third is a rhyming conclusion. After a 4-bar introduction, the

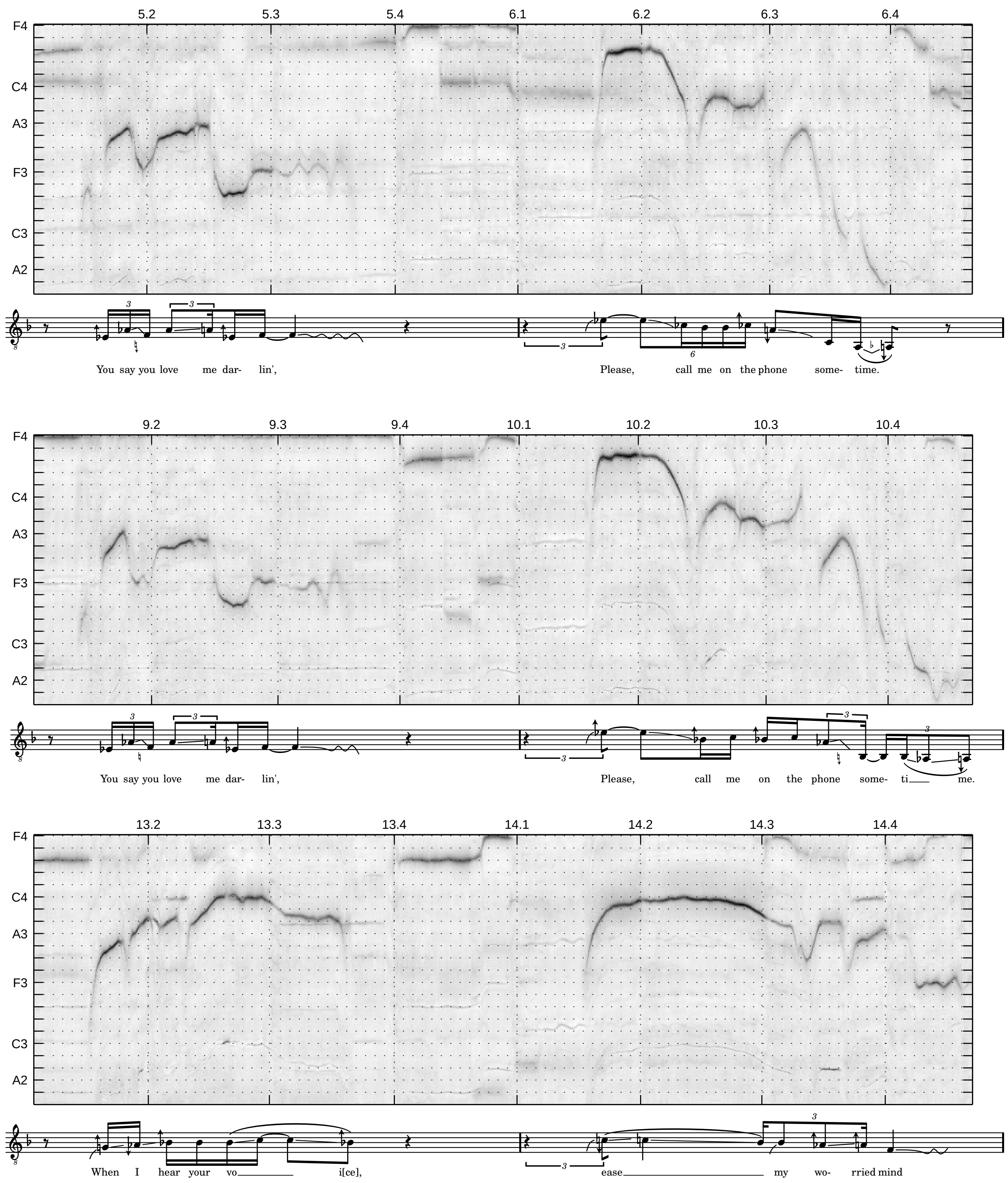


Figure 5. Muddy Waters, "Long distance call" (1951): F0gram showing continuous pitch contours of the voice, and approximate musical transcription informed by the analysis of the F0gram.

first stanza extends from measure 5 to 16. Figure 5 shows the F0gram and the transcription of the six measures where Muddy Waters sings the lyrics: mm. 5-6, 9-10 and 13-14. Each of these 2-bar vocal phrases is followed by a 2-bar instrumental response, omitted in the figure.

The musical transcription offered here is informed by the analysis of the F0gram,[2] and differs in many substantial details from published transcriptions [11], as well as from what was perceived by highly trained musicians that

[2] Just as with pitches, Muddy Waters' treatment of durations is equally flexible. The note values chosen for the transcription are approximate, and the vertical alignment with the F0gram is not always perfect.

were asked to listen to the recording. Observing the F0gram it is easy to see why: the melody consists mostly of continuously varying pitches, with few moments of stability other than the resolution on the tonic, and these exhibit a wide terminal vibrato. In this context, the perception of definite notes requires a decision on the part of the listener, that is partly subjective. For example, the first three notes ("You say you") are normally perceived as F-A♭-F, but a closer inspection reveal that the first note is actually a fast "scoop" around a slightly high E♭, and the second a continuous glide from below A♭ to around A♮. This behavior is consistent when the phrase is repeated (m. 9). A similar treatment of the third as a blue note (i.e., as a flexible pitch area) can be observed on the word "phone" on mm. 6 and 10. The previous words ("call me on the"), also move within a continuous pitch region, this time around the 4th and 5th degrees (B♭ and C). The long notes that begin the second half of each line ("*plea*-se" around E♭ on the first and second line, and "*ea*-se" around C on the third) exhibit all the same arch-like melodic contour, with wider and faster ascending and descending movements at the beginning and the end of the note, and a slow curve during the sustain part. A particularly expressive effect results from the fact that the C is hardly reached for an instant at the peak of the arch, the rest of the time the melody is kept moving slowly around a somewhat flat fifth. The F0gram also shows, through different shades in the grayscale, that the dynamics of the phrase follow a similar arch-like contour. The most ambiguous moments in terms of pitch are the endings of the first and second lines ("sometime"), with fast portamentos into the lower register that give these passages a speech-like quality.

In many Western vocal practices, continuous inflections of pitch are common when connecting the different —stable—notes of a melody (*legato* singing), as well as in the form of vibrato, fluctuations of pitch around a central perceived note. The application of the analysis tools proposed here permits a clear visualization of two salient traits of this passage: a melody consisting mostly of time varying pitches with relatively few moments of stability, and the establishment of continuous tonal regions non reducible to single pitches in a discrete scale.

5. DISCUSSION AND FUTURE WORK

By means of the analysis of two music recordings, the usefulness of the introduced techniques for computer aided musicology was illustrated, in particular for discussing expressive performance nuances related to pitch intonation. The result of the analysis by itself reveals important aspects of the music at hand, difficult to asses otherwise. The computational techniques implemented are oriented towards the precise representation of pitch fluctuations.

Two graphical software tools are released with this work to allow the application of the described methods by the research community. [3] One of them is a Vamp plugin [4] for Sonic Visualiser that computes the pitch contours representation. Within this application several other features are available that can assist the analysis. A Matlab® GUI is also released that includes additional functionalities and information, better suited for signal-processing researchers.

The improvement of the pitch contours representation and its application to different music scenarios are the following directions for future research.

[3] Available, as well as the audio clips, from `http://iie.fing.edu.uy/investigacion/grupos/gpa/ismir2012`

[4] `http://www.vamp-plugins.org/`

6. ACKNOWLEDGMENTS

This work was supported by the R+D Program of the Comisión Sectorial de Investigación Científica (CSIC), Universidad de la República, Uruguay.

7. REFERENCES

[1] I. Bent and A. Pople. "Analysis". *Grove Music Online*. Accessed June 16, 2012.

[2] P. Cancela, E. López, and M. Rocamora. Fan chirp transform for music representation. In *13th Int. Conf. on Digital Audio Effects, Austria*, Sep. 2010.

[3] R. Cogan. *New images of musical sound*. Harvard University Press. Cambridge, Massachusetts, 1985.

[4] J. Durrieu, B. David, and G. Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *Selected Topics in Signal Processing, IEEE Journal of*, 5(6):1180 –1191, Oct. 2011.

[5] A. Klapuri. A method for visualizing the pitch content of polyphonic music signals. In *10th Int. Society for Music Information Retrieval Conf., Japan*, 2009.

[6] A. Krishnaswamy. Pitch measurements versus perception of south indian classical music. In *Proc. of the Stockholm Music Acoustics Conf., Sweden, Aug.*, 2003.

[7] D. Leech-Wilkinson. *The Changing Sound of Music: Approaches to Studying Recorded Musical Performance*. Published online, London: CHARM, 2009.

[8] T. Licata, editor. *Electroacoustic Music - Analytical Perspectives*. Greenwood Press, 2002.

[9] L. Litova-Nikolova. *Bulgarian folk music*. Bulgarian academic monographs. Marin Drinov Academic Pub. House, 2004.

[10] S. Petrov, M. Manolova, and D. Buchanan. "Bulgaria". *Grove Music Online*. Accessed April 5, 2012.

[11] F. Sokolow and D. Rubin. *Muddy Waters - Deep Blues*. Hal Leonard Corporation, 1995.

[12] Musics & musicians of the world: Bulgaria. AUVIDIS/UNESCO, 1983.

[13] L. Weruaga and M. Képesi. The fan-chirp transform for nonstationary harmonic signals. *Signal Processing*, 87(6):1504–1522, 2007.

A MULTIPITCH APPROACH TO TONIC IDENTIFICATION IN INDIAN CLASSICAL MUSIC

Justin Salamon, Sankalp Gulati and Xavier Serra
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
{justin.salamon,sankalp.gulati,xavier.serra}@upf.edu

ABSTRACT

The tonic is a fundamental concept in Indian classical music since it constitutes the base pitch from which a lead performer constructs the melodies, and accompanying instruments use it for tuning. This makes tonic identification an essential first step for most automatic analyses of Indian classical music, such as intonation and melodic analysis, and raga recognition. In this paper we address the task of automatic tonic identification. Unlike approaches that identify the tonic from a single predominant pitch track, here we propose a method based on a multipitch analysis of the audio. We use a multipitch representation to construct a pitch histogram of the audio excerpt, out of which the tonic is identified. Rather than manually define a template, we employ a classification approach to automatically learn a set of rules for selecting the tonic. The proposed method returns not only the pitch class of the tonic but also the precise octave in which it is played. We evaluate the approach on a large collection of Carnatic and Hindustani music, obtaining an identification accuracy of 93%. We also discuss the types of errors made by our proposed method, as well as the challenges in generating ground truth annotations.

1. INTRODUCTION

One of the fundamental concepts in Indian classical music is the tonic. The tonic is a base pitch chosen by the performer, and serves as the foundation for the melodic tonal relationships throughout the performance. Every performer chooses a tonic pitch which best allows them to fully explore their vocal (or instrumental) pitch range for a given raga exposition [3]. Consequently, all accompanying instruments are tuned with relation to the tonic chosen by the lead performer.

Since the entire performance is relative to the tonic (corresponding to the *Sa* note of the raga), the lead performer needs to hear the tonic pitch throughout the concert. This is provided by a constantly sounding drone which plays in the background and reinforces the tonic. The drone may be produced by a variety of instruments such as the *Tanpura*, the electronic *Shruti box*, or by the sympathetic strings of an instrument such as the *Sitar* or *Veena*. Along with the tonic, the drone typically produces other important notes in the raga such as the *Pa* (fifth) or the *Ma* (fourth), and slightly less often the seventh (*Ni*), depending on the choice of raga. This drone serves as the reference sound that establishes all the harmonic and melodic relationships during a given performance. Other notes used in the performance derive their meaning and purpose in relation to the *Sa* and the tonal context established by the particular raga [2].

When considering the computational analysis of Indian classical music, it becomes evident that identifying the tonic is a crucial first step for more detailed tonal studies such as intonation [9], motif analysis [13] and raga recognition [1]. This makes automatic tonic identification a fundamental research problem. However, despite its importance in Indian classical music, the problem of automatic tonic identification has received very little attention from the research community to date.

To the best of our knowledge, all previous approaches for automatic tonic identification are based on applying monophonic pitch trackers to the audio recording, meaning they solely use the information proportioned by the predominant melody [16]. In some cases a monophonic pitch tracker is used even though the audio recording contains several instruments playing simultaneously [12]. These approaches have also been fairly restricted in terms of the musical content studied: in [16] only the *Alap* sections of 118 solo vocal recordings are used for evaluation, and in [12] the evaluation material is restricted to *Sampurna raga*. Both approaches also restrict the allowed frequency range for the tonic to a single octave, a limitation which can not be imposed if we wish to devise a single method for tonic identification for both male and female vocal performances.

In this paper we propose a method for tonic identification in Indian classical music based on a multipitch analysis of the audio signal. The motivation for a multipitch approach is twofold: first, the music material under investigation often includes several instruments playing simultaneously. Apart from the lead performer, recordings contain the drone instrument, and may also include other predominant instruments such as the violin, as well as percussive instruments. Second, we know that the tonic is continually reinforced by the drone instrument, an important fact that

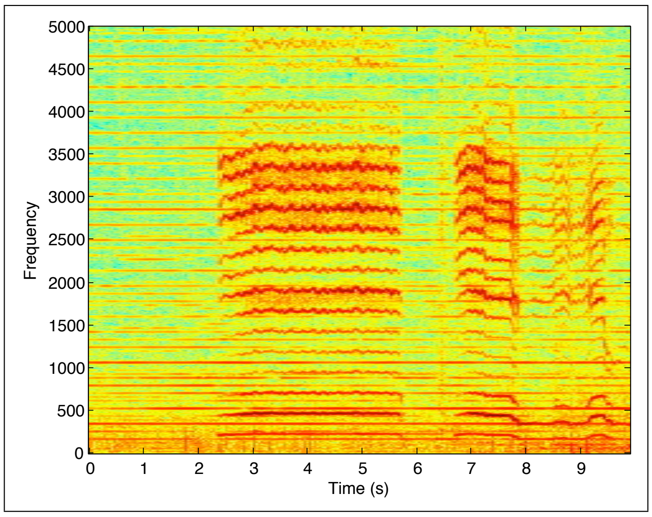


Figure 1. Spectrogram of an excerpt of Hindustani music with two clearly visible types of harmonic series, one belonging to the drone and the other to the lead voice.

is not exploited if we only extract a single pitch estimate for each frame of the recording. To illustrate this point, in Figure 1 we display the spectrogram for an excerpt of Hindustani music [2]. Two types of harmonic series are clearly visible in the spectrogram: the first type of harmonic series, which consist of almost perfectly flat lines, belong to the notes of the drone instrument (playing *Sa* and *Pa*). The second type of harmonic series (which starts roughly at time 2s) belongs to the voice of the lead performer. Evidently, if we only consider the pitch of the lead performer, we loose the pitch information proportioned by the drone instrument which in this case is a better indicator of the tonic pitch.

At the outset of this study, we defined three goals for the method to be developed: first, it should be applicable to a wide range of performances, including both the Carnatic [18] and Hindustani musical styles, male and female singers, and different recording conditions. Second, the approach should identify the tonic pitch in the correct octave, without restricting the allowed frequency range to a single octave. Finally, the approach should be able to identify the tonic using a limited segment of the full recording, and this segment can be taken from any part of the piece.

The structure of the remainder of the paper is as follows. In Section 2 we present our proposed tonic identification method. In Section 3 we describe the evaluation methodology employed in this study, including the music collection used for evaluation and the annotation procedure used to generate the ground truth. Then, in Section 4 we present and discuss the results of the evaluation, and finally in Section 5 we provide some conclusions and proposals for future work.

2. PROPOSED METHOD

The proposed method is comprised of four main blocks: sinusoid extraction, salience function, candidate generation and tonic selection. The first two blocks of the system were originally proposed as part of a predominant melody extraction system [14, 15], and have been adapted here for the task of tonic identification. In the following sections we describe the processing steps involved in each of the four blocks of the system. A block diagram of the proposed method is provided in Figure 2.

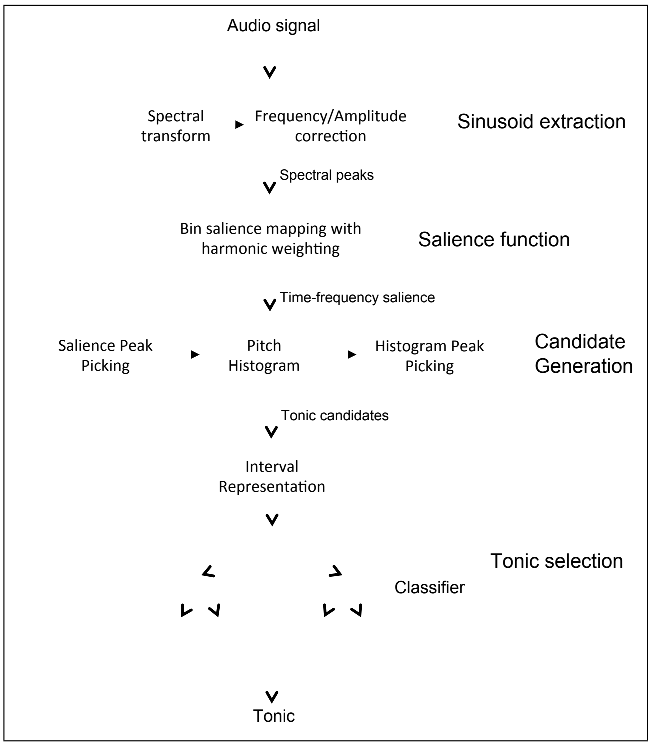


Figure 2. Block diagram of the proposed tonic identification method.

2.1 Sinusoid Extraction

In the first step of the method, we extract sinusoidal components, i.e. spectral peaks, from the audio signal. The sinusoid extraction process is divided into two stages as depicted in Figure 2: spectral transform and sinusoid frequency/amplitude correction.

We start by applying the Short-Time Fourier Transform (STFT) given by:

$$X_l(k) = \sum_{n=0}^{M-1} w(n) \cdot x(n + lH) e^{-j\frac{2\pi}{N}kn}, \qquad (1)$$
$$l = 0, 1, \ldots \text{ and } k = 0, 1, \ldots, N-1$$

where $x(n)$ is the time signal, $w(n)$ the windowing function, l the frame number, M the window length, N the FFT length and H the hop size. We use the Hann windowing function with a window size of 46.4ms, a hop size of 2.9ms and a $\times 4$ zero padding factor, which for data sampled at $f_S = 44.1$kHz gives $M = 2048$, $N = 8192$ and $H = 128$. Given the FFT of a single frame $X_l(k)$, spectral peaks are selected by finding all the local maxima k_m of the magnitude spectrum $|X_l(k)|$.

The location of the spectral peaks is limited to the bin frequencies of the FFT, which for low frequencies can result in a relatively large error in the estimation of the peak

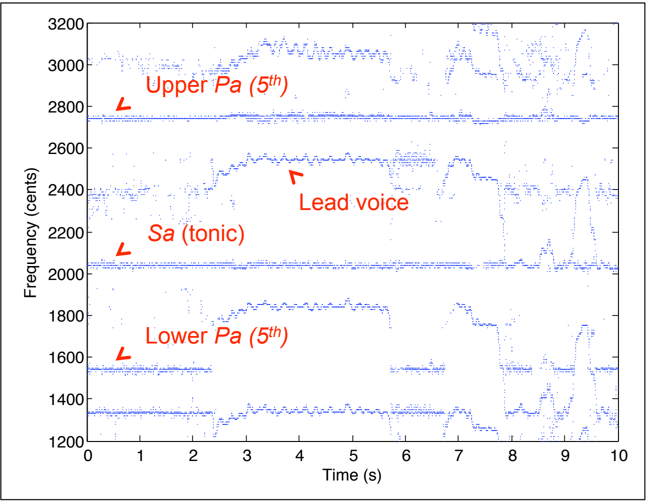


Figure 3. Peaks of the salience function for an excerpt of Hindustani music.

frequency. To overcome this quantisation, in the second stage of this block we apply the approach described in [4], in which the phase spectrum $\phi_l(k)$ is used to calculate the peak's instantaneous frequency (IF) and amplitude, which provide a more accurate estimate of the peak's true frequency and amplitude.

2.2 Salience Function (Multipitch Representation)

We use the extracted spectral peaks to compute a *salience function* – a multipitch time-frequency representation of pitch salience over time. The salience computation is based on harmonic summation similar to [8], where the salience of a given frequency is computed as the sum of the weighted energies found at integer multiples (harmonics) of that frequency. Peaks of the salience function at each frame represent salient pitches in the music recording. Note that whilst the concepts of pitch (which is perceptual) and fundamental frequency (which is a physical measurement) are not identical, for simplicity in this paper we will use these two terms interchangeably.

Our salience function covers a pitch range of nearly five octaves from 55Hz to 1.76kHz, quantized into 600 bins on a cent scale (10 cents per bin). The reader is referred to [14, 15] for further details about the mathematical formulation of the salience function. In Figure 3 we plot the peaks of the salience function for the same excerpt from Figure 1. The tonic (*Sa*) pitch which is played by the drone instrument is clearly visible, as well as the upper and lower fifth (*Pa*), and the pitch trajectory of the voice.

2.3 Tonic Candidate Generation

As explained earlier, the peaks of the salience function represent the pitches of the voice and other predominant instruments present in the recording at every point in time. Thus, by computing a histogram of the pitch values for the entire excerpt, we obtain an estimate of which pitches are repeated most often throughout the excerpt. Though pitch histograms have been used previously for tonic identification [12], they were constructed using only the most predominant pitch at each frame, which means that in many cases the tonal information provided by the drone instrument is not taken into consideration.

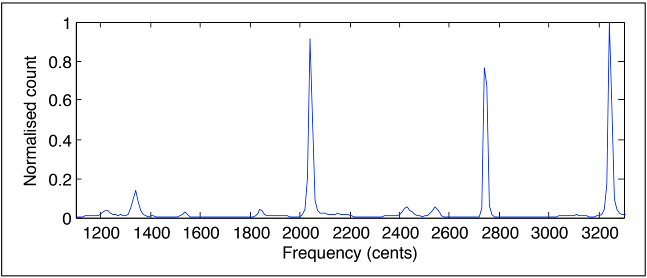


Figure 4. Pitch histogram for an excerpt of Hindustani music.

We start by taking the peaks of the salience function at each frame. Since the frequency range for the tonic pitch selected by singers in Indian classical music is relatively limited, we can reduce the range from which salient pitches are selected. To ensure we cover the complete range for both male and female singers, we consider salient pitches with a fundamental frequency ranging from 110 Hz to 370 Hz. Importantly, note that this range spans almost 2 octaves, meaning the system must be able to identify not only the correct tonic pitch class, but also the octave in which it is played. Within this range, at each frame we take the top five peaks (pitches) of the salience function.

The selected pitches are used to construct a pitch histogram. As the drone is usually weaker than the lead voice, we avoid weighting each peak by its magnitude. The resulting pitch histogram goes from 110 Hz to 370 Hz and has a resolution of 10 cents. Peaks of the histogram represent the most frequent pitches in the excerpt, one of which will be the tonic. In Figure 4 we present the histogram computed from the complete 3 minute excerpt used in the previous examples. The pitch axis is plotted in cents, and the histogram is normalised by the magnitude of its highest peak. For the excerpt under consideration, we note three clear peaks: the tonic *Sa* (2040 cents), the upper *Pa* (2740 cents) and the tonic again, one octave up (3240 cents). This illustrates one of the challenges the system will have to deal with – selecting the tonic at the correct octave. It also highlights another important issue – the peak corresponding to the tonic will not always be the highest peak in the histogram, meaning the (perhaps naïve) approach of selecting the highest peak of the histogram would not provide satisfactory results.

2.4 Tonic Selection

As the tonic will not always be the highest peak of the histogram, we take the top 10 peaks of the pitch histogram p_i $(i = 1 \ldots 10)$, one of which represents the pitch of the tonic. As mentioned in the introduction, all other notes present in the musical piece are tuned with relation to the tonic. Bearing this in mind, we hypothesize that the tonic can be identified based on the pitch intervals between the most frequent notes in the recording and their rate of occurrence. For example, in the excerpt in Figure 3, the drone plays the tonic alongside the lower and upper fifth. Thus, a fifth relationship between two frequent notes might serve as a good indicator for the tonic.

In the study of Western music, templates learned from music cognition experiments have been used for the related task of key detection, where a pitch histogram (derived from a symbolic representation of the musical piece) is matched against templates representing the probability of different pitch classes given a certain tonal context [10]. Approaches based on training a classifier to determine the key of a musical piece using chroma features automatically extracted from the audio signal have also been proposed [5]. In this study, we propose a classification approach to automatically learn the best set of rules for selecting the tonic, based on the pitch intervals between the most frequent notes in the piece and their relative rate of occurrence (as indicated by the magnitude of the peaks of the pitch histogram).

We start by annotating for each piece the rank $i = I$ of the tonic (in terms of peak magnitude) out of the top 10 peaks p_i of the pitch histogram. Then, we encode the 10 tonic candidates as the distance (in semitones) between every candidate p_i and the highest candidate in the histogram p_1. This gives us a set of features f_i ($i = 1 \ldots 10$), where f_i represents the distance (in semitones) between p_i and p_1. The features f_i and the annotated rank of the tonic I are used to train a classifier for selecting the tonic. That is, we pose the task of tonic identification as a classification problem where we have 10 classes (10 candidates) and the classifier must choose the rank of the candidate corresponding to the tonic. Note that for all files in our collection the tonic was always amongst the top 10 peaks p_i of the pitch histogram.

For classification we use the Weka data-mining software [7]. We start by performing attribute selection using the *CfsSubsetEval* attribute evaluator and BestFirst search method [6] with a 10-fold cross validation, only keeping features that were used in at least 80% of the folds. The selected features were: f_2, f_3, f_5, f_6, f_8 and f_9. Then, we train a C4.5 decision tree [11] in order to learn the optimal set of rules for selecting the tonic based on the pitch intervals between the tonic candidates. Note that we also evaluated other classification algorithms, namely support vector machines (SMO with polynomial kernel) and an instance-based classifier (k*) [19]. However, the accuracy obtained using the decision tree was significantly higher (6% better than SVM and 5% better than k*), and so for the rest of the paper we will focus on the results obtained using this classifier. Additionally, using a decision tree has the advantage that the resulting classification rules can be easily interpreted and, as shall be seen, are musically meaningful.

The resulting tree is presented in Figure 5. As it turns out, only 3 features are finally used: f_2, f_3 and f_5. Another interesting observation is that the pitch intervals used by the tree for making decisions correspond quite well to the intervals between the notes commonly played by the drone instrument: 5 (i.e. 500 cents) corresponds to the interval between the lower *Pa* and the tonic *Sa*, and 7 (700 cents) to the interval between the *Sa* and upper *Pa*. Note that a distance of 500 cents may also correspond to the distance between the *Sa* and upper *Ma*, which might be a cause for confusion in our system, and we will assess this when we analyse the results.

Examining the rules of the tree, we see that the most important relationship is between the top two peaks of the histogram. When the second highest peak is more than 500 cents above the highest peak, the latter is chosen as the tonic. Examining the data we found that this almost always corresponds to one of two cases – the second peak is either *Pa* (i.e. *Pa* tuning) or *Sa* one octave above the tonic. Branching left, the tree checks whether the highest peak is actually *Pa* (700 cents above the tonic). To confirm this it checks if the third peak is found 500 cents above the highest peak (thus corresponding to *Sa* one octave above the tonic). In this case the highest peak is indeed *Pa*, and the second highest peak is the tonic. Otherwise, we have a case of *Ma* tuning (the second peak is tuned to *Ma*), and the highest peak is the tonic. Similar interpretations can be made for the remaining rules of the tree.

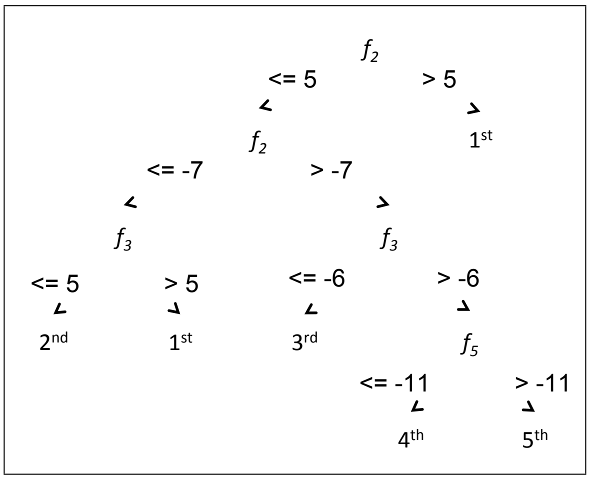


Figure 5. Obtained decision tree for tonic identification.

3. EVALUATION METHODOLOGY

3.1 Music Collection

The music collection used to evaluate the proposed approach was compiled as part of the CompMusic project [17]. It consists of 364 excerpts of Indian classical music including both Hindustani (38%) and Carnatic (62%) music. The excerpts were extracted from 231 unique performances by 36 different artists, including both male (80%) and female (20%) singers. Every excerpt is 3 minutes long, and extracted from either the beginning, middle or end of the full recording (for recordings longer than 12 minutes we are able to extract all 3 excerpts, for shorter recordings a single excerpt from the beginning of the piece was taken). Including excerpts from sections other than the beginning of the piece is important, since in both the Hin-

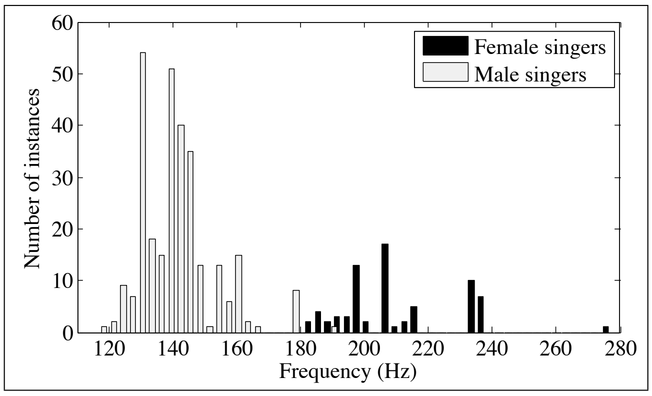


Figure 6. Distribution of tonic frequency for male and female vocal performances in our music collection.

dustani and Carnatic music traditions different sections of a performance can have very different acoustic characteristics. In Figure 6 we display the distribution of tonic frequencies in our collection for both male and female singers.

3.2 Annotation Procedure

The tonic frequency for each excerpt was manually annotated by the authors. To assist the annotation process, we used the candidate generation part of our proposed method to extract 10 candidate frequencies for the tonic in the range of 110 Hz to 300 Hz. The annotator could then listen to the candidate frequencies one by one together with the original recording in order to identify the tonic frequency. Note that for all excerpts in our collection the true tonic frequency was present in one of the 10 candidates provided by the system.

It is worth noting that as part of the annotation process, the listener must determine the octave in which the tonic is played. Since the drone instrument may play the tonic pitch in two octaves simultaneously, the octave of the tonic is determined by the vocal range of the singer rather than the drone instrument directly. Whilst in most cases the correct octave is fairly unambiguous for vocal performances, we encountered a small number of cases in which determining the octave of the tonic was more difficult. In future work, we intend to study the relation between performer and drone instrument in greater depth, as well as conduct listening tests to assess the degree of agreement between listeners when asked to determine the octave of the tonic.

4. RESULTS

We evaluate the proposed classification-based approach using 10-fold cross validation. The experiment is repeated 10 times, and the average results for all 10 repetitions are reported. In Figure 7 we present the classification accuracy obtained for our collection of 364 excerpts, as well as a breakdown of the results based on musical style and gender of the lead performer.

We see that the proposed approach obtains a classification accuracy (hence tonic identification accuracy) of 93% for our complete collection. Importantly, since the allowed

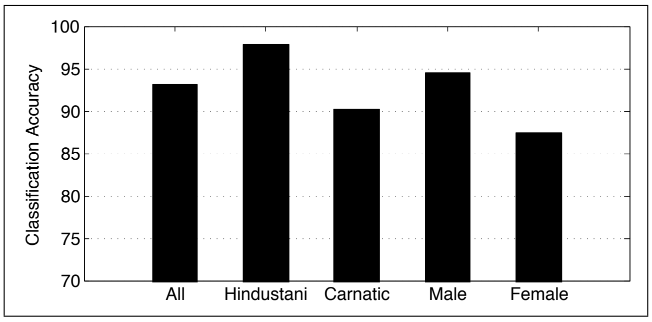


Figure 7. Classification accuracy for the proposed approach. All excerpts 93%, Hindustani 98%, Carnatic 90%, Male 95% and Female 88%.

tonic frequency range spans more than one octave, it means we are correctly identifying not only the pitch-class of the tonic, but also the octave at which it is played. Next, we examine the results depending on the musical style. We see that we have almost perfect classification for Hindustani music (98%), whilst for Carnatic music the performance is somewhat lower (90%). When examining the data, we noted that in the Carnatic excerpts there were more cases where the *Tanpura* was quite weak (in terms of loudness). Consequently, this results in frames where the pitch corresponding to the tonic does not have a prominent peak in the salience function. This in turn means the peak of the pitch histogram which corresponds to the tonic has a fairly low rank, leading to incorrect identification of the tonic.

When considering identification accuracy as a function of the gender of the lead performer, we see that the system performs better for pieces performed by male singers compared to those performed by female singers. A possible cause for this is the different amount of male and female performances in our collection. Since there are considerably more male performances, the rules learned by the system are better suited for identifying the tonic in this type of musical material. Another factor that was identified as influential was the frequency range used to compute the pitch histogram. Whilst our frequency range covers the entire range in which we expect to find the tonic for both male and female cases, for high frequency tonics this range will not include the higher *Sa* one octave above the tonic. As it turns out, the presence of a higher *Sa* is one of the cues used by the system, and for many female excerpts it is outside the range of the pitch histogram. In the future, we intend to experiment with different frequency ranges for the pitch histogram, as well as consider separate ranges for male and female performances to see whether performance can be improved by including this extra piece of information prior to classification.

As a final step in our analysis of the results, we checked what types of errors were the most common in our evaluation. We found that for male singers the most common error was selecting the higher *Pa* or *Ma* as the tonic, whilst for females it was selecting the lower *Pa* or *Ma*. This is understandable, as these are two important notes that are often played by the drone instrument in addition to the tonic. The difference in tonic frequency for males and females, together with the frequency range used for the pitch histogram, explains why for males we erroneously select a higher note, whilst for females we select a lower one. Additionally, for female singers we found that the confusion was often caused due to the use of *Ma* tuning (*Sa* - *Ma* - *Sa*) of the drone instrument. If the higher *Sa* is not present, the *Ma* tuning is equivalent to a rotated version of *Pa* tuning, resulting in the wrong rule being applied.

5. CONCLUSION

In this paper we presented a novel approach for tonic identification in Indian classical music. Our method is based on a multipitch analysis of the audio signal, in which the predominant pitches in the mixture are used to construct a pitch histogram representing the most frequently played notes in the piece. In this way, our representation also captures the notes played by the drone instrument, and not only the pitch of the lead performer. Using a classification approach, we were able to automatically learn the best set of rules for tonic identification given our pitch histogram representation. The resulting decision tree was evaluated on a large collection of excerpts consisting of a wide selection of pieces, artists and recording conditions, and was shown to obtain high tonic identification accuracy. Importantly, the approach is suitable for both Hindustani and Carnatic music, male and female performances, and only requires a short excerpt of the full performance. In addition, the rules learned by the system are easy to interpret and musically coherent.

Following presentation of the results, we discussed the types of errors most commonly made by the proposed tonic identification method, and the main causes for these errors where identified. Finally, we proposed some directions for future work, including a study of tonic octave perception, considering different frequency ranges for the pitch histogram in our proposed method, and devising gender-specific tonic identification approaches.

6. ACKNOWLEDGMENTS

The authors would like to thank Joan Serrà, Emilia Gómez and Perfecto Herrera for their useful comments. This research was funded by the Programa de Formación del Profesorado Universitario (FPU) of the Ministerio de Educación de España and the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 267583 (CompMusic).

7. REFERENCES

[1] P. Chordia, J. Jagadeeswaran, and A. Rae. Automatic carnatic raag classification. *J. of the Sangeet Research Academy (Ninaad)*, 2009.

[2] A. Danielou. *The Ragas of Northern Indian Music*. Munshiram Manoharlal Publishers, New Delhi, 2010.

[3] B. C. Deva. *The Music of India: A Scientific Study*. Munshiram Manoharlal Publishers, Delhi, 1980.

[4] K. Dressler. Sinusoidal extraction using an efficient implementation of a multi-resolution FFT. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, pages 247–252, Montreal, Quebec, Canada, Sept. 2006.

[5] E. Gómez and P. Herrera. Estimating the tonality of of polyphonic audio files: Cognitive versus machine learning modelling strategies strategies. In *5th Int. Conf. on Music Info. Retrieval*, Barcelona, Spain, Oct. 2004.

[6] M. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1999.

[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11:10–18, November 2009.

[8] A. Klapuri. Multiple fundamental frequency estimation by summing harmonic amplitudes. In *7th Int. Conf. on Music Info. Retrieval*, Victoria, Canada, October 2006.

[9] G. K. Koduri, J. Serrà, and X. Serra. Characterization of intonation in carnatic music by parametrizing pitch histograms. In *13th Int. Soc. for Music Info. Retrieval Conf.*, Porto, Portugal, Oct. 2012.

[10] C. L. Krumhansl. *Cognitive Foundations of Musical Pitch*. Oxford University Press, New York, 2001.

[11] R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[12] T.V. Ranjani, H.G.; Arthi, S.; Sreenivas. Carnatic music analysis: Shadja, swara identification and rAga verification in AlApana using stochastic models. *Applications of Signal Processing to Audio and Acoustics (WASPAA), IEEE Workshop*, pages 29–32, 2011.

[13] J. C. Ross, T. P. Vinutha, and P. Rao. Detecting melodic motifs from audio for Hindustani classical music. In *13th Int. Soc. for Music Info. Retrieval Conf.*, Porto, Portugal, Oct. 2012.

[14] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, Aug. 2012.

[15] J. Salamon, E. Gómez, and J. Bonada. Sinusoid extraction and salience function design for predominant melody estimation. In *Proc. 14th Int. Conf. on Digital Audio Effects (DAFX-11)*, pages 73–80, Paris, France, Sep. 2011.

[16] R. Sengupta, N. Dey, D. Nag, A. Datta, and A. Mukerjee. Automatic Tonic (SA) Detection Algorithm in Indian Classical Vocal Music. In *National Symposium on Acoustics*, pages 1–5, 2005.

[17] X. Serra. A multicultural approach in music information research. In *12th Int. Soc. for Music Info. Retrieval Conf.*, Miami, USA, Oct. 2011.

[18] T. Viswanathan and M. H. Allen. *Music in South India*. Oxford University Press, 2004.

[19] I. H. Witten and E. Frank. *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann, Waltham, USA, 2nd edition, 2005.

SEPARATING PRESENTATION AND CONTENT IN MEI

Laurent Pugin
Swiss RISM / Fribourg University
laurent.pugin@rism-ch.org

Johannes Kepper
Edirom
kepper@edirom.de

Perry Roland
University of Virginia
pdr4h@eservices.virginia.edu

Maja Hartwig
Edirom
maja.hartwig@gmx.de

Andrew Hankinson
McGill University, Schulich School of Music
andrew.hankinson@mail.mcgill.ca

ABSTRACT

Common Western music notation is traditionally organized on staves that can be grouped into systems. When multiple systems appear on a page, they are arranged from the top to the bottom of the page, similar to lines of words in a text document. Encoding music notation documents for printing requires this arrangement to be captured. However, in the music notation model proposed by the Music Encoding Initiative (MEI), the hierarchy of the XML sub-tree representing the music emphasizes the content rather than the layout. Since systems and pages do not coincide with the musical content, they are encoded in a secondary hierarchy that contains very limited information. In this paper, we present a complementary solution for augmenting the level of detail of the layout of musical documents; that is, the layout information can be encoded in a separate sub-tree with cross-references to other elements holding the musical content. The major advantage of the proposed solution is that it enables multiple layout descriptions, each describing a different visual instantiation of the same musical content.

1. INTRODUCTION

Common Western music notation is a system made up of structured symbols organized upon a group of horizontal lines, commonly called a “staff”, which acts as a bidimensional reference system. The horizontal axis represents time while the vertical axis indicates pitch. Staves can be grouped into systems, where the systems contain concurrent streams of musical events aligned vertically and where each staff encompasses a defined pitch range. Systems are arranged across as many pages as necessary to accommodate the musical content. When multiple systems appear on a page, multiple systems are arranged from the top of the page to the bottom, similar to paragraphs in a text document.

Numerous schemes have been developed for encoding music notation [8]. Over the last decade, XML has been increasingly used for defining encoding schemes, for example, in the MusicXML[1] interchange format [2] and the IEEE1599[2] standard [6]. More recently, with a major release in 2010 and with the upcoming 2012 release, the music notation model proposed by the Music Encoding Initiative[3] (MEI) has begun to take a leading role. Developed by a community of scholars, it acts as an extensible music document encoding framework that can be customized for specific needs [5].

For XML encoding schemes, such as MEI, that aim to take into account the graphical context of the notation, the organization of the notation into staves, systems, and pages often needs to be captured. Whereas a page-based approach will have the page at the top of the XML hierarchy, a content-based approach will place an element with semantic meaning at the top of the hierarchy, relegating the visual appearance to a secondary role. Music notation itself is obviously multi-hierarchical, and both approaches reflect valid perspectives. However, a basic principle of XML design is that it requires a single hierarchy to become the primary ordering mechanism of the music notation description. Other hierarchies inherent in music notation may then be implemented using alternative techniques such as standoff markup.

Currently, MEI emphasizes the logical content of the notation. For example, in the case of CMN, it employs measures at the top of the hierarchy. Pages and systems are captured using the same milestone technique that TEI offers; that is, page and system breaks are represented by the empty elements <pb/> and <sb/> respectively. It is fairly easy to convert between measure-based and page-based hierarchies using XSLT stylesheets, analogous to MusicXML’s conversion between time-based and part-based file organization. However, there are additional complicating factors in the case of MEI. For example, when multiple sources are described within a single encoding, which is a significant design goal of MEI, the sources do not necessarily agree with regard to page and system breaks. Furthermore, they might use a different

[1] <http://www.makemusic.com/musicxml>
[2] <http://www.mx.dico.unimi.it>
[3] <http://www.music-encoding.org >

score order or map instruments to staves differently. Even the number of instruments or staves may differ between multiple sources. Although MEI is currently capable of dealing with these circumstances, the markup is often verbose, repetitive, and difficult to comprehend quickly. In this paper, we present a complementary module for MEI that provides for more detailed capture of layout information and better separation of musical content and visual presentation. The next section describes the objectives pursued, followed by a section on related work. We then present the module we developed for MEI and conclude the paper with remarks on future work.

2. OBJECTIVES

There are at least two use-cases that would benefit from a clearer separation of layout-related information and the musical content as proposed in this paper. The first use-case is when precise descriptions in the encoding of existing source materials are required. A typical example is the use of MEI as an output of and archival format for optical music recognition (OMR) software applications [4]. In such a use, it is necessary to be able to record the exact position of the elements on the page. In OMR transcriptions, each note, each music symbol, but also each staff and each system requires its coordinate to be stored in the MEI encoding. Diplomatic transcriptions with exact coordinates are not only useful as interchange and training data for adaptive OMR software applications, but they can also be used in digital edition environments for producing transcription image overlays. A diplomatic transcription can be shown directly on top of the original source, either for highlighting a particular aspect of the source or simply for facilitating its readability. Examples already exist for text editions [7], and a similar approach for music could very well be envisaged with MEI.

Such a model would also serve the second use-case, which is the preparation of different renditions from the same musical material, the typical case being an edition of the full score and, in parallel, an edition of the performers' parts. While it is relatively easy to extract parts from a score encoded in MEI, there will always be cases where human intervention will be required to finalize the layout of the parts, whatever the automatic layout capabilities of the rendering software application used. The modifications can include additional dynamic markings, lyrics, directives and similar musical information encoded in nearby staves. The ideal solution is to encode only the layout modifications applied to the parts so that additional changes to the score would automatically be reflected in the parts. This means that a <note> element for which only the stem direction is changed in the layout need not be duplicated. Features like this already exist in some music notation software applications, such as in Sibelius©, which includes a so-called Dynamic Parts™ functionality. However, they are not designed to handle multiple sources. Having an option to record this type of layout information in an optimized manner would certainly be valuable.

2.1 Requirements

In order optimally to increase the level of detail of the documents encoded in MEI, it is necessary to achieve a solution that will not overload the logical sub-tree that holds the musical content. When mingled with notation content, page and system milestone markers complicate the encoding of content. Adding more detailed layout information, such as page size, results in further complication.

The solution should avoid overlapping hierarchy problems whenever possible. Page breaks and system breaks embedded in the content sub-tree represent a non-concurrent hierarchy. Multiple sources requiring different presentation exacerbate the problem by creating multiple instances of non-concurrent hierarchies.

Furthermore, it is important for the solution to limit strictly the amount of duplicated data in the encoding. For example, in the case of "score and parts" editions, when the data for the parts duplicates that of the score, the score data and the parts data may become desynchronized. However, since the duration of a note in the parts should be the same as in the score, employing a reference system eliminates this possibility.

The proposed solution should not require the user to choose between content-based or page-based approaches but should supplement the current content-focused representation of MEI instead. With that in mind it becomes clear that the use of this additional layout information has to be optional – for users, but also for applications. This means that applications unaware of this proposal may safely ignore it, that the additional information provided by this proposal must leave the musical content sub-tree untouched as much as possible, and that links added between the content and the layout elements must not preclude the encoding and decoding of the musical content on its own.

3. RELATED WORK

For some aspects, the problem described above is similar to what is achieved by OMR software applications such as Photoscore© that extend MusicXML in order to store exact positioning information. However, it is done in a non standard way and is application dependent. There are also several standard existing encoding strategies and formats that seem to be relevant to the problem described above. The following section will introduce them briefly and discuss their applicability for describing multiple renditions or sources of the same musical content.

3.1 TEI Encoding Model for Genetic Editions

The aim of providing a detailed model of how content is laid out on the page is similar to the goal pursued by the TEI Workgroup on Genetic Editions [3]. Their model essentially follows a document-centric approach, as opposed to the traditional text-centric approach for TEI. The alternative hierarchy of this model privileges the document and is organized as follows:

- Document
  - Writing surface (page, double page, folium, etc.)
    - Zone
      - Text, lines or tables

The model is designed for encoding complex cases of manuscripts in various stages of creation. Its purpose is to trace and encode their genesis. In that regard, it is different from what we hope to achieve for MEI because this model aims principally for a chronological ordering of zones in one document rather than transcribing or defining the layout of multiple documents separately.

Furthermore, the TEI encoding model for genetic editions is an alternative model for encoding a document. It is not designed to be applied on top of an existing, traditional TEI encoding. Links are not maintained between the textual content of the document and separately encoded layout information. For this reason, some TEI projects adopt a cumbersome double-encoding approach, with one encoding for the representation of the source text(s) and a second encoding for the documentary edition [1], which it would be desirable to avoid.

3.2 XSL:FO

One of our design goals is to offer a method that provides a description of how the content of an encoding should be presented. This mechanism must be capable of describing different rendering outputs of the same musical content. XSL:FO (eXtensible Stylesheet Language: Formatting Objects) appears to be useful in this context as it allows a set of rules to be specified for the transformation of the content of an encoded document using a defined page layout. For this purpose, it uses templates which are instantiated as often as necessary during processing, until the entire content is rendered. Using XSL:FO <block> elements for systems, staves, and layers, the general layout of pages containing music notation can be described. XSL:FO can define the margins of the page, padding between and size of systems and staves, and so on.

Despite its initial promise, because XSL:FO is content-agnostic it cannot be used to adjust the layout in response to the content as required by music notation. For instance, in opera or other equally large scores, it is quite common that only the staves of the active voices or instruments be present. This leads to variation in the size and content of systems that is only achievable in XSL:FO by providing a large number of separate templates for each distinct case. Additionally, these templates need to be called explicitly by the user, so that a fully automatic rendering of the content is no longer possible. Furthermore, XSL:FO does not provide mechanisms for capturing the coordinate information necessary for diplomatic transcription of the sources. For these reasons, a template-driven language such as XSL:FO is not suitable for the description of content-dependent layout.

3.3 Scalable Vector Graphics

Instead of using templates for laying out pages, a description of the already laid-out pages could be another possibility. A legitimate approach for this would be to use Scalable Vector Graphics (SVG) markup to describe individual pages. There are already processors that generate SVG output from MEI markup. The problem with using SVG, however, is that it makes it nearly impossible to maintain a connection to the logical content. Because the SVG markup represents the graphical primitives of music notation (lines, note head shapes, etc.) and not the semantic information, changes in the content require the primitives to be recalculated. For example, the SVG markup for the representation of a beam would be made up of filled parallelogram shapes, one for each beam line, with their size and position on the page. Changing the pitch of a single note within the <beam> element in the MEI data would require the size and position of all the graphical components of the beam to be recomputed. Since SVG describes already-processed data, it is inappropriate for storing layout information in a flexible way despite its utility as an output format.

4. THE MEI LAYOUT MODULE

As mentioned above, XSL:FO offers general instructions on how to process data, whereas SVG is more appropriate for already-processed data. The ideal solution for MEI lies between the two: a description of what is in a source, or what should appear on every page in a rendered edition, without duplicating the content and without requiring additional processing of the data.

4.1 General organization

A solution to this problem is to store the layout information in a dedicated sub-tree separate from the musical content. The sub-tree is represented by a <layoutGrp> element within the <music> element. It may contain an arbitrary number of <layout> elements, each of them describing a different visualization of the same musical content.

For example, for the case illustrated in Figure 1 with two sources A and B, the musical content of both sources will be encoded following the traditional approach of MEI, in a single hierarchy with <app> elements for encoding their differences. At the same time, each source will be described further by its own layout sub-tree, if need be in parallel with its related facsimile.

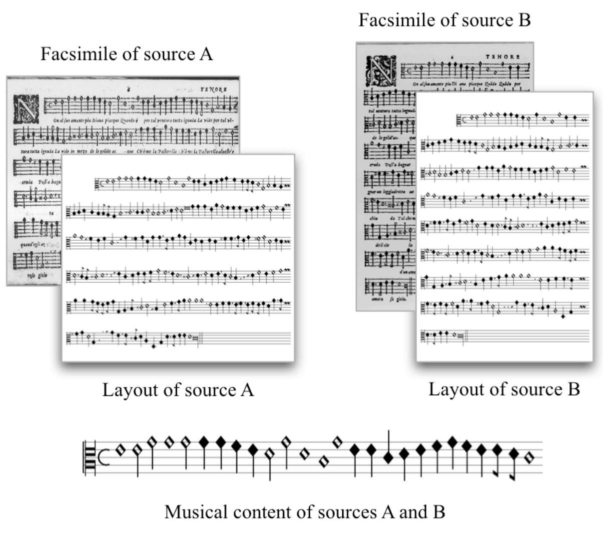


Figure 1. An example of two sources as organized with the layout module in MEI. While they share the same musical content, each layout is described in its own sub-tree.

The <layout> element is expected to have a @type attribute for indicating whether it is intended for "transcription" or "rendering". The <layout> element contains a sequence of <page> elements, each with page-level metadata and nesting <system>, <laidoutStaff> and <laidoutLayer> child elements that can precisely represent how each of them is positioned on the page. The hierarchy can be summarized as follows:

- layoutGrp
  - layout ('transcription' or 'rendering')
    - page
      - system
        - laidoutStaff
          - laidoutLayer

At the lowest level, the <laidoutLayer> element contains a list of <laidoutElement> children. Each <laidoutElement> acts as a generic container that can refer to any element within the corresponding <layer> element in the musical content sub-tree.

The <system>, <laidoutStaff>, <laidoutLayer> and <laidoutElement> elements all have attributes for storing their coordinate position (@lrx, @lry, @ulx and @uly) in "transcriptional" layouts.

4.2 Referencing system

The links that are established between the layout and the elements in the musical content sub-tree are a keystone of the module. Every <page> and <system> in the layout sub-tree is linked to its related <pg> and <sb> elements in the musical content sub-tree. In order to limit the modifications of the musical content sub-tree as much as possible, the links operate deliberately from the layout towards the content, and not the reverse. Each <page> element is expected to have a @pbrefs attribute with the list of XML IDs of <pb> elements in the musical content sub-tree to which it applies, as illustrated in Figure 2. Similarly, <system> elements have a @sbrefs attribute containing a list of <sb> elements. Therefore, the correct insertion of <pb> and <sb> elements is the only change to the logical tree required for this proposal.

For the <laidoutStaff> and <laidoutLayer> elements, the link with the musical content sub-tree is established using a @staff attribute that refers to the @n attribute of a <staff> element in the musical content sub-tree. Finally, <laidoutElement> elements have a @target attribute for referencing elements in the musical content sub-tree.

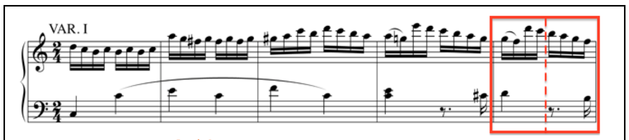
```xml
<music>
  <facsimile source="A">
    <!-- facsimile for source A -->
  </facsimile>
  <facsimile source="B">
    <!-- facsimile for source B -->
  </facsimile>
  <layoutGrp>
    <layout source="A" type="transcription">
      <page pbrefs="pb-A-1">
        <!-- the page layout in source A -->
      </page>
    </layout>
    <layout source="B" type="transcription">
      <page pbrefs="pb-B-1">
        <!-- the page layout in source B -->
      </page>
    </layout>
  </layoutGrp>
  <body>
    <mdiv>
      <score>
        <scoreDef barplace="mensur" key.sig="0">
          <staffGrp>
            <staffDef clef.shape="C" clef.line="3"/>
          </staffGrp>
        </scoreDef>
        <section>
          <staff n="1">
            <layer n="1">
              <pb xml:id="pb-A-1" source="A"/>
              <pb xml:id="pb-B-1" source="B"/>
              <sb xml:id="sb-A-1-1" source="A"/>
              <sb xml:id="sb-B-1-1" source="B"/>
              <!-- the musical content in A and B -->
            </layer>
          </staff>
        </section>
      </score>
    </mdiv>
  </body>
</music>
```

Figure 2. The scaffold encoding for the example given in Figure 1. The <pb> elements in the score are referenced from the <page> elements in the layout.

4.3 Overlapping hierarchies

As we have seen, a fundamental reason why it is advantageous to keep the layout information in a separate sub-tree is because the layout represents a distinct hierarchy that might overlap with the content hierarchy. A typical case is when a system break occurs in the middle of a measure. In such a situation, the same system is indicated in MEI by several <sb> elements, one in every layer where the system break occurs. The encoding in Figure 3 gives an example for such a case with a fictitious system break introduced in the middle of measure number five. In practice, this system break could be present in one or more sources, or it could be desired in a specific rendering. Notice that there are two <sb> elements, one for each layer.

As illustrated in Figure 4, in the corresponding layout sub-tree of this example, the second system references the two new <sb> elements via its @sbrefs attribute.



```
<measure n="5" xml:id="m5">
  <staff n="1" xml:id="m5s1">
    <layer n="1" xml:id="m5s1l1">
      <beam>
        <note xml:id="m5s1e1" pname="g" oct="5" dur="16"/>
        <note xml:id="m5s1e2" pname="f" oct="5" dur="16"/>
        <note xml:id="m5s1e3" pname="d" oct="6" dur="16"/>
        <note xml:id="m5s1e4" pname="c" oct="6" dur="16"/>
      </beam>
      <sb xml:id="sb-X-2-1" source="X"/>
      <beam>
        <note xml:id="m5s1e5" pname="b" oct="5" dur="16"/>
        <note xml:id="m5s1e6" pname="a" oct="5" dur="16"/>
        <note xml:id="m5s1e7" pname="g" oct="5" dur="16"/>
        <note xml:id="m5s1e8" pname="f" oct="5" dur="16"/>
      </beam>
    </layer>
  </staff>
  <staff n="2" xml:id="m5s2">
    <layer n="1" xml:id="m5s2l1">
      <note xml:id="m5s2e1" pname="d" oct="4" dur="4"/>
      <sb xml:id="sb-X-2-2" source="X"/>
      <rest xml:id="m5s2e2" dur="8" dots="1"/>
      <note xml:id="m5s2e3" pname="b" oct="3" dur="16"/>
    </layer>
  </staff>
  <slur staff="1" startid="#m5s1e1" endid="#m5s1e2"/>
</measure>
<sb xml:id="sb-Y-2-1" source=Y"/>
```

Figure 3. The customary encoding of measure 5 with an additional internal <sb>. The beginning of the new system is represented by two <sb> elements in the content sub-tree.

```
<page n="1">
  <system n="1">
    <laidOutStaff staff="1">
      <laidOutLayer>
        <!-- previous measures -->
        <!-- first half of measure 5 -->
        <!-- musical content up to the sb -->
      </laidOutLayer>
    </laidOutStaff>
    <laidOutStaff staff="2">
      <laidOutLayer>
        <!-- previous measures -->
        <!-- first half of measure 5 -->
        <!-- musical content up to the sb -->
      </laidOutLayer>
    </laidOutStaff>
  </system>
  <system n="2" sbrefs="sb-X-2-1 sb-X-2-2">
    <laidOutStaff staff="1">
      <laidOutLayer>
        <!-- second half of measure 5 -->
        <!-- musical content from the sb -->
        <!-- next measures -->
      </laidOutLayer>
    </laidOutStaff>
    <laidOutStaff staff="2">
      <laidOutLayer>
        <!-- second half of measure 5 -->
        <!-- musical content from the sb -->
        <!-- next measures -->
      </laidOutLayer>
    </laidOutStaff>
  </system>
</page>
```

Figure 4. The proposed encoded layout for the example given in Figure 3. The second <system> contains references to the two <sb> elements.

4.4 Content selection

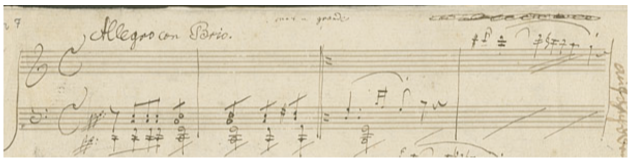
Implicitly, we expect the <laidoutLayer> element to include all elements contained in the corresponding <layer> element of the musical content sub-tree. This means that in the example illustrated by Figures 3 and 4, the content of the measure will be rendered implicitly up to the <sb> element for the system that ends in the middle of measure five and from the <sb> element for the next system. The use of <laidoutElement> for each element is optional for rendering layouts, but it is required for transcription layouts because in that case, we need to be able to store the coordinate positions of the elements.

In some cases, however, a more granular way of selecting content might be required. For example, it might be necessary to hide an element in a specific layout. For this purpose, the <laidoutElement> element has an @ignore attribute.

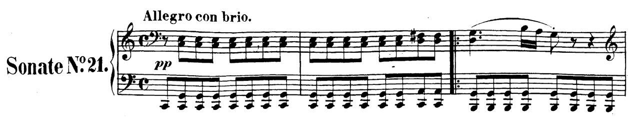
The selection of content can also be performed at the <laidoutStaff> level. For example, a layout for only one staff in the score will have only a single <laidoutStaff> element in each <system> element. Implicitly, all the other staves will not be included in that layout. Similarly, it is possible to change the order of the staves in a specific layout just by modifying the order of the <laidoutStaff> elements.

4.5 Textual and layout variants

In MEI, all variants are traditionally encoded with <app> and <rdg> elements in the music content sub-tree. In some cases, however, variants do not necessarily represent a textual difference between the sources because the musical content represented by the notation is identical. In Figure 5, we can see two examples of the beginning of Beethoven's "Waldstein" sonata. The right hand is written on the lower staff in the autograph and on the upper staff in the edition of Breitkopf & Härtel. Traditionally, this difference could be regarded as a variant in music critical editing even though the musical content is actually the same. However, it would not be possible to "hear" the difference between the two versions.



Autograph manuscript



Leipzig, Breitkopf & Härtel, (Serie 16, Plate B.144)

Figure 5. The beginning of Beethoven's "Waldstein" sonata No. 21. The right hand is written on the lower staff in the manuscript and on the upper one in the edition.

The layout module is designed in such a way that it is possible to encode purely presentational differences between sources at the layout level. In the <laidoutStaff> element, following a content selection method as described above, it is possible to retrieve content from another staff of the musical content sub-tree. In our example, this means that the lower <laidoutStaff> in the layout of the manuscript would pull the content from the first staff, assuming that the music content is encoded as in the edition. Even at its current experimental stage, this practice could represent a significant conceptual change in critical editing. The layout encoding itself becomes the way to represent layout variants, reserving the traditional <app> and <rdg> elements in the musical content sub-tree for textual differences.

5. CONCLUSION AND FUTURE WORK

We believe that the proposed solution is a novel method of encoding MEI documents because it creates a separation between the content of music notation and its possible realizations. Multiple realizations of the same musical content can be stored in parallel, each with its own specific layout information. The layout information can also provide additional functionality. It can be used for specifying how the content appears in an already-existing source, but it can also be used for specifying how the content must be rendered when creating a new edition. The first use is particularly interesting for OMR software applications and for producing image overlays for displaying a transcription directly on top of the facsimile image of the original source. The second use is particularly convenient for storing refined layout information for the parts of an encoded full score. The proposed module lays the basis for a new way of organizing the information contained in an existing MEI encoding.

This approach, however, also raises an interesting question regarding the line between content and presentation in music notation that we hope will receive more attention. There is clearly no fixed border because in music notation layout is a constituent component. The proposed layout module does not attempt to define an absolute boundary, but is intended to be flexible. In practice, the more varied the layout of the sources and the more detailed the layout information recorded, the less it will be desirable to keep layout information in the musical content sub-tree as has been MEI practice so far. With these changes, the musical content sub-tree may become a more abstract representation of the music.

The next step will be to finalize the module in preparation for the next official release of MEI, including preparing guidelines for its usage. We also expect to have to add more features at the <laidoutElement> level depending on thorough testing. For example, it would be logical to expect the module to handle the transposition of instruments when generating parts.

Because creating an encoding with multiple layouts in a general purpose XML editor will be unmanageable, tools that implement at least some of the features of this new module will be a high priority. Currently, the module is being implemented in the Aruspix software application, which will be used for prototyping and providing some more actual examples.

The authors believe that this proposal has great potential to enhance MEI's interoperability, to accelerate its further adoption by the scholarly community, and thus to reinforce its leading role in the digital humanities.

5.1 Availability

The module is available in the incubator of the MEI project.[1] It needs to be compiled with the Roma processor [5].

6. REFERENCES

[1] G. Brüning, K. Henzel, and D. Pravida: "Rationale of multiple encoding in the genetic Faust edition," *Journal of the Text Encoding Initiative*, [Forthcoming].

[2] M. Good and G. Actor: "Using MusicXML for file interchange," *Proceedings of the 3rd International Conference on WEB Delivering of Music,* p. 153, 2003.

[3] F. Jannidis (chair): "An encoding model for genetic editions," http://www.tei-c.org/Activities/Council/Working/tcw19.html 2011.

[4] A. Hankinson, L. Pugin, and I. Fujinaga: "An interchange format for optical music recognition applications*," Proceedings of the 11th International Conference on Music Information Retrieval,* pp. 51–56, 2010.

[5] A. Hankinson, P. Roland, and I. Fujinaga: "The music encoding initiative as a document-encoding framewor*k," Proceedings of the 12th International Conference on Music Information Retrieval,* pp. 293–298, 2011.

[6] L. A. Ludovico: "Key concepts of the IEEE 1599 standard," *Proceedings of the IEEE CS Conference: The Use of Symbols to Represent Music and Multimedia Objects,* p. 15–26, 2008.

[7] D. Oberhelman: "Quijote Interactivo (Interactive Quixote)*,*" *Reference Reviews,* Vol. 25 No. 5, pp. 33–34, 2011.

[8] E. Selfridge-Field: *Beyond MIDI: The Handbook of Musical Codes*, MIT Press, Cambridge MA, 1997.

[1] <http://code.google.com/p/mei-incubator/source/browse/>

A STUDY OF INTONATION IN THREE-PART SINGING USING THE AUTOMATIC MUSIC PERFORMANCE ANALYSIS AND COMPARISON TOOLKIT (AMPACT)

Johanna Devaney
CNMAT, UC Berkeley/
School of Music
The Ohio State University
j@devaney.ca

Michael Mandel
Audience Inc./
College of Engineering
The Ohio State University
mim@mr-pc.org

Ichiro Fujinaga
CIRMMT
Schulich School of Music
McGill University
ich@music.mcgill.ca

ABSTRACT

This paper introduces the Automatic Music Performance Analysis and Comparison Toolkit (AMPACT), is a MATLAB toolkit for accurately aligning monophonic audio to MIDI scores as well as extracting and analyzing timing-, pitch-, and dynamics-related performance data from the aligned recordings. This paper also presents the results of an analysis performed with AMPACT on an experiment studying intonation in three-part singing. The experiment examines the interval size and drift in four ensembles' performances of a short exercise by Benedetti, which was designed to highlight the conflict between Just Intonation tuning and pitch drift.

1. INTRODUCTION

In the early 20th century, psychologist Carl Seashore and his colleagues at the University of Iowa undertook extensive work in performance analysis of singing, examining dynamics, intonation, and vibrato [22]. Their analyses were based on amplitude and frequency information extracted from recordings with phonophotographic apparati. These manual methods were extremely labourious and limited the number of recordings that could be accurately analyzed. Recent developments in digital signal processing have allowed for many of these manual processes to be performed computationally.

The MATLAB-based Automatic Performance Analysis and Comparison Toolkit (AMPACT) collects existing tools and introduces a new MIDI-audio alignment algorithm. The alignment algorithm is able to accurately identify onsets and offsets in the difficult cases of the singing voice and instruments with non-percussive onsets and can be trained to work on recordings of a range of voice types and instruments. The analysis portion of the toolkit includes tools for extracting of various performance parameters related to timing, pitch, and dynamics. AMPACT also includes tools for comparing data across multiple performances. The purpose of the toolkit is to facilitate empirical analysis of musical performance for those without extensive technical training.

This paper also presents an experiment on intonation in three-part singing, which used AMPACT to extract and analyze the intonation data. The experiment uses a short exercise by a music theorist, Benedetti (1530–1590), designed to result in varying amounts of pitch drift when different idealized tunings are applied to it. The exercise was performed numerous times by four different ensembles and the resultant recordings were analyzed in terms of melodic/vertical interval tuning and pitch drift.

2. PREVIOUS WORK

2.1 Automatic Performance Data Extraction

Currently, there are no robust automated methods for estimating note onsets and offsets in the singing voice. Although much work has been done in the area of note onset detection [1], accurate detection of onsets for the singing voice and other instruments without percussive onsets is not a solved problem. Collins used a pitch detector for estimating non-percussive onset detection [3]. He improved on the number of onsets detected within a 100 ms tolerance window over the phase deviation approach described in [1] (58% versus 45%), with comparable false positives (36% versus 37%), Friberg, Schoonderwaldt, and Juslin developed an onset and offset detection algorithm that was evaluated on electric guitar, piano, flute, violin, and saxophone [10]. They reported an onset estimation accuracy of 16 ms and an offset estimation accuracy of 146 ms. Toh, Zhang, and Wang describe a system for automatic onset detection for solo singing voice that accurately predicts 85% of onsets to within 50 ms of the annotated ground truth [23]. These algorithms often require a significant amount of manual correction to obtain sufficient accuracy for performance analysis. Furthermore, offset detection is required for measurements related to duration, intonation, vibrato, and dynamics, but most of these algorithms do not provide it.

2.2 Studies of Intonation in Vocal Ensembles

In the absence of robust automated methods for estimating note onsets and offsets, studies of the singing voice have relied on manual annotation of notes' onsets and offsets. Vurma and Ross studied 13 professional singers' melodic intonation in their performances of short exercises using PRAAT for F_0 analysis [2]. They observed that ascending and descending semitones were smaller than equal temperament and that ascending and descending fifths were larger than equal temperament [25]. Howard studied two *a cappella* SATB quartets and found that they used non-equal temperament with a tendency towards, though not full compliance with, Just-Intonation [13]. He also argued that in pieces with modulation, that since the ensembles used non-equal temperament, pitch drift is necessary for choirs to stay in tune [14]. Howard used electroglottographs to obtain F_0 estimates in order to avoid the complication of polyphonic F_0 estimation.

3. AMPACT

3.1 Overview

AMPACT [1] automatically analyzes performance data from monophonic or quasi-polyphonic recordings. The algorithms included in the toolkit make use of the information available in the score about what notes are expected in the performance and the order in which they will occur. AMPACT provides estimates of note onsets and offsets for tones with non-percussive onsets (e.g., vocalists) that are more robust than existing blind onset detection or alignment algorithms. The analysis portion of the toolkit allows for the extraction of various performance parameters: inter-onset intervals between notes; tempo information; relative dynamic level between notes; mean frequency for each note and interval sizes in cents; and vibrato rate and depth. The statistical tools allow comparisons of different performances of the same musical material or piece. A schematic of the analysis components of AMPACT is shown in Figure 1.

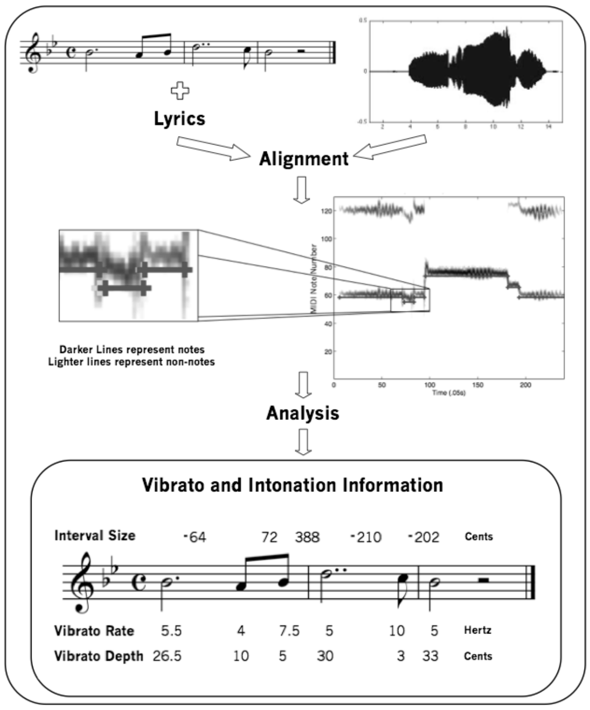

Lyrics
Alignment
MIDI Note Number
Time (.05s)
Darker Lines represent notes
Lighter lines represent non-notes
Analysis
Vibrato and Intonation Information
Interval Size -64 72 388 -210 -202 Cents
Vibrato Rate 5.5 4 7.5 5 10 5 Hertz
Vibrato Depth 26.5 10 5 30 3 33 Cents

Figure 1. Schematic of the Automatic Performance and Analysis Toolkit (AMPACT).

[1] Available for download at www.ampact.org

3.2 MIDI-Audio Alignment

AMPACT uses a MIDI-audio alignment algorithm in order to identify the beginning and ending of all of the notes in a performance. A MIDI version of the score, which is a quantized version of all of the pitch and timing information in the audio, is adjusted so that its timing information corresponds to that of the audio. The algorithm in AMPACT refines the results of an existing Dynamic Time Warping (DTW) approach, described in [18], with a hidden Markov model (HMM). The HMM is trained on the acoustic properties of the melodic line being aligned and, in the case of the singing voice, is guided by the lyrics in the score. The HMM both increases the accuracy of the initial alignment and labels transient and steady-state sections of each note. Identification of the steady-state sections of notes is important because they correspond to the pitched sections.

3.2.1 Hidden Markov Model

This section describes the details of the HMM, with a particular focus on modelling the solo singing voice, namely the observations, states, transition probabilities, and use of a DTW alignment as a prior. The observed variables modelled by the HMM are the square root of periodicity, power, and F_0 estimates provided by the YIN algorithm [5] for each frame. The F_0 estimates from YIN are a somewhat noisy cue, especially for the silence and transient states, but are important because they assist alignment when the note changes under a single vowel.

The three acoustic events are modelled in the HMM: silence, transient, and steady state. In the singing voice, transients occur when a consonant starts or ends a syllable, while vowels produce the steady-state portion of the note. In instruments, the occurrences of transients are influenced by articulation. The transition probabilities were calculated from example recordings of the singing voice. Two versions of the state sequences are implemented. The first allows each state to be visited, shown in Figure 2. The second is modified by the lyrics in the score; transients were only inserted when an unvoiced consonant began or ended a syllable and silences were inserted only at the end of phrases, shown in Figure 3.

The initial DTW alignment is used as a prior to guide the HMM. The use of the DTW alignment obviates the need to encode information about the score in the HMM. By assuming that the DTW alignment is roughly correct, it is not necessary to rely excessively on noisy F_0 estimates in the HMM. This simplifies the design of the HMM and allows the same HMM seed to be used for each note. One issue with this approach is that it cannot adjust the initial alignment by more than one note, so the initial alignment has to be relatively accurate.

The HMM was implemented in MATLAB with Kevin Murphy's HMM Toolbox [17] and uses Alain de Cheveigné's MATLAB implementation of the YIN algorithm [4] as well as Dan Ellis' MATLAB implementation of DTW MIDI-audio alignment [9]. An evaluation of the alignment algorithm is described in [7].

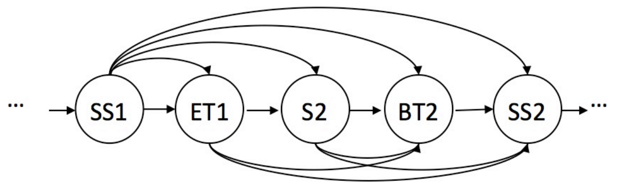


Figure 2. Three-state basic state sequence seed in the HMM: steady state (SS), transient (T), silence (S). The ending transient (ET) and the beginning transient (BT) both have the same observation distribution.

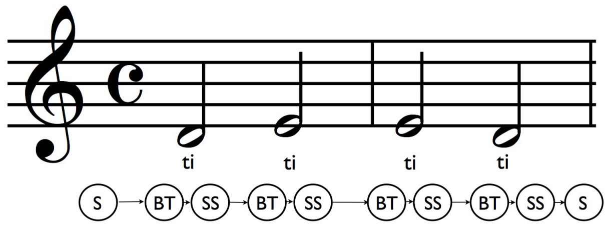


Figure 3. State sequence adapted to sung text.

3.2.2 Performance Data Analysis and Comparison

The alignment algorithm provides information about the note onset and offset times, which AMPACT saves in the MIDI toolbox's note-matrix format [24] from which a MIDI file can be saved. The onset and offset locations also delineate the starting and ending points for calculating pitch- and dynamics-related parameters of each note. Onset and offset information is also saved in as an Audacity-readable label file [15], which allows for manual correction of any alignment errors AMPACT may make.

The YIN algorithm is used for F_0 estimation. One advantage of YIN is that it allows for specification of minimum and maximum expected F_0s, which AMPACT sets according to the note information in the aligned score. The maximum F_0 is set to one whole tone above the corresponding note in the score and the minimum F_0 is set to one whole tone below. This is a very useful feature for recordings that are not strictly monophonic, such as recordings from close miking in ensemble performance.

Perceived pitch is calculated using a weighted mean based on the F_0's rate of change [12]. This mean is calculated by assigning a higher weighting to the frames where the F_0 has a lower rate of change and a lower weighting to those with a higher rate of change. The threshold between high and low rates of change is set at 1.41 octaves/second, based on the vibrato rate and depth values reported in [20; 21]. Vibrato is calculated by finding the dominant frequency of the FFT of the pitch contour. Dynamics are calculated using the implementation in Genesis Acoustics Loudness Toolbox for MATLAB [11] of Glasberg and Moore's model for estimating loudness in time-varying sounds [16]. AMPACT also includes tools for statistical comparison of performances through a wrapper for various t-test, ANOVA, and linear regression functions in MATLAB.

4. INTONATION EXPERIMENT

AMPACT was used to extract and analyze intonation data in an experiment on four three-part vocal ensembles. The ensembles' performances were analyzed with regard to melodic whole-tone tunings, a range of vertical interval tunings, and overall pitch drift. A pre-release version of AMPACT was used by the authors in larger-scale experiment on solo singing in [6].

4.1 Method

4.1.1 Experimental Material

The experimental material is a three-part chord progression written by Bendedetti that was designed to show that singers would not tune Justly with the current sustained note since strict adherence to Just Intonation would result in a significant pitch drift that is not observable in performances of the progression [19]. The progression is built from a seed two-measure progression that is repeated four times. If the singers were to tune in Just Intonation to the sustained note, rather than the bass note, the ensemble would drift up a syntonic comma (21.5 cents) by the end of each seed, resulting in a total upwards drift of 86 cents by the end of the four repetitions. In contrast, if the singers were to tune to the bass in each vertical sonority, with D, A, or G in the bass, there should be no drift. The calculations for both tuning scenarios are shown in Figure 4.

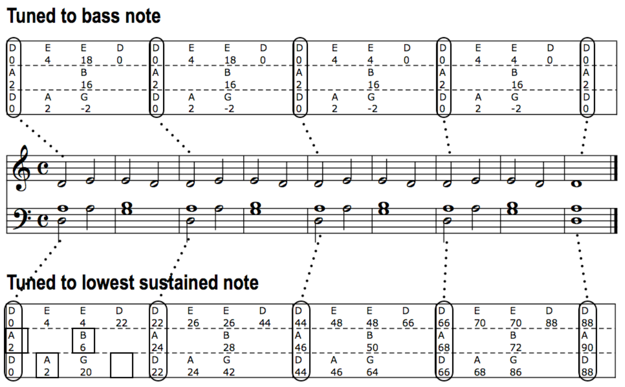


Figure 4. Theoretical tuning for Benedetti progression used as experimental material. The numbers in the tables at the top and bottom of the figure indicate the tuning in relation to the starting pitch in the bass.

4.1.2 Participants

Four three-part ensembles participated in this experiment. Ensemble 1 served as a pilot study with semi-professional alto, tenor, and bass singers who performed without a conductor. The ensemble had an average age of 26 years (*SD* = 3.6), with an average of 6.5 years of private voice lessons (*SD* = 4.5) and 6.5 years of regular practice (*SD* = 2.5). Ensembles 2, 3, and 4 consisted of professional singers who regularly sang together with the conductor used in the experiment. These ensembles had an average age of 42 years (*SD* = 9), an average of 7.75 years of private voice lessons (*SD* = 0.5) and 24 years of regular practice (*SD* = 10). The singers in both ensembles were experienced in singing *a cappella* Renaissance music and were asked to sing with their normal tuning.

Ensembles 2 and 4 consisted of alto, tenor, and bass singers while Ensemble 3 consisted of soprano, alto, and tenor. Ensembles 1 and 2 were recorded in a 4.85m x 4.50m x 3.30m lab with low-noise, minimal reflections, and short reverberation time. The singers were miked with cardioid headband mics (DPA 4088-F). The microphones were run through an RME Micstasy 8-channel microphone preamplifier and an RME Madi Bridge into a Mac Pro computer for multi-track recording. Ensembles 3 and 4 were recorded on the altar of St. Mathias‘ Church, a church in Montreal dating from 1912 with wooden floors, limestone walls, and seating for 350 people. As with the lab environment, the singers were miked with cardioid headband mics, although a portable Zaxcom Deva 16 digital recorder was used for the rest of the recording setup.

4.2 Results

4.2.1 Interval Size

The mean and standard deviation of the interval sizes for the melodic whole tones are shown in Table 1. The majority of the means were within one standard deviation of the equal tempered 200 cent tuning. The main exception to this was Ensemble 1 where the whole tone tended to be smaller. In particular, the middle voice whole tones, which were closer to the 182 cent Minor Just Intonation whole tone. Just over half of the singers' whole tones (12/20) were within one standard deviation of the Pythagorean/Major Just Intonation (204 cents) whole tone.

Vertical intervals were calculated for each half-measure between all of the voices: lowest voice to middle voice, lowest voice to upper voice, and middle voice to upper voice. The onset and offset times for the vertical intervals were determined by the upper voice. Overall, there were 51 vertical intervals in each rendition: 4 Minor Thirds (m3), 8 Major Thirds (M3), 9 Perfect Fourths (P4), 17 Perfect Fifths (P5), 4 Major Sixths (M6), and 9 Perfect Octaves (P8). The means and standard deviations for each type of vertical interval across all of the singers in each ensemble are shown in Table 2. There was a wide range in the mean values for both the vertical m3 and M3, specifically 300–322 cents for the m3 and 375–413 cents for the M3. When the standard deviations are taken into account, the m3 encompassed the Pythagorean (294 cent), equal tempered (300 cents), and Just Intonation (316 cents) tunings. Likewise, the M3 range encompassed the Just Intonation (386 cents), equal tempered (400 cents), and Pythagorean (408 cents) tunings. The range of the means for the M6 encompassed only the equal tempered tuning (900 cents) since the means were all larger than the Just Intonation tuning (884 cents) and marginally smaller than the Pythagorean (905 cents). The tunings for the P4 (498 cents), P5 (702 cents), and P8 (1200 cents) are common to both the Pythagorean and Just Intonation systems and are close to the values for equal temperament (500, 700, and 1200 cents, respectively); the ranges for these intervals encompassed all of these tunings.

		Voices				
		Top		**Middle**		**Bottom**
Ensemble		Up	Down	Up	Down	Down
1	**Mean**	**199**	**195**	**185**	**183**	**191**
	SD	*6*	*4*	*6*	*10*	*7*
	N	12	12	12	12	12
2	**Mean**	**192**	**191**	**207**	**210**	**189**
	SD	*6*	*16*	*12*	*13*	*6*
	N	16	16	16	16	16
3	**Mean**	**199**	**199**	**199**	**196**	**198**
	SD	*9*	*10*	*8*	*8*	*11*
	N	16	16	16	16	16
4	**Mean**	**189**	**194**	**196**	**196**	**195**
	SD	*13*	*8*	*10*	*13*	*13*
	N	20	20	20	20	20

Table 1. Mean, standard deviation, and number of instances of the ascending and descending melodic whole tone sizes for all ensembles, broken down by voice.

		Vertical Interval Types					
Ensemble		**m3**	**M3**	**P4**	**P5**	**M6**	**P8**
1	**Mean**	**322**	**376**	**509**	**701**	**893**	**1201**
	SD	*7*	*9*	*10*	*6*	*6*	*7*
	N	12	24	27	51	12	27
2	**Mean**	**300**	**413**	**497**	**705**	**903**	**1206**
	SD	*12*	*11*	*17*	*14*	*15*	*12*
	N	16	32	36	68	16	36
3	**Mean**	**307**	**397**	**507**	**704**	**904**	**1209**
	SD	*8*	*11*	*12*	*11*	*11*	*9*
	N	16	32	36	68	16	36
4	**Mean**	**301**	**406**	**493**	**702**	**896**	**1202**
	SD	*14*	*15*	*13*	*12*	*10*	*12*
	N	20	40	45	85	20	45

Table 2. Mean, standard deviation, and number of instances of the vertical interval sizes between the three voices across all renditions by all of the ensembles.

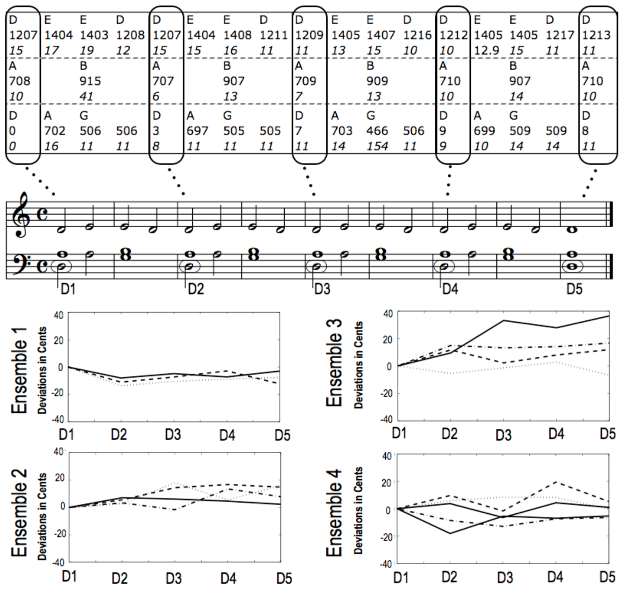
An ANOVA analysis for each ensemble was run on the melodic interval data with whole tone size as the dependant variable and direction and singer identity as independent variables. In Ensemble 1, there was no significant effect for direction, though the middle singer's whole tones were significantly smaller than the other two singers, F (2, 56) = 24.59, p < 0.001. Ensemble 2 was similar, with no effect for direction and a significant effect for the middle singer, though in this case the middle singer's whole tones were significantly larger than the other two singers, F (2, 75) = 24.52, p < 0.001. There were no significant effects for direction or singer identity in Ensembles 3 and 4. A separate ANOVA was run with direction and group identity. There was no overall effect for direction, but Ensemble 1's whole tones were significantly smaller on average than Ensembles 2 and 3, F (3, 311) = 6.96, p < 0.001.

Separate ANOVAs were also run on each vertical interval to test for group effects. Ensemble 1's m3 intervals were significantly larger on average than the other three ensembles, F (3, 59) = 11.93, < 0.001, so much so that their mean overshot the 316 cent Just Intonation value. In contrast,, Ensemble 1's M3 intervals were significantly smaller on average than the other ensembles', F (3, 127) = 50.31, p < 0.001 and were so small that they undershot the 386 cent Just Intonation value. For the P4, Ensembles 1 and 3 were significantly larger than Ensembles 2 and 4, F (3, 143) = 11.75, p < 0.001. There were no significant effects between the ensembles for the P5, M6, or P8.

4.2.2 Pitch Drift

In order to assess whether the ensembles drifted in the manner predicted by Benedetti, the perceived pitch estimates in cents were calculated for each note in each rendition in relation to the rendition's opening D in the bass. The table at the top of Figure 5 shows the means and standard deviations for each note across all of the ensembles. Overall there was a slight drift upwards of 8 cents in the lower voice, 10 cents in the middle voice, and 13 cents in the upper voice. This drift is much smaller than the one predicted by the calculations in the lower chart in Figure 5, suggesting that the singers were tuning to the bass note rather than the lowest sustained note.

Figure 5 also shows the drift in the bass voice for each ensemble through plots of the perceived pitch (relative to the starting note) of the D at the start of each seed progression. Ensemble 1 was the most consistent with itself across performances, exhibiting only a small amount of drift from the starting pitch. Ensembles 2 and 3 both tended to drift upwards with Ensemble 3 showing a greater amount of variability in the amount of drift. Ensemble 4 had little drift overall but showed a large amount of variation within each performance.



D 1207 *15*	E 1404 *17*	E 1403 *19*	D 1208 *12*	D 1207 *15*	E 1404 *15*	E 1408 *16*	D 1211 *11*	D 1209 *11*	E 1405 *13*	E 1407 *15*	D 1216 *10*	D 1212 *10*	E 1405 12.9	E 1405 *15*	D 1217 *11*	D 1213 *11*
A 708 *10*		B 915 *41*		A 707 *6*		B 907 *13*		A 709 *7*		B 909 *13*		A 710 *10*		B 907 *14*		A 710 *10*
D 0 *0*	A 702 *16*	G 506 *11*	506 *11*	D 3 *8*	A 697 *11*	G 505 *11*	505 *11*	D 7 *11*	A 703 *14*	G 466 *154*	506 *11*	D 9 *9*	A 699 *10*	G 509 *14*	509 *14*	D 8 *11*

Figure 5. Summary of the amount of drift in each ensemble's renditions of the Benedetti chord progression. The lines in the each plot link the perceived pitch estimates for the notes D1-D5 in each rendition.

4.3 Discussion

Overall the singers tended towards equal temperament. The vast majority of the means of the melodic and vertical intervals were within one standard deviation of equal temperament. The melodic intervals that were not within one standard deviation of equal temperament were much smaller, falling instead within one standard deviation of the 182 cent minor Just Intonation semitone. Likewise, most of the outlying vertical intervals fell within one standard deviation of non-equal temperament idealized tunings (either Just Intonation of Pytheagorean): Ensemble 1's m3 mean was within one standard deviation of the 316 cent Just Intonation tuning; Ensemble 1's M3 mean was within one standard deviation of the 386 cent Just Intonation tuning; Ensemble 2's M3 mean was within one standard deviation of the 408 cent M3. The ANOVA analysis revealed some significant effects for singer and group identity for some of the ensembles. The lack of a significant effect for direction in the ANOVA analysis of the whole tone tuning mirrors our earlier findings for professional solo singers [8].

The ensembles did drift up slightly on average, but not to the extent they would have if the singers were tuning in Just Intonation to the lowest sustained note. This is not surprising, as such a rapid drift, 88 cents over eight measures, is highly unlikely since it implies that the singers were not retaining their starting pitch as a reference only a few tens of seconds after it was sung.

5. CONCLUSIONS

This paper presented AMPACT, a MATLAB toolkit for automatically extracting, analyzing, and comparing performance data from monophonic recordings for which a score is available. The alignment algorithm in AMPACT works well on sounds without a clearly defined onset, making it useful for the singing voice and instruments with non-percussive onsets. This paper also demonstrates the use of AMPACT in extracting and analyzing the data for an experiment on vocal intonation. The experiment with four three-part ensembles found that the singers tended toward equal temperament and did not exhibit a large amount of drift in an exercise by Renaissance theorist Benedetti. The detailed analysis of singing intonation in this study was facilitated by the automated nature of AMPACT, not only in terms of time savings but also in consistency of data extraction.

6. ACKNOWLEDGMENTS

The authors would like to thank Dan Ellis for his assistance in developing the alignment algorithm in AMPACT. We would also like to thank Jonathan Wild and Peter Schubert for their contributions to the experiment. We would also to acknowledge the contributions of the personal at the Centre for Research in Music Media and Technology (CIRMMT) and Center for New Music and Audio Technologies (CNMAT) to various aspects of this project, particularly David Wessel. This work was supported by funding from the Fonds de recherche sur la société et la culture (FQRSC) and Social Sciences and Humanities Research Council of Canada (SSHRC).

7. REFERENCES

[1] Bello, J. P., L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler. 2005. A tutorial on onset detection in music signals. *TASLP* 13 (5): 1035–47.

[2] Boersma, P. 2001. PRAAT, a system for doing phonetics by computer. *Glot International* 5 (9/10): 341–5.

[3] Collins, N. 2005. Using a pitch detector for onset detection. In *Proceedings of ISMIR*, 100–6.

[4] de Cheveigné, A. 2002. *YIN MATLAB implementation.* http://audition.ens.fr/adc/sw/yin.zip.

[5] de Cheveigné, A., and H. Kawahara. 2002. YIN, a fundamental frequency estimator for speech and music. *JASA* 111 (4): 1917–30.

[6] Devaney, J., M. Mandel, I., D. P. W. Ellis, and I. Fujinaga. 2011. Automatically extracting performance data from recordings of trained singers. *Pyschomusicology* 21 (1–2): 108–36.

[7] Devaney, J., M. I. Mandel, and D. P. W. Ellis. 2009. Improving MIDI-audio alignment with acoustic features. In *Proceedings of WASPAA*, 45–8.

[8] Devaney, J., J. Wild, and I. Fujinaga. 2011 Intonation in solo vocal performance: A study of semitone and whole tone tuning in undergraduate and professional sopranos. In *Proceedings of the International Symposium on Performance Science*, 219–24.

[9] Ellis, D. P. W. 2008. *Aligning MIDI scores to music audio*. http://www.ee.columbia.edu/~dpwe/resources/matlab/alignmidiwav/.

[10] Friberg, A., E. Schoonderwaldt, and P. N. Juslin. 2007. CUEX: An algorithm for extracting expressive tone variables from audio recordings. *Acta Acustica united with Acustica* 93: 411–20.

[11] *Genesis Acoustics Loudness Toolbox for Matlab*. 2010. http://www.genesis-acoustics.com/.

[12] Gockel, H., B. C. J. Moore, and R. P. Carlyon. 2001. Influence of rate of change of frequency on the overall pitch of frequency-modulated tones. *JASA* 109 (2): 701–12.

[13] Howard, D. M. 2007. Equal or non-equal temperament in a cappella SATB singing. *Logopedics Phoniatrics Vocology* 32: 87–94.

[14] Howard, D. M. 2007. Intonation drift in a capella soprano, alto, tenor, bass quartet singing with key modulation. *Journal of Voice* 21 (3): 300–15.

[15] Mazzoni, D., and R. Dannenberg. 2000. *Audacity*. http://audacity.sourceforge.net/.

[16] Moore, B. C. J., and B. R. Glasberg. 2010. The role of temporal fine structure in harmonic segregation through mistuning. *JASA* 127 (1): 5–8.

[17] Murphy, K. 1998. *Hidden Markov Model (HMM) Toolbox for Matlab*. http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html.

[18] Orio, N., and D. Schwarz. 2001. Alignment of monophonic and polyphonic music to a score. In *Proceedings of ICMC*, 155–8.

[19] Palisca, C. V. 1994. *Studies in the History of Italian Music and Music Theory*. Oxford, UK: Oxford University Press.

[20] Prame, E. 1994. Measurements of the vibrato rate of ten singers. *JASA* 96 (4): 1979–84.

[21] Prame, E. 1997. Vibrato extent and intonation in professional western lyric singing. *JASA* 102 (1): 616–21.

[22] Seashore, C. 1938. *Psychology of Music*. Iowa City, IA: University of Iowa Press. Original edition, New York, NY: Dover Publications.

[23] Toh, C. C., B. Zhang, and Y. Wang. 2008. Multiple-feature fusion based on onset detection for solo singing voice. In *Proceedings of ISMIR*, 515–20.

[24] Toiviainen, P., and T. Eerola. 2004. *MIDI Toolbox: MATLAB Tools for Music Research*. http://www.jyu.fi/musica/miditoolbox/

[25] Vurma, A., and J. Ross. 2006. Production and perception of musical intervals. *Music Perception* 23 (4): 331–44.

INFLUENCE OF PEAK SELECTION METHODS ON ONSET DETECTION

Carlos Rosão
ISCTE-IUL
L2F/INESC-ID Lisboa
rosao@l2f.inesc-id.pt

Ricardo Ribeiro
ISCTE-IUL
L2F/INESC-ID Lisboa
rdmr@l2f.inesc-id.pt

David Martins de Matos
IST/UTL
L2F/INESC-ID Lisboa
david@l2f.inesc-id.pt

ABSTRACT

Finding the starting time of musical notes in an audio signal, that is, to perform onset detection, is an important task as this information can be used as the basis for high-level musical processing tasks. Many different methods exist to perform onset detection. However their results depend on a Peak Selection step that makes the decision whether an onset is present at some point in time. In this paper we review a number of different Peak Selection methods and compare their influence in the performance of different onset detection methods and on 4 distinct onset classes. Our results show that the post-processing method used deeply influences both positively and negatively the results obtained.

1. INTRODUCTION

In general, music is composed by sounds generated simultaneously by several musical instruments of different kinds [7]. Thus, one can consider the notes played by these musical instruments as the basic unit or syllable for a musical signal [7]. These notes are what allows us humans to clap our hands when listening to a music or whistle/hum the melody of a familiar song [5].

There has been intense research in this area for quite some time, mostly because the information about the starting moments of musical notes can be used as a first step for high-level music processing techniques, such as Chord Estimation, Harmonic Description or Music Genre Classification.

In this paper we are mainly interested in studying how the post-processing part of the onset detection methods, that is, the Peak Selection part in Fig. 1, responsible for deciding whether a point in time is an onset, influences the results obtained. This can be of great help in case one wants to know the more appropriate Onset Detection method – and consequently Peak Selection Method – to use in a particular application.

In the next section, we will present the most common onset detection methods, while in Section 3 we introduce the Peak Selection Methods used. Section 4 describes our experiments and discusses the obtained results. The paper ends with final remarks and future work.

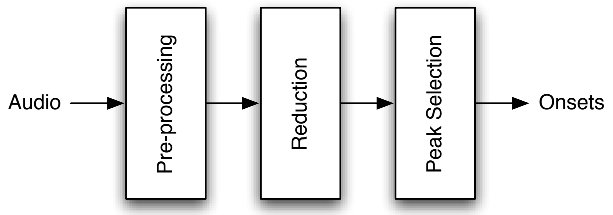


Figure 1. Traditional onset detection work-flow [4].

2. ONSET DETECTION METHODS

Many Onset Detection Methods have been proposed during the years and most of them follow the general scheme in Fig. 1 which comprises the following steps [1, 4, 5]:

- Pre-processing of the signal in order to highlight its most important properties [1, 4].
- Creation of a Onset Detection Function, also called Onset Strength Signal (OSS)[1], that is, a function whose peaks should correspond to onset times [2].
- Peak Selection, in order to decide which peaks in the Onset Detection Function are onsets.

Next, we briefly review the Onset Detection Functions later used to assess the influence of the Peak Selection part of detecting onsets. For a more general overview of onset Onset Detection Functions, check, for instance, [12] and for a thorough comparison of the performances of the different OSS check, for instance [1] or [13].

In order to detect variations in the properties of the audio signal [2], one can create an OSS by lowering the sample rate of the signal without losing relevant information. This a process called Reduction [1].

All the OSS we will explore are based on Spectral Features of the signal. In order to change from the time-domain to the spectral-domain representation of the audio, we make use of the Short-time Fourier Transform (STFT).

High Frequency Content Making use of the fact that typically, when compared to other audio sources, an onset has relative high energy in higher frequencies [1,

[1] In this paper we use the terms Onset Detection Function and OSS interchangeably.

11], it is possible to create a Onset Detection Function that weights each STFT bin proportionally to its frequency. This function is called High Frequency Content (HFC).

Spectral Difference Another possibility to define an OSS is to create a function that measures the variation of magnitude between frequency bins [2, 4]. This type of OSS is called Spectral Difference or Spectral Flux (SF).

Phase Deviation One can also look for onsets by searching for irregularities in the phase of consecutive frequency bins [2], and that is what does the Phase Deviation (PD) Onset Detection Function.

It is possible to improve this function by weighting – Weighted Phase Deviation (WPD) – and normalization [2].

Complex Domain It is possible to combine information from the both the energy and phase of the spectrum to create a Complex Domain (CD) function [3]. This kind of function looks for irregularities in the steady-state of the signal [2] .

A possible improvement for this method is to rectify the function so that it ignores offsets and focuses on onsets [2] – Rectified Complex Domain (RCD).

3. PEAK SELECTION METHODS

A function created with any of the methods introduced in Section 2 will typically show well-localized maxima in positions corresponding to onset times [1]. To extract the onset times from the OSS, Peak Selection methods are used that typically include the steps: Post-processing, Thresholding and Peak-picking.

3.1 Post-processing

Post-processing aims at making the Onset Detection Function uniform so that the processes of thresholding and peak-picking will be easier. This process of increasing the uniformity of the Onset Detection Function typically makes use of normalization methods and filters.

The normalization typically works in one of two ways [2, 5]: (i) Subtract the average value of the function from each value, so that the average will be zero and then divide by the maximum value so that the function will be in the interval [-1,1]; (ii) Subtract the average value of the function from each value and then divide by the maximum absolute deviation, so that the average will be 0 and the standard deviation 1.

The filters used are typically low-pass filters [1, 2, 5], which, in general, select low frequencies up to the cut-off frequency (f_c) and attenuate frequencies higher than f_c [14] and can be defined as

$$y_i = \alpha x_i + (1 - \alpha) y_{i-1} \qquad (1)$$

where α is the smoothing factor.

3.2 Thresholding

In order to separate event-related from non-event-related peaks in the post-processed Onset Detection Function, d, it is common to build a threshold [1].

One can define a constant threshold [8], δ, although this type of threshold is not appropriate, because it does not consider the great dynamics common in a musical signal, leading to weak results [1]. It is much more common to use adaptive thresholds [1, 2, 5]. An adaptive threshold can be constructed in several ways. The best way to overcome problems when facing music pieces with great dynamic change is to build a threshold function based on the local mean (Eq. 2) or local median (Eq. 3) of the Onset Detection Function, d [6].

$$\tilde{\delta}(n) = \delta + \lambda\,\text{mean}(|d(n-M)|, ..., |d(n+M)|) \qquad (2)$$

$$\tilde{\delta}(n) = \delta + \lambda\,\text{median}(|d(n-M)|, ..., |d(n+M)|) \qquad (3)$$

Where λ and δ are positive constants, that can be tweaked, and M is the size of a window around each of the points of the Onset Detection Function.

3.3 Peak-picking

After building a threshold function, one must choose which values of the Onset Detection Function that are larger than the threshold correspond to onsets.

One can consider every value greater than the threshold ($w = 0$ in the following equation) as an onset, or one can add the condition that it must be a local maximum ($w > 0$) [2,4] (where w is a tweakable parameter that corresponds to the size of a window around the value):

$$o(n) = \begin{cases} 1 & \text{if } d(n) > \tilde{\delta}(n) \\ & \text{and} \quad d(n-w) \leq d(n) \leq d(n+w), \\ \\ 0 & \text{otherwise.} \end{cases} \qquad (4)$$

4. RESULTS

In this section we will present the evaluation methods and dataset used as well as discuss the results obtained.

4.1 Evaluation Methods

When evaluating onset detection methods, the most common criterion is the F-measure, that is defined in Eq. 5.

$$\text{F-measure} = \frac{2}{\frac{1}{P} + \frac{1}{R}} = \frac{2PR}{P+R} \qquad (5)$$

With Precision, P, and Recall, R, which can be computed in terms of the False Positive (FP), True Positive (TP) and False Negative (FN). In the particular case of onset detection, one can interpret the TP as the correctly detected onsets, the FP as falsely detected onsets and the FN as onsets that were not detected.

The Precision, that is, the fraction of retrieved instances that are relevant is defined in Eq. 6.

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (6)$$

On the other hand, the Recall, that is, the fraction of relevant instances that are retrieved, is obtained by Eq. 7.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

The Mirex Onset Detection Task specifications [9], and most of the papers in this area, consider onsets detected as TP if they are in a window of $50ms$ around the annotated onset. On the other hand, if more than one detection falls inside the same tolerance window, only one is counted as TP, the others are considered as FP. When a detection is inside the tolerance window of two onset annotations, one TP and one FN are counted. We will evaluate our results according to these specifications.

4.2 Dataset

To run our experiments, we used a dataset built by Bello et al. for [1], referred to as the Bello Dataset.

The Bello Dataset is a hand-labelled and annotated dataset first proposed in [1] and used in several papers, such as [2, 5]. It contains commercial and non-commercial recordings, covering a variety of musical styles and instrumentations, totalling 23 songs and 1065 onsets [1]. The songs are available in WAV format (sample rate 22.050 kHz, mono, 16 bit) and their onset positions (in seconds) in text format.

The recordings of the dataset can be divided in 4 classes, according to the characteristics of their onsets: Complex Mixture (Mix), Pitched Non-Percussive (PNP), Pitched Percussive (PP), and Non-Pitched Percussive (NPP) as shown in Table 1.

	No. Songs	No. Onsets
Mix	7	271
PNP	1	93
PP	9	489
NPP	6	212
Total	23	1065

Table 1. Bello Dataset Structure

One can think of Mix onsets as onsets produced by any polyphonic music where several instruments are playing together, something that happens, for instance, in a rock or pop song. The NPP onsets are the ones typically produced by percussion instruments such as drums or cymbals, while the PP onsets are those that have a percussive characteristic but, nonetheless, still maintain a well defined pitch; this type of onsets appears, for instance, when a piano is playing. Finally, the PNP onsets are those that do not have percussive characteristics and have a very well defined pitch; this category contains onsets from instruments such as bowed strings or wind instruments.

4.3 Experiments

In order to assess the influence of Peak Selection Methods on the results of onset detection, different simulations were run each with a particular Peak Selection Method. These methods were selected because they have been used in recent work [1, 2, 5].

We used the following abbreviations to name the used Peak Selection Methods:

norm Normalize the Onset Detection Function by dividing by the absolute maximum and subtracting the average value, so that the average will be zero.

stdev Normalize the Onset Detection Function by dividing by the maximum standard deviation and subtracting the average value, so that the average will be zero.

mean Create a running mean threshold (Eq. 2).

median Create a running median threshold (Eq. 3).

filter Before normalization, smooth the Onset Detection Function by applying a simple low-pass filter (Eq. 1).

no-filter Do not apply the low-pass filter, that is, do not use smoothing.

local-max Consider as onsets every value in the Onset Detection Function that is larger than zero, larger than the threshold and is a local maximum in a window of 3 samples around it. I.e., use $w = 3$ in Eq. 4.

no-local-max Consider as onset every value greater than the threshold. In other words, use $w = 0$ in Eq. 4.

	A	B	C	D	E
norm	×	×		×	×
stdev			×		
mean		×			
median	×		×	×	×
filter				×	
local-max	×	×	×	×	

Table 4. Components of the Peak Selection Methods A, B, C, D and E.

First we run our experiments with the Peak Selection Method median-norm-no-filter-local-max (A), then we replaced the running mean threshold with a running average threshold with parameter $M = 10$ by running the experiments with the Peak Selection Method mean-norm-no-filter-local-max (B). After that, in order to assess the influence of the type of normalization, we ran the experiments by replacing the norm type of normalization with the stdev type of normalization, that is, using the Peak Selection Method median-stdev-no-filter-local-max (C).

We also tested the influence of a smoothing step before the Peak Selection – with the use of a simple low-pass filter – by running the experiments with the median-norm-filter-local-max (D) Peak Selection Method.

Finally, to test the peak picking algorithm's influence, we ran the experiments without the local maximum condition, that is we used the median-norm-no-filter-no-local-max (E) Peak Selection Method.

	A			B			C			D			E		
OSS	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R
HFC	0.921	0.922	0.920	0.922	0.957	0.901	0.921	0.922	0.920	0.823	0.913	0.766	0.622	0.525	0.798
SF	0.931	0.946	0.926	0.943	0.957	0.937	0.934	0.946	0.932	**0.939**	**0.953**	**0.933**	**0.782**	**0.709**	**0.903**
PD	0.652	0.573	0.819	0.650	0.571	0.819	0.652	0.573	0.819	0.628	0.586	0.749	0.520	0.417	0.893
WPD	0.916	0.959	0.882	0.922	0.933	0.918	0.914	0.945	0.891	0.828	0.900	0.778	0.603	0.507	0.816
CD	**0.947**	**0.978**	**0.923**	**0.946**	**0.987**	**0.913**	**0.943**	**0.970**	**0.923**	0.872	0.931	0.835	0.583	0.482	0.820
RCD	0.933	0.977	0.903	0.933	0.966	0.913	0.936	0.977	0.908	0.909	0.919	0.904	0.419	0.298	0.824

Table 2. Results with P, Precision, F, F-measure and R, Recall, for NPP onsets in the Bello Dataset using all the 5 Peak Selection methods (A, B, C, D, E).

	A			B			C			D			E		
OSS	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R
HFC	0.838	0.846	0.830	0.848	0.829	0.867	0.842	0.846	0.839	0.576	0.607	0.547	0.523	0.437	0.651
SF	**0.961**	**0.968**	**0.954**	**0.965**	**0.978**	**0.953**	**0.961**	**0.969**	**0.954**	**0.876**	**0.878**	**0.874**	**0.893**	**0.868**	**0.921**
PD	0.497	0.410	0.734	0.488	0.414	0.740	0.388	0.278	0.823	0.529	0.323	0.823	0.368	0.256	0.732
WPD	0.810	0.796	0.826	0.811	0.793	0.830	0.811	0.793	0.830	0.470	0.545	0.414	0.666	0.641	0.692
CD	0.883	0.892	0.874	0.899	0.876	0.923	0.903	0.883	0.923	0.441	0.547	0.370	0.543	0.488	0.611
RCD	0.882	0.880	0.883	0.891	0.863	0.920	0.881	0.823	0.947	0.599	0.574	0.625	0.734	0.664	0.820

Table 3. Results with P, Precision, F, F-measure and R, Recall, for PP onsets in the Bello Dataset using all the 5 Peak Selection methods (A, B, C, D, E).

4.4 Discussion

While running the experiments, we fixed the window size of each STFT at 1024 samples (that is 46.4 ms in these 22.05 kHz sampled signals) with a hop size of 50%. The parameters δ and λ were tweaked, in order to obtain the values that maximize the f-measure.

The results obtained by running our experiments with all the Peak Selection Methods described in the previous section are shown in Tables 2, 3, 5 and 6.

In order to compare the methods, we consider as base the results with the Peak Selection Method A and compare all others with this one. First, we will analyse the influence of the Peak Selection Methods on the results obtained for the different onset classes, next, we will analyse the influence of the Peak Selection Methods on each OSS, and, finally, we will make a global balance about the significance of the compared results of the different Peak Selection Methods.

4.4.1 Onset Classes

The differences between running the experiments by using a running-median threshold – Peak Selection Method A – or a running-mean threshold – Peak Selection Method B – have mixed behaviours according to the onset classes. In the NPP and PP classes, the mean gives slightly better results (1pp[2] better) than the median, while it improves for certain OSS it gives worse results for others, but just 1-2pp differences for better or for worse. On the other hand, the running-mean threshold is prone to give worse results by around 2-3pp in the Mix onset class.

To use a normalization based on the maximum standard deviation – Peak Selection Method C – when comparing to a normalization based on the maximum absolute value – Peak Selection Method A – gives mixed behaviours according to the onset classes. In the NPP and PNP onset classes, the results remain almost the same (the changes are less than 1pp) while for the PP the relevant changes are a decrease of around 10pp for the PD function and a performance increase of about 3pp for the HFC and CD functions. When it comes to the Mix onset class, the results for the HFC and PD functions remain just the same, but the other OSS functions have worse f-measure (2-3pp).

When smoothing the Onset Detection Function – Peak Selection Method D – the results become quite different. For the NPP onset class, the SF becomes slightly better (less than 1pp), while for all the other OSS, the results become poorer from 3 to 10pp. In the case of PP onsets, the filter improves about 3pp on the PD function, although it decreases the results significantly (10 to 40pp) for all other OSS. In the PNP onset classes, the behaviour is mixed according to the onset class. We have a positive boost of around 20pp for the PD OSS while for all the other functions the results get worse from 4pp to 30pp. For the Mix onset class, the results get considerably worse for all the OSS.

Finally, when dropping the local maximum condition in the peak picking algorithm – Peak Selection Method E – the results become quite different, but there is a general trend easy to spot: the results get worse for every OSS without exception. In the NPP the results are 15 to 50pp worse, while for the PP the results are 13 to 25pp worse. For PNP onsets, in general, the results are around 30pp worse while for Mix onsets the results vary from 10pp to 30pp worse.

4.4.2 OSS

Moving from running-median threshold to running-mean threshold – Peak Selection Method B – gives, in general, slight improvements for the HFC OSS in all the onset classes, while for the SF OSS the behaviour is mixed. It improves slightly the SF in PP, NPP and PNP onset classes, while decreasing the performance in the Mix class, although these improvements and decreases are very small (1-3pp). We have similar behaviour for the WPD, CD and RCD Onset Detection Functions, with the increases and decreases not going beyond 3pp. In the case of the PD OSS, the re-

[2] pp – percentage point.

	A			B			C			D			E		
OSS	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R
HFC	0.553	0.519	0.591	0.552	0.519	0.591	0.553	0.519	0.591	0.405	0.471	0.355	0.358	0.242	0.688
SF	**0.911**	**0.888**	**0.935**	**0.915**	**0.858**	**0.978**	**0.914**	**0.914**	**0.914**	**0.869**	**0.847**	**0.892**	**0.696**	**0.595**	**0.839**
PD	0.615	0.479	0.860	0.615	0.479	0.860	0.615	0.479	0.860	0.803	0.770	0.839	0.184	0.101	1
WPD	0.660	0.602	0.731	0.670	0.626	0.720	0.670	0.626	0.720	0.465	0.506	0.430	0.463	0.343	0.710
CD	0.684	0.650	0.720	0.680	0.644	0.720	0.677	0.657	0.699	0.388	0.444	0.344	0.409	0.295	0.667
RCD	0.808	0.745	0.882	0.808	0.745	0.882	0.808	0.745	0.882	0.503	0.500	0.505	0.562	0.425	0.828

Table 5. Results with P, Precision, F, F-measure and R, Recall, for PNP onsets in the Bello Dataset using all the 5 Peak Selection methods (A, B, C, D, E).

	A			B			C			D			E		
OSS	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R
HFC	0.812	0.753	0.881	0.814	0.757	0.881	0.814	0.757	0.881	0.597	0.686	0.528	0.512	0.435	0.626
SF	**0.880**	**0.922**	**0.842**	**0.867**	**0.895**	**0.841**	**0.867**	**0.889**	**0.846**	**0.853**	**0.844**	**0.863**	**0.679**	**0.693**	**0.665**
PD	0.540	0.396	0.851	0.540	0.403	0.818	0.544	0.409	0.808	0.491	0.373	0.718	0.458	0.337	0.713
WPD	0.832	0.762	0.811	0.801	0.797	0.806	0.809	0.791	0.822	0.587	0.630	0.564	0.557	0.625	0.505
CD	0.866	0.798	0.854	0.844	0.807	0.870	0.843	0.792	0.881	0.541	0.586	0.518	0.522	0.545	0.509
RCD	0.824	0.823	0.770	0.795	0.818	0.761	0.814	0.814	0.803	0.715	0.680	0.745	0.650	0.652	0.653

Table 6. Results with P, Precision, F, F-measure and R, Recall, for Mix onsets in the Bello Dataset using all the 5 Peak Selection methods (A, B, C, D, E).

sults are quite similar for all the onset classes.

By using a normalization based on the maximum standard deviation – Peak Selection Method C – the results are not very different from the results obtained by using a normalization based on the maximum absolute value – Peak Selection Method A. In the case of the HFC, SF, and RCD, we obtain practically the same results (they change by no more than 1pp) for all the onset classes. In the case of the PD OSS, we have losses of about 10pp for the PP onset class but for the other classes the results remain basically the same (they change by less than 1pp). For the WPD and CD functions the behaviour is mixed, that is, for some onset classes the results improve while for others the results get poorer, although the magnitude of the changes in this OSS is less than 2pp, which means that the changes are not very significant. This Peak Selection Method improves the CD in the PP class, but makes its results worse in the PNP and Mix classes. On the other hand, it improves the WPD in the PNP class, but makes it worse in the Mix class.

The use of a smoothing filter on the Onset Detection Function – Peak Selection Method D – causes the results, in general, to be much different than the results obtained with the Peak Selection Method A. For the HFC OSS, the results decrease from 10 to 25pp and for the SF the tendency is the same, except that for the NPP onset class the results improve slightly (less than 1pp) and the global losses are not so pronounced: they reach at most 9pp. In the case of the PD function we obtain mixed behaviour: for the NPP and Mix onsets the results are 2.5 and 5pp worse respectively while for the PP onsets the results improve by 3pp and for the PNP we have a 20pp improvement. The results get about 2 to 34pp and 7.5 to 44pp worse for the WPD and CD OSS respectively, while for the RCD OSS the results remain similar for NPP class, but get 9 to 30pp worse for the other onset classes. The filter has some kind of "good" effect only on the PD OSS, maybe because this kind of function is the most irregular and the filter brings some positive uniformity, and on the other OSS one obtains an excess of uniformity with the filter, decreasing the precision of the OSS.

Dropping the local maximum condition in the peak picking algorithm – Peak Selection Method E – makes, in general, the results be much worse than the results of the Peak Selection Method A. For the HFC the results are all around 30pp worse while the results can be to 20pp worse for the SF, 40pp worse for the PD and to 34pp worse for the WPD. For the complex domain family, the results can be to 40pp worse for the CD and 50pp worse for the RCD.

4.4.3 *Balance*

Having in mind the discussion of the two previous subsections, we can make a global balance. First of all, in general, the differences between the results obtained by applying a running mean and a running median threshold are not statistically significant ($W = 291, p = 0.959$ in the Wilcoxon signed rank sum test with continuity correction[3]) and they are dependent upon the particular onset class and OSS, which implies that for certain applications that need just a certain type of onsets, one specific type of threshold can be chosen in favour of the other.

Concerning the normalization methods, the differences between the results obtained with the two kinds of normalization used are not statistically significant ($W = 290, p = 0.975$ in the Wilcoxon signed rank sum test with continuity correction).

On the other hand, the results obtained by the usage of a smoothing filter get significantly poorer ($W = 427, p = 0.004$ in the Wilcoxon signed rank sum test with continuity correction) in most of the cases, except for the single case of the PD OSS. This means that one should not use a smoothing filter at all (except maybe for the single case of the PD function) or try to test a different filter from the one used in this study.

Finally, not using the local maximum condition makes the results get significantly poorer ($W = 500, p < 0.001$ in the Wilcoxon signed rank sum test with continuity cor-

[3] All statistical tests were obtained using R [10].

rection), which means that one should really use the local maximum condition.

5. CONCLUSIONS

In this paper we have compared the influence of 5 distinct Peak Selection Methods on the performance of some of the most common onset detection methods. Our comparison focused on both the influence of the peak selection on each particular OSS but also on the influence of the results in each onset classes.

We have found that, in general, the Peak Selection Method used can be of great influence on the results obtained, but not all of them have the same magnitude of influence. Globally, the influence of using a running-mean or running-average threshold and of using a normalization based on the maximum absolute value or on the maximum standard deviation is quite small (at best around 3-4pp) and can be both positive or negative, depending on the cases. On the other hand using a low-pass filter as a first smoothing step and not using a local maximum condition as final step can be of great negative influence, sometimes worse by 50pp.

We also noticed that, globally, the SF OSS is the most robust to Peak Selection changes, and the PD is the most susceptible to changes.

In the future this work can be extended by adding a few Onset Detection methods to the comparison and also by testing more Peak Selection Methods. One possibility is to add more types of filters to the smoothing to see if the negative influence continues or is just something related to the filter we used. We also intend to check if these conclusions apply to a larger dataset.

6. ACKNOWLEDGEMENTS

We would like to thank Juan Pablo Bello at the NYU for freely providing the dataset we used for our experiments.

This work was partially supported by national funds through FCT – Fundação para a Ciência e a Tecnologia, under project PEst-OE/EEI/LA0021/2011.

7. REFERENCES

[1] 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 Speech and Audio Processing*, 13(5):1035–1047, 2005.

[2] S. Dixon. Onset Detection Revisited. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, pages 133–137, September 2006.

[3] C. Duxbury, J.P. Bello, M. Davies, and M. Sandler. A combined phase and amplitude based approach to onset detection for audio segmentation. In *Proc. 4th European Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS-03)*, pages 275–280, Singapore, 2003. World Scientific Publishing Co. Pte. Ltd.

[4] F. Eyben, S. Böck, B. Schuller, and A. Graves. Universal Onset Detection with Bidirectional Long Short-Term Memory Neural Networks. In *11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 589–594, 2010.

[5] A Holzapfel, Y Stylianou, A C Gedik, and B Bozkurt. Three Dimensions of Pitched Instrument Onset Detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1517–1527, August 2010.

[6] I. Kauppinen. Methods for detecting impulsive noise in speech and audio signals. In *14th International Conf. on Digital Signal Processing Proc. DSP 2002 (Cat. No.02TH8628)*, volume 2, pages 967–970. IEEE.

[7] A. Klapuri and M. Davy, editors. *Signal Processing Methods for Music Transcription*. Springer, 2006.

[8] A.P. Klapuri, A.J. Eronen, and J.T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–355, 2006.

[9] MIREX. Mirex 2011: Audio onset detection task. `http://www.music-ir.org/mirex/wiki/2011:Audio_Onset_Detection`, May 2011.

[10] R Development Core Team. R: A language and environment for statistical computing. 2008. ISBN 3-900051-07-0.

[11] X. Rodet and F. Jaillet. Detection and modeling of fast attack transients. In *Proc. of the International Computer Music Conference*, pages 30–33, 2001.

[12] C. Rosão and R. Ribeiro. Trends in Onset Detection. In *Proc. of the 2011 Workshop on Open Source and Design of Communication*, pages 75–81. ACM, 2011.

[13] C. Rosão, R. Ribeiro, and D. Martins de Matos. Comparing Onset Detection Methods Based on Spectral Features. In *Proc. of the 2012 Workshop on Open Source and Design of Communication*. ACM, 2012.

[14] U. Zölzer, X. Amatriain, D. Arfib, J. Bonada, G. De Poli, P. Dutilleux, G. Evangelista, F. Keiler, A. Loscos, D. Rocchesso, M. Sandler, X. Serra, and T. Todoroff. *DAFX:Digital Audio Effects*. Wiley, 2002.

EVALUATION OF MUSICAL FEATURES FOR EMOTION CLASSIFICATION

Yading Song, Simon Dixon, Marcus Pearce
Centre for Digital Music, Queen Mary University of London
{yading.song, simon.dixon, marcus.pearce}@eecs.qmul.ac.uk

ABSTRACT

Because music conveys and evokes feelings, a wealth of research has been performed on music emotion recognition. Previous research has shown that musical mood is linked to features based on rhythm, timbre, spectrum and lyrics. For example, sad music correlates with slow tempo, while happy music is generally faster. However, only limited success has been obtained in learning automatic classifiers of emotion in music. In this paper, we collect a ground truth data set of 2904 songs that have been tagged with one of the four words "happy", "sad", "angry" and "relaxed", on the Last.FM web site. An excerpt of the audio is then retrieved from 7Digital.com, and various sets of audio features are extracted using standard algorithms. Two classifiers are trained using support vector machines with the polynomial and radial basis function kernels, and these are tested with 10-fold cross validation. Our results show that spectral features outperform those based on rhythm, dynamics, and, to a lesser extent, harmony. We also find that the polynomial kernel gives better results than the radial basis function, and that the fusion of different feature sets does not always lead to improved classification.

1. INTRODUCTION

In the past ten years, music emotion recognition has attracted increasing attention in the field of music information retrieval (MIR) [16]. Music not only conveys emotion, but can also modulate a listener's mood [8]. People report that their primary motivation for listening to music is its emotional effect [19] and the emotional component of music has been recognised as most strongly associated with music expressivity [15].

Recommender systems for managing a large personal music collections typically use collaborative filtering [28] (historical ratings) and metadata- and content-based filtering [3] (artist, genre, acoustic features similarity). Emotion can be easily incorporated into such systems to subjectively organise and search for music. Musicovery [1], for example, has successfully used a dimensional model of emotion within its recommendation system.

[1] http://musicovery.com/

Although music emotion has been widely studied in psychology, signal processing, neuroscience, musicology and machine learning, our understanding is still at an early stage. There are three common issues: 1. collection of ground truth data; 2. choice of emotion model; 3. relationships between emotion and individual acoustic features [13].

Since 2007, the annual Music Information Retrieval Evaluation eXchange (MIREX) [2] has organised an evaluation campaign for MIR algorithms to facilitate finding solutions to the problems of audio music classification. In previous studies, significant research has been carried out on emotion recognition including regressor training: using multiple linear regression [6] and Support Vector Machines (SVM) [23,37], feature selection [35,36], the use of lyrics [13] and advanced research including mood classification on television theme tunes [30], analysis with electroencephalogram (EEG) [18], music expression [32] and the relationship with genre and artist [12]. Other relevant work on classification suggests that feature generation can outperform approaches based on standard features in some contexts [33].

In this paper, we aim to better explain and explore the relationship between musical features and emotion. We examine the following parameters: first, we compare four perceptual dimensions of musical features: *dynamics*, *spectrum*, *rhythm*, and *harmony*; second, we evaluate an SVM associated with two kernels: polynomial and radial basis functions; third, for each feature we compare the mean and standard deviation feature value. The results are trained and tested using semantic data retrieved from last.fm [3] and audio data from 7digital [4].

This paper is structured as follows. In section 2, three psychological models are discussed. Section 3 explains the dataset collection we use in training and testing. The procedure is described in section 4, which includes data preprocessing (see section 4.1), feature extraction (see section 4.2) and classification (see section 4.3). Section 5 explains four experiments. Finally, section 6 concludes the paper and presents directions for future work.

[2] http://www.music-ir.org/mirex/wiki/MIREX_HOME
[3] http://www.last.fm/
[4] http://www.7digital.com/

2. PSYCHOLOGICAL EMOTION MODELS

One of the difficulties in representing emotion is to distinguish music-induced emotion from perceived emotion because the two are not always aligned [5]. Different psychological models of emotion have been compared in a study of perceived emotion [7].

Most music related studies are based on two popular approaches: categorical [10] and dimensional [34] models of emotion. The categorical approach describes emotions with a limited number of innate and universal categories such as happiness, sadness, anger and fear. The dimensional model considers all affective terms arising from independent neurophysiological systems: valence (negative to positive) and arousal (calm to exciting). Recently a more sophisticated model of music-induced emotion - the Geneva Emotion Music Scale (GEMS) model - consisting of 9 dimensions, has been proposed [42]. Our results and analysis are based on the categorical model since we make our data collection through human-annotated social tags which are categorical in nature.

3. GROUND-TRUTH DATA COLLECTION

As discussed above, due to the lack of ground truth data, most researchers compile their own databases [41]. Manual annotation is one of the most common ways to do this. However, it is expensive in terms of financial cost and human labour. Moreover, terms used may differ between individuals. Different emotions may be described using the same term by different people which would result in poor prediction [38]. However, with the emergence of music discovery and recommendation websites such as last.fm which support social tags for music, we can access rich human-annotated information. Compared with the traditional approach of web mining which gives noisy results, social tagging provides highly relevant information for music information retrieval (MIR) and has become an important source of human-generated contextual knowledge [11]. Levy [24] has also shown that social tags give a high quality source of ground truth data and can be effective in capturing music similarity [40].

The five mood clusters proposed by MIREX [14] (such as rollicking, literate, and poignant) are not popular in social tags. Therefore, we use four basic emotion classes: *happy*, *angry*, *sad* and *relaxed*, considering these four emotions are widely accepted across different cultures and cover the four quadrants of the 2-dimensional model of emotion [22]. These four basic emotions are used as seeds to retrieve the top 30 tags from last.fm. We then obtain a list of songs labelled with the retrieved tags. Table1 and table 2 show an example of the retrieved results.

Given the retrieved titles and the names of the singers, we use a public API to get preview files. The results cover different types of pop music, meaning that we avoid particular artist and genre effects [17]. Since the purpose of this step is to find ground truth data, issues such as cold start, noise, hacking, and bias are not relevant [4, 20].

Most datasets on music emotion recognition are quite small (less than 1000 items), which indicates that 2904 songs (see table 3) for four emotions retrieved by social tags is a good size for the current experiments. The dataset will be made available [5], to encourage other researchers to reproduce the results for research and evaluation.

Happy	**Angry**	**Sad**	**Relax**
happy	angry	sad	relax
happy hardcore	angry music	sad songs	relax trance
makes me happy	angry metal	*happysad*	relax music
happy music	angry pop music	sad song	jazz relax
happysad	angry rock	sad & beautiful	only relax

Table 1. Top 5 tags returned by last.fm

Singer	**Title**
Noah And The Whale	5 Years Time
Jason Mraz	I'm Yours
Rusted Root	Send Me On My Way
Royksopp	Happy Up Here
Karen O and the Kids	All Is Love

Table 2. Top songs returned with tags from the "happy" category.

Emotion	**Number of Songs**
Happy	753
Angry	639
Sad	763
Relaxed	749
Overall	2904

Table 3. Summary of ground truth data collection

4. PROCEDURES

The experimental procedure consists of four stages: data collection, data preprocessing, feature extraction, and classification, as shown in figure 1.

4.1 Data Preprocessing

As shown in Table 1, there is some noise in the data such as confusing tags and repeated songs. We manually remove data with the tag *happysad* which existed in both the happy and sad classes and delete the repeated songs, to make sure every song will only exist once in a single class. Moreover, we convert our dataset to standard wav format (22,050 Hz sampling rate, 16 bit precision and mono channel). The song excerpts are either 30 seconds or 60 seconds, representing the most salient part of the song [27], therefore there is no need to truncate. At the end, we normalise the excerpts by dividing by the highest amplitude to mitigate the *production effect* of different recording levels.

4.2 Feature Extraction

As suggested in the work of Saari and Eerola [35], two different types of feature (mean and standard deviation) with

[5] The dataset can be found at *https://code.soundsoftware.ac.uk/projects-/emotion-recognition*

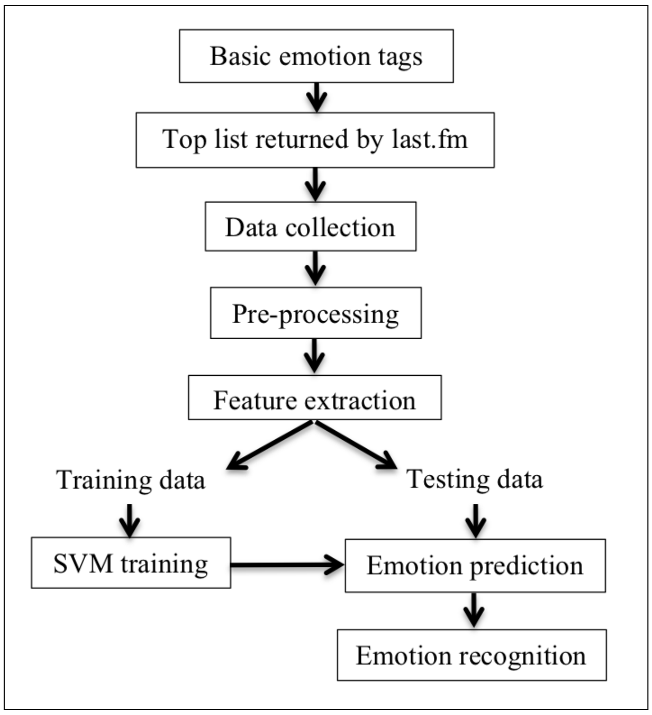


Figure 1. Procedure

a total of 55 features were extracted using the MIR toolbox [6] [21] (shown in table 4). The features are categorized into the following four perceptual dimensions of music listening: *dynamics*, *rhythm*, *spectral*, and *harmony*.

4.3 Classification

The majority of music classification tasks [9] (genre classification [25, 39], artist identification [29], and instrument recognition [31]) have used k-nearest neighbour (K-NN) [26] and support vector machines (SVM) [2]. In the case of audio input features, the SVM has been shown to perform best [1].

In this paper, therefore, we choose support vector machines as our classifier, using the implementation of the sequential minimal optimisation algorithm in the Weka data mining toolkit [7]. SVMs are trained using polynomial and radial basis function (RBF) kernels. We set the cost factor $C = 1.0$, and leave other parameters unchanged. An internal 10-fold cross validation is applied. To better understand and compare features in four perceptual dimensions, our experiments are divided into four tasks.

Experiment 1: we compare the performance of the two kernels (polynomial and RBF) using various features.

Experiment 2: four classes (perceptual dimensions) of features are tested separately, and we compare the results to find a dominant class.

Experiment 3: two types of feature descriptor, mean and standard deviation, are calculated. The purpose is to compare values for further feature selection and dimensionality reduction.

[6] Version 1.3.3: https://www.jyu.fi/music/coe/materials/mirtoolbox
[7] http://www.cs.waikato.ac.nz/ml/weka/

Dimen.	No.	Features	Acronyms
Dynamics	1-2	RMS energy	RMSm, RMSstd
	3-4	Slope	Ss, Sstd
	5-6	Attack	As, Astd
	7	Low energy	LEm
Rhythm	1-2	Tempo	Ts, Tstd
	3-4	Fluctuation peak (pos, mag)	FPm, FMm
	5	Fluctuation centroid	FCm
Spec.	1-2	Spectrum centroid	SCm, SCstd
	3-4	Brightness	BRm, BRstd
	5-6	Spread	SPm, SPstd
	7-8	Skewness	SKm, SKstd
	9-10	Kurtosis	Km, Kstd
	11-12	Rolloff95	R95s, R95std
	13-14	Rolloff85	R85s, R85std
	15-16	Spectral Entropy	SEm, SEstd
	17-18	Flatness	Fm, Fstd
	19-20	Roughness	Rm, Rstd
	21-22	Irregularity	IRm. IRstd
	23-24	Zero crossing rate	ZCRm, ZCRstd
	25-26	Spectral flux	SPm, SPstd
	27-28	MFCC	MFm, MFstd
	29-30	DMFCC	DMFm, DMFstd
	31-32	DDMFCC	DDm, DDstd
Harmony	1-2	Chromagram peak	CPm, CPstd
	3-4	Chromagram centroid	CCm, CCstd
	5-6	Key clarity	KCm, KCstd
	7-8	Key mode	KMm, KMstd
	9-10	HCDF	Hm, Hstd

Table 4. The feature set used in this work; m = **mean**, std = **standard deviation**.

Experiment 4: different combinations of feature classes (e.g., spectral with dynamics) are evaluated in order to determine the best-performing model.

5. RESULTS

5.1 Experiment 1

In experiment 1, SVMs trained with two different kernels are compared. Previous studies [23] have found in the case of audio input that the SVM performs better than other classifiers (Logistic Regression, Random Forest, GMM, K-NN and Decision Trees). To our knowledge, no work has been reported explicitly comparing different kernels for SVMs. In emotion recognition, the radial basis function kernel is a common choice because of its robustness and accuracy in other similar recognition tasks [1].

	Polynomial		RBF		
Feature Class	**Accuracy**	**Time**	**Accuracy**	**Time**	**No.**
Dynamics	37.2	0.44	26.3	32.5	7
Rhythm	37.5	0.44	34.5	23.2	5
Harmony	47.5	0.41	36.6	27.4	10
Spectral	**51.9**	0.40	48.1	14.3	32

Table 5. Experiment 1 results: time = model building time, No. = number of features in each class

The results in table 5 show however that regardless of the features used, the polynomial kernel always achieved the higher accuracy. Moreover, the model construction times for each kernel are dramatically different. The average construction time for the polynomial kernel is 0.4 seconds, while the average time for the RBF kernel is 24.2

seconds, around 60 times more than the polynomial kernel. The following experiments also show similar results. This shows that polynomial kernel outperforms RBF in the task of emotion recognition at least for the parameter values used here.

5.2 Experiment 2

In experiment 2, we compare the emotion prediction results for the following perceptual dimensions: *dynamics*, *rhythm*, *harmony*, and *spectral*. Results are shown in figure 2). Dynamics and rhythm features yield similar results, with harmony features providing better results, but the spectral class with 32 features achieves the highest accuracy of 51.9%. This experiment provides a baseline model, and further exploration of multiple dimensions is performed in experiment 4.

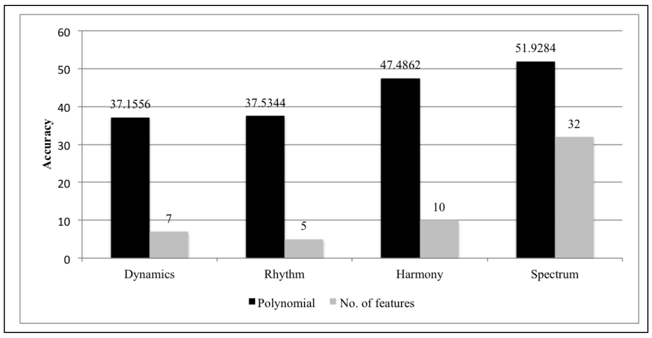


Figure 2. Comparison of classification results for the four classes of features.

5.3 Experiment 3

In this experiment, we evaluate different types of feature descriptors, mean value and standard deviation for each feature across all feature classes, for predicting the emotion in music. The results in table 6 show that the use of both mean and standard deviation values gives the best results in each case. However, the processing time increased, so choosing the optimal descriptor for each feature is highly desirable. For example, choosing only the mean value in the harmony class, we lose 2% of accuracy but increase the speed while the choice of standard deviation results in around 10% accuracy loss. As the number of features increases, the difference between using mean and standard deviation will be reduced. However, more experiments are needed to explain why the mean in harmony and spectral features, and standard deviation values of dynamics and rhythm features have higher accuracy scores.

5.4 Experiment 4

In order to choose the best model, the final experiment fuses different perceptual features. As presented in table 7, optimal accuracy is not produced by the combination of all features. Instead, the use of spectral, rhythm and harmony (but not dynamic) features produces the highest accuracy.

Features Class	Polynomial	No. features
Dynamics all	**37.2**	7
Dynamics mean	29.7	3
Dynamics std	*33.8*	3
Rhythm all	37.5	5
Rhythm mean	28.7	1
Rhythm std	**34.2**	1
Harmony all	**47.5**	10
Harmony mean	*45.3*	5
Harmony std	38.3	5
Spectral all	**51.9**	32
Spectral mean	*49.6*	16
Spectral std	47.5	16
Spec+Dyn all	**52.3**	39
Spec+Dyn mean	*50.5*	19
Spec+Dyn std	48.7	19
Spec+Rhy all	**52.3**	37
Spec+Rhy mean	*49.8*	17
Spec+Rhy std	47.8	17
Spec+Har all	**53.3**	42
Spec+Har mean	*51.3*	21
Spec+Har std	50.3	21
Har+Rhy all	**49.1**	15
Har+Rhy mean	*45.6*	6
Har+Rhy std	41.2	6
Har+Dyn all	**48.8**	17
Har+Dyn mean	*46.9*	8
Har+Dyn std	42.4	8
Rhy+Dyn all	**41.7**	12
Rhy+Dyn mean	32.0	4
Rhy+Dyn std	*38.8*	4

Table 6. Comparison of mean and standard deviation (std) features.

Features	Accuracy	No. features
Spec+Dyn	52.3	39
Spec+Rhy	52.3	37
Spec+Har	53.3	42
Har+Rhy	49.1	15
Har+Dyn	48.8	17
Rhy+Dyn	41.7	12
Spec+Dyn+Rhy	52.4	44
Spec+Dyn+Har	53.8	49
Spec+Rhy+Har	**54.0**	47
Dyn+Rhy+Har	49.7	22
All Features	53.6	54

Table 7. Classification results for combinations of feature sets.

6. CONCLUSION AND FUTURE WORK

In this paper, we collected ground truth data on the emotion associated with 2904 pop songs from last.fm tags. Audio features were extracted and grouped into four perceptual dimensions for training and validation. Four experiments were conducted to predict emotion labels. The results suggest that, instead of the conventional approach using SVMs trained with a RBF kernel, a polynomial kernel yields higher accuracy. Since no single dominant features have been found in emotion recognition, we explored the performance of different perceptual classes of feature for predicting emotion in music. Experiment 3 found that dimensionality reduction can be achieved through removing either mean or standard deviation values, halving the number of features used, with, in some cases, only 2% accuracy loss. The last experiment found that inclusion of dynamics features with the other classes actually impaired the performance of the classifier while the combination of spectral, rhythmic and harmonic features yielded optimal performance.

In future work, we will expand this research both in depth and breadth, to find features and classes of features which best represent emotion in music. We will examine higher-level dimensions such as temporal evolution features, as well as investigating the use of auditory models. Using the datasets retrieved from Last.fm, we will compare the practicability of social tags with other human-annotated datasets in emotion recognition. Through these studies of subjective emotion, we will develop methods for incorporating other empirical psychological data in a subjective music recommender system.

7. ACKNOWLEDGEMENTS

We acknowledge the support of the Queen Mary University of London Postgraduate Research Fund (QMPGRF) and the China Scholarship Council. We would like to thank the reviewers and Emmanouil Benetos for their advice and comments.

8. REFERENCES

[1] K. Bischoff, C. S. Firan, R. Paiu, W. Nejdl, C. Laurier, and M. Sordo. Music Mood and Theme Classification - A Hybrid Approach. In *10th International Society for Music Information Retrieval Conference*, number Ismir, pages 657–662, 2009.

[2] E. Boser, N. Vapnik, and I. M. Guyon. Training Algorithm Margin for Optimal Classifiers. In *ACM Conference on Computational Learning Theory*, pages 144–152, 1992.

[3] P. Cano, M. Koppenberger, and N. Wack. Content-based Music Audio Recommendation. In *Proceedings of the 13th annual ACM international conference on Multimedia*, number ACM, pages 211–212, 2005.

[4] O. Celma. Foafing the Music : Bridging the Semantic Gap in Music Recommendation. In *The Semantic Web-ISWC*, 2006.

[5] T. Eerola. Are the Emotions Expressed in Music Genre-specific? An Audio-based Evaluation of Datasets Spanning Classical , Film , Pop and Mixed Genres. *Journal of New Music Research*, 40(March 2012):349–366, 2011.

[6] T. Eerola, O. Lartillot, and P. Toiviainen. Prediction of Multdimensional Emotional Ratings in Music from Audio Using Multivariate Regression Models. In *10th International Society for Music Information Retrieval Conference*, number Ismir, pages 621–626, 2009.

[7] T. Eerola and J.K. Vuoskoski. A Comparison of the Discrete and Dimensional Models of Emotion in Music. *Psychology of Music*, 39(1):18–49, August 2010.

[8] Y. Feng and Y. Zhuang. Popular Music Retrieval by Detecting Mood. In *International Society for Music Information Retrieval Conference*, volume 2, pages 375–376, 2003.

[9] Z. Fu, G. Lu, K.M. Ting, and D. Zhang. A Survey of Audio-based Music Classification and Annotation. *IEEE Transactions on Multimedia*, 13(2):303–319, 2011.

[10] K. Hevner. Experimental studies of the elements of expression in music. *The American Journal of Psychology*, 48:246–268, 1936.

[11] X. Hu, M. Bay, and J.S. Downie. Creating a Simplified Music Mood Classification Grouth-truth Set. In *International Conference on Music Information Retrieval*, pages 3–4, 2007.

[12] X. Hu and J.S. Downie. Exploring Mood Metadata: Relationships with Genre, Artist and Usage Metadata. In *8th International Conference on Music Information Retrieval*, 2007.

[13] X. Hu, J.S. Downie, and A.F. Ehmann. Lyric Text Mining in Music Mood Classification. In *10th International Society for Music Information Retrieval Conference*, number Ismir, pages 411–416, 2009.

[14] X. Hu, J.S. Downie, C. Laurier, and M. Bay. The 2007 MIREX Audio Mood Classification Task: Lesson Learned. In *International Society for Music Information Retrieval Conference*, pages 462–467, 2008.

[15] P. N. Juslin, J. Karlsson, E. Lindström, A. Friberg, and E. Schoonderwaldt. Play it Again with Feeling: Computer Feedback in Musical Communication of Emotions. *Journal of experimental psychology. Applied*, 12(2):79–95, June 2006.

[16] 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 International Society for Music Information Retrieval Conference*, number Ismir, pages 255–266, 2010.

[17] Y.E. Kim, D.S. Williamson, and S. Pilli. Towards Quantifying the Album Effect in Artist Identification. In *International Society for Music Information Retrieval Conference*, 2006.

[18] S. Koelstra, C. Muhl, and M. Soleymani. Deap: A Database for Emotion Analysis Using Physiological Signals. *IEEE Trans. on Affective Computing*, pages 1–15, 2011.

[19] C.L. Krumhansl. Music : A Link Between Cognition and Emotion. *American Psychological Society*, 11(2):45–50, 2002.

[20] P. Lamere. Social Tagging and Music Information Retrieval. *Journal of New Music Research*, 37(2):101–114, June 2008.

[21] O. Lartillot and P. Toiviainen. MIR in Matlab (II): A Toolbox for Musical Feature Extraction from Audio. In *International Conference on Music Information Retrieval*, number Ii, pages 237–244, 2007.

[22] C. Laurier and J. Grivolla. Multimodal Music Mood Classification Using Audio and Lyrics. In *Int. Conf. Machine Learning and Applications*, pages 1–6, 2008.

[23] C. Laurier, P. Herrera, M. Mandel, and D. Ellis. Audio Music Mood Classification Using Support Vector Machine. In *MIREX task on Audio Mood Classification*, pages 2–4, 2007.

[24] M. Levy. A Semantic Space for Music Derived from Social Tags. In *Austrian Compuer Society*, volume 1, page 12. Citeseer, 2007.

[25] B. Lines, E. Tsunoo, G. Tzanetakis, and N. Ono. Beyond Timbral Statistics : Improving Music Classification Using Percussive. *IEEE Transactions on Audio, Speech and Language Processing*, 19(4):1003–1014, 2011.

[26] T. M. Cover and P. E. Hart. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[27] K.F. MacDorman, S. Ough, and C. Ho. Automatic Emotion Prediction of Song Excerpts: Index Construction, Algorithm Design, and Empirical Comparison. *Journal of New Music Research*, 36(4):281–299, December 2007.

[28] T. Magno and C. Sable. A Comparison of Signal of Signal-based Music Recommendation to Genre Labels, Collaborative Filtering, Musicological Analysis, Human Recommendation and Random Baseline. In *Proceedings of the 9th International Conference of Music Information Retrieval*, pages 161–166, 2008.

[29] M. Mandel. Song-level Features and Support Vector Machines for Music Classification. In *Proc. International Conference on Music Information Retrieval*, 2005.

[30] M. Mann, T.J. Cox, and F.F. Li. Music Mood Classification of Television Theme Tunes. In *12th International Society for Music Information Retrieval Conference*, number Ismir, pages 735–740, 2011.

[31] J. Marques and P.J. Moreno. A Study of Musical Instrument Classification Using Gaussian Mixture Models and Support Vector Machines, 1999.

[32] L. Mion and G.D. Poli. Score-Independent Audio Features for Description of Music Expression. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):458–466, 2008.

[33] F. Pachet and P. Roy. Analytical Features: A Knowledge-Based Approach to Audio Feature Generation. *EURASIP Journal on Audio, Speech, and Music Processing*, 2009(2):1–23, 2009.

[34] J.A. Russell, A. Weiss, and G.A. Mendelsohn. Affect Grid: A Single-item Scale of Pleasure and Arousal. *Journal of Personality and Social Psychology*, 57(3):493–502, 1989.

[35] P. Saari, T. Eerola, and O. Lartillot. Generalizability and Simplicity as Criteria in Feature Selection: Application to Mood Classification in Music. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6):1802–1812, 2011.

[36] E.M. Schmidt, D. Turnbull, and Y.E. Kim. Feature Selection for Content-Based, Time-Varying Musical Emotion Regression Categories and Subject Descriptors. In *Multimedia Information Retrieval*, pages 267–273, 2010.

[37] B. Schuller, J. Dorfner, and G. Rigoll. Determination of Nonprototypical Valence and Arousal in Popular Music: Features and Performances. *EURASIP Journal on Audio, Speech, and Music Processing*, 2010:1–19, 2010.

[38] D. Turnbull, L. Barrington, and G. Lanckriet. Five Approaches to Collecting Tags for Music. In *Proceedings of the 9th International Conference of Music Information Retrieval*, pages 225–230, 2008.

[39] G. Tzanetakis and P. Cook. Musical Genre Classification of Audio Signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[40] D. Wang, T. Li, and M. Ogihara. Tags Better Than Audio Features? The Effect of Joint use of Tags and Audio Content Features for Artistic Style Clutering. In *11th International Society on Music Information Retrieval Conference*, number ISMIR, pages 57–62, 2010.

[41] D. Yang and W.S. Lee. Disambiguating Music Emotion Using Software Agents. In *Proceedings of the 5th International Conference on Music Information Retrieval*, pages 52–58, 2004.

[42] M. Zentner, D. Grandjean, and K.R. Scherer. Emotions evoked by the sound of music: characterization, classification, and measurement. *Emotion (Washington, D.C.)*, 8(4):494–521, August 2008.

BLAST FOR AUDIO SEQUENCES ALIGNMENT: A FAST SCALABLE COVER IDENTIFICATION TOOL

Benjamin Martin[1], **Daniel G. Brown**[2], **Pierre Hanna**[1], **Pascal Ferraro**[1]

[1]Université de Bordeaux
CNRS, LaBRI, UMR 5800
{bmartin,hanna,ferraro} @labri.fr

[2]University of Waterloo
Cheriton School of Computer Science
dan.brown@uwaterloo.ca

ABSTRACT

Searching for similarities in large musical databases is common for applications such as cover song identification. These methods typically use dynamic programming to align the shared musical motifs between subparts of two recordings. Such music local alignment methods are slow, as are the bioinformatics algorithms they are closely related to. We have adapted the ideas of the Basic Local Alignment Search Tool (BLAST) for biosequence alignment to the domain of aligning sequences of chroma features. Our tool allows local music sequence alignment in near-linear time. It identifies small regions of exact match between sequences, called seeds, and builds local alignments that include these seeds. Seed determination is a key issue for the accuracy of the method and closely depends on the database, the representation and the application. We introduce a particular seeding approach for cover detection, and evaluate it on both a 2000-piece training set and the million song dataset (*MSD*). We show that the heuristic alignment drastically improves time computation for cover song detection. Alignment sensitivity is still very high on the small database, but is dramatically weakened on the *MSD*, due to differences in chroma features. We discuss the impact of different choices of these features on alignment of musical pieces.

1. INTRODUCTION

During the last decade, an increasing number of large music datasets have become available. One may now access a huge amount of music audio, stored for instance on personal computers, mobile devices or online. In this context, Music Information Retrieval (MIR) focuses on automatic classification, organization, description of music content.

To assess musical similarities between pieces, for example, a major challenge of MIR is analysing acoustic content. Using signal processing techniques, music features are first infered from audio content. Each of them are related to a specific aspect of music. A fairly typical MIR approach consists in obtaining such a feature for short frames of audio signal, hence computing a symbolic string representation that carries the change in some musical property along a given music track. Such symbolic representations can be further analysed and compared to detect meaningful similarities. MIR studies typically employ comparison techniques either custom built or adapted from other fields of information science [6]. Many such techniques account for slight variations in musical features. The way we perceive music is known to work at different time scales, and allows for slight differences in music information received over time. For instance, one may easily recognize a chorus sung twice in a song although the singer may sing different lyrics, the melody may have changed, or the instruments playing may be different in both occurrences.

Among the many applications of automatic assessment of musical similarities in music datasets, cover song identification has been of major concern over the last few years [16]. Cover songs are usually defined as multiple renditions of the same original music piece. They may be played by other performers from different music genres or with distinct recording environments [16]. Although two cover versions of an original song may differ widely in instrumentation, singing voices, background noise, key, structural arrangement, genre, *etc.*, they should hold enough musical similarity for human perception to identify them as renditions of the same piece. To detect such similarities, most retrieval systems use dynamic programming [16]. A key drawback of such systems is their inability to efficiently scale to the range of musical data available in musical platforms, *i.e.* to the order of tens of millions of tracks [6, 16].

We propose an indexing method that substantially increases the efficiency of alignment-based retrieval systems. Our method uses a widely known bio-sequence indexing technique, BLAST [2]. We adapt this method by investigating the distribution of symbols among features sequences, and deducing a strategy for efficient indexing. An empirical study and an evaluation are performed on a custom-built cover song dataset, as well as on the Million Song Dataset (*MSD*) [5]. The remainder of this paper is organized as follows. Previous works are described in Section 2. Section 3.1 presents audio representations and alignment techniques used, and describes the principle of the bio-indexing tool we propose to adapt. Section 3 details the investigation over feature sequences, and describes the particular settings of our music application. Finally, Section 5 presents results obtained for a practical cover song identification, while concluding remarks and perspectives are depicted in Section 6.

2. RELATED WORK

Several heuristics for reducing the computational cost of dynamic programming have been introduced for MIR applications. Dannenberg and Hu [7] proposed to discover patterns in audio sequences by partially exploring the search space around bands, relying on global thresholds that indicate limits on the deviation from diagonals, implicitly assuming that insertions and deletions are rare. Combined to a clustering technique, this process was used to deduce suboptimal alignments and infer the structure of audio pieces.

Kilian and Hoos [12] already introduced BLAST in MIR context, in order to search for approximate patterns in symbolic music. They tested the technique on MIDI excerpts and emphasized the efficiency of the alignment of similar patterns with the bi-directional extension of exact match regions. Authors infered a high potential for BLAST in a non-symbolic input configuration, but did not test it explicitly [13]. Although the work presented in this paper adapts the same BLAST algorithm, it is subtantially different from this study. First, our technique is applied to feature sequences obtained from audio signals. More importantly, we aim at comparing distinct songs with possibly similar regions, not discovering similar patterns inside a single piece. Finally, the tool is used in our case as a filtering technique to allow efficient identification of cover songs.

Analogously, many studies adapt exact audio identification systems to account for musical variations (see [14] and references therein). For instance, Kurth and Muller [14] presented an efficient matching technique robust to typical variations in interpretation of classical music. The approach here is reversed relative to our: an exact fast fingerprinting system is adapted to account for local slight variations, whereas we propose to enhance the efficiency of an accurate slow approximate identification technique. They report an acceleration of the matching process by a factor of 15 to 20 while keeping a high robustness to interpretation variations. However, they emphasize that such a speed-up factor is suitable for efficiently handling datasets in the order of tens of thousands tracks [14]. To handle fast search in larger datasets, an indexing system for cover song identification on the *MSD* was recently proposed [4]. Landmarks estimations are performed from *MSD* chroma features, and heuristic jumpcodes between landmarks encode variations of pitch content along cover versions. Authors substantially reduce the problem to binary identification tasks, and report a fair effectiveness/efficiency tradeoff with a speed of about 200 seconds to query the *MSD*. The method we propose in this paper aims at reducing this cover song querying time to the order of a few seconds while keeping a good accuracy in a standard cover retrieval task.

3. COMPARING MUSIC SEQUENCES

3.1 Music representation

Pitch content plays an important role in the structure of audio pieces, in particular for Western music genres. Common compositional processes in such music are organized around melodic and harmonic sequences that listeners identify, consciously or not, as independent phrases or themes.

Pitch Class Profiles (PCP), also known as *chroma features*, are frequently used to describe these types of information. These features classify spectral energies into bins corresponding to the frequency class where they appear, each class taking into account the cyclical perception of pitch in human auditory system. The number of pitch classes p corresponds to the number of frequency bands considered in each octave. The parameter p is usually set to 12 to respect the common note scale, but higher values (generally multiples of 12) can improve the robustness to tuning issues [9]. Chroma features are usually considered as the most robust representation for cover song detection [16].

3.2 Music sequences alignment

This symbolic representation as a sequence of symbols or *string* can be used to define a similarity measure between two audio pieces. Such a metric is expected to isolate significantly similar sections, or repetitions, assessing their resemblance.

3.2.1 Relevance for music information

Repetitions in strings have been studied extensively, either for locating exact repeats or for identifying substrings that are duplicated within a certain tolerance. In the context of music sequences, musical similarity does not rely on exact matches since variations, such as transpositions, interpretation variations, rhythmic irregularities, background noise, may alter the representing sequences.

Alignment approaches are well suited for an accurate recognition of such variations. Widely used for biological sequences, such techniques have been extremely successful in identifying approximate repetitions between local patterns in DNA or RNA strings that reflect, for instance, gene homologies. The relevance of such methods for music information lies in the same "evolutionary" aspect of musical patterns that may slightly change along a single piece or accross similar pieces. Playing variations of some musical theme implies changing sound events such as notes, rhythms, or lyrics, which echoes the mutation of nucleotides of proteins during biological evolution. Therefore, alignment techniques are frequently used for identifying similar patterns in cover songs.

3.2.2 Alignment and dynamic programming

The first accurate distance measure for approximate string comparison in the context of biogical sequences is often credited to Needleman and Wunsch [15]. A string u of length n can be transformed into a string v of length m by applying *edit operations* on the symbols of u and v. These operations are insertions, deletions or substitutions, and each is assigned a cost. The *edit distance* between u and v is defined by the minimum total cost of edit operations required to transform u into v. The *global alignment* of u and v identifies the positions in the sequence u that are not changed during the process of transforming u to v, and their new position in that sequence. A variant of this comparison method, *local alignment* [18], allows finding and extracting a pair of regions, one from each string, which exhibit the highest similarity according to the scoring scheme assigned to edit operations. In a musical context, this might correspond to finding matches between two verses of a song, or between smaller approximately duplicated harmonic figures. Its computation is typically performed by a dynamic programming algorithm filling a $(n+1)\times(m+1)$ matrix. The local alignment of u and v can be seen as the best scoring path in the dynamic programming matrix. This edit path is easily computed in practice by tracing back the series of operations performed. More information about alignment algorithms can be found in [10].

In the context of pitch content, an improvement of local alignment [1] allows taking into account a frequent variation of musical patterns in terms of harmony, namely local transposition. In Western popular music, for instance, local transposition happens when an occurrence of a structural pattern (*e.g.* chorus) is be played a few semitones higher than usual. The improvement of the alignment technique consists in adding a new edit operation, the local transposition of a string versus another. Consequently, we compute several matrices that estimate every possible local transposition, and allow a jump from one matrix to another by paying a corresponding transposition cost [1]. This variant yields more accurate alignments of pitch content, but it is much slower to compute in practice. If the alphabet of symbols has a symbols, the slowdown is a factor of a.

3.2.3 Complexity

Dynamic programming gives $\Theta(nm)$ running time for computing optimal alignment scores. Tracing back the effectively aligned substrings requires $\mathcal{O}(n+m)$. Therefore, to compare a new song of length n with a database of k pieces, each of average length m, requires $\Theta(knm)$ time. The naive space complexity is $\Theta(nm)$, where we store the entire dynamic programming matrix, although a simple trick allows reducing it to $\Theta(\max(m,n))$ by keeping only the last computed lines [10].

Local alignment techniques are particularly useful for the accurate identification of strong similarities between sequences [16]. However, the slowness of the dynamic programming makes them heavy to compute and unadapted to fast querying of large-scale datasets that comprise millions of sequences. Facing a similar challenge, bioinformatics researchers developed a fast heuristic-based search tool dedicated to efficient indexing for local alignment.

3.3 BLAST

The Basic Local Alignment Search Tool (BLAST) [2], reduces the computational cost of local alignment. BLAST relies on the observation that when querying a new sequence to a large database, there are likely only a small number of good alignments, so it filters the database to avoid computing irrelevant alignments of unrelated sequences. BLAST partially explores the dynamic programming search space to filter out many irrelevant comparisons before computing local alignments. It consists of several heuristic layers of rules for refining the potential regions of strong similarity, as described in the next sections.

3.3.1 Seeding the search space

The main heuristic of BLAST lies in the assumption that significant local alignments include small exact matches. As represented in Fig. 1-(i), the dashed edit path of the local alignment of u and v contains diagonal sections, that partially correspond to runs of matches (plain lines), *i.e.* exact repetitions between small sections of u and v. The first step of BLAST finds these small common substrings in order to seed the search space for later local alignments. A practical way of indexing the search space is to fix a seed length N, and index every N-length substring (or words) of every sequence of the dataset in a fast access data structure.

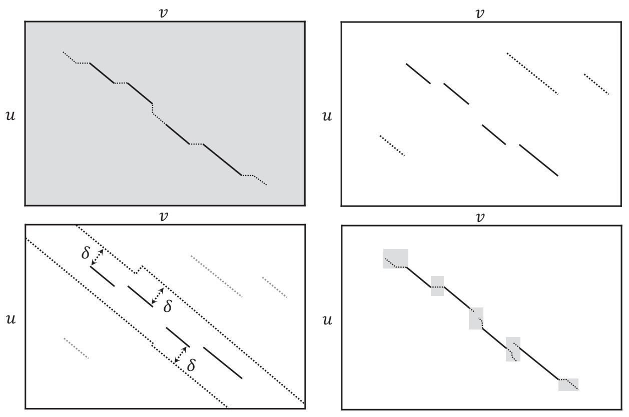


Figure 1. Seeding the search space. Top-Left: exact similar sections (plain lines) inside a local alignment path (dashed line). Top-Right: automatic seeding of the search space, where identified regions may belong to the local alignment path (plain lines) or not (dashed lines). Bottom-left: as result of $\mathcal{F}_1$, seeds are quickly clustered and isolated grey dashed seeds are eliminated. Bottom-right: as a result of $\mathcal{F}_2$, seed extensions roughly depict the local alignment path. Background grey parts highlight the number of dynamic programming computations required.

3.3.2 Filtering seeds

Once seeded as in Fig. $1-(ii)$, the search space includes hit regions that may correspond to high-scoring local alignment of sequences (plain segments) or to suboptimal regions (dashed segments), where the exact seed match arose due to coincidence, not true similarity. The second step of BLAST filters out most seeds that do not correspond to desired local alignments.

A first filtering technique $\mathcal{F}_1$ consists in quickly clustering seeds among the search space, and identifying isolated hits. As illustrated in Fig.1, every correct heuristic alignment should have several seeds around a diagonal (plain lines) that sketches the actual local alignment. Consequently, a pair of sequences that does not have regions comprising a significant number of hits may be filtered out. We use a threshold δ to stand for the maximum inter-diagonal distance allowed between two consecutive seeds to be considered as around the same diagonal, *i.e.* potentially belonging to the same alignment.

A more accurate, also common, filter $\mathcal{F}_2$ is seed extension. Each seed is extended in both directions to determine wether it corresponds to a local similar region or not. We denote by (i,k) and (j,l) the coordinates of a seed in the search space, *i.e.* the hit is between the exactly matching substrings $u[i\cdots j]$ and $v[k\cdots l]$. We compute two small alignments, one starting from (i,k) rolling up towards the top left corner of the search space, and the other one starting from (j,l) going towards the bottom right corner. Each of these alignments, that may be gapped or ungapped, are quickly stopped if their score drops off under a threshold value X. This way, extending unrelated sequences quickly results in stopping the computation, while seeds from locally similar regions grow towards alignments [3].

4. INDEXING MUSIC DATA

4.1 Evaluation framework

4.1.1 Database

Two datasets were used to evaluate our system. The first set, *TSD*, consists of $2,514$ Western popular music pieces comprising 514 cover songs distributed in 17 cover classes, coming from personal music collections [1]. We elected as a second set the million song dataset (*MSD*) [5], that comprises one milliong songs of various musical genres [2]. It includes the Second Hand Songs dataset [3], which comprises $18,196$ cover songs distributed in $5,854$ cover classes. This set is to our knowledge the largest available set of cover song audio features.

Both datasets provide chroma feature sequences. However, it is worth noting that the implementation of such features significantly differs in both cases. In *TSD*, we used our own implementation of *Harmonic Pitch Class Profiles* [9] with a constant frame size and a resolution of 36 subdivisions per octave. This representation was successfully applied for past MIREX tasks [4]. In *MSD*, chroma features were computed using The EchoNest API [5], and consist of segment-synchronized 12-dimensional chroma features, as described in [11]. Thus, each *MSD* chroma represents pitch content on 12 dimensions for a variable audio frame (generally between 80 to 300ms with no overlap), whereas each *TSD* chroma represents pitch content on 36 dimensions for a constant audio frame (743ms with half overlap). The difference between these two approaches turns out to have major implications.

4.1.2 Alphabet definition

In its general definition, each B-dimensional chroma feature consists of B bins that may take any positive value, so their domain is infinite. However, as explained in Section 3.3, the heuristic alignment relies on the identification of exact similar regions in discrete sequences. Hence, it is critical for a BLAST approach to first project representing sequences onto a finite alphabet. A natural quantization is to detect the most probable chord, by looking for the highest scoring triad (root, major/minor third, fifth), and assigning a symbol corresponding to the index of this best triad. We represent each chroma feature over a 12-letter alphabet $\Delta = \{a, b, c, \ldots, k, l\}$ by the root of the predominant chord played. The chord mode (major/minor) is not taken into account since a root-exclusive representation seems to provide sufficiently meaningful audio representations (see Section 4.2). The 36-dimensional chroma features are also reduced down to 12 possible root chord symbols by keeping the multiple of 3 closest to the index of the highest triad (reduction to a 12 note scale with robustness to de-tuning).

[1] See http://www.labri.fr/perso/bmartin/ISMIR12 for the list
[2] http://labrosa.ee.columbia.edu/millionsong/
[3] http://www.secondhandsongs.com
[4] MHRAF submissions in Struct.Seg-11, Cover song-10
[5] http://the.echonest.com/

		TSD		*MSD*	
	Seed size	False negative	False positive	False negative	False positive
(i)	3	0.00	8.91	0.11	4.70
	4	0.03	3.69	4.28	1.15
	5	0.84	1.68	18.3	0.39
	6	4.50	0.82	37.2	0.16
	7	11.7	0.44	54.7	0.08
	8	21.8	0.25	68.6	0.04
(ii)	3	0.00	1.06	0.84	1.33
	4	0.21	0.41	7.08	0.23
	5	2.59	0.18	22.6	0.06
	6	6.67	0.08	41.8	0.02
	7	14.9	0.04	58.9	0.01
	8	26.0	0.02	72.1	.003

Table 1. Sensitivity/specificity tradeoff on *TSD* and *MSD*. Scores are given as percentages. In (i), all words are indexed in the dataset. In (ii), mono-symbolic words are not indexed.

This projection is likely to introduce inconsistencies in compared sequences, due to such a simplistic analysis. However, in practice the method only requires small sections of aligned sequences to be identical in order to assess their similarity, and is tolerant to sparse analysis errors to some extent.

4.1.3 Transposition invariance

Transposition is a very common variation among cover versions. Either globally on an entire rendition or locally accross structural parts, transpositions have to be taken into account in similarity analysis [17]. As highlighted in [14], the indexing strategy should hash harmonic progressions instead of absolute pitch content. While looking for exact matches between two sequences, compared regions are thus transposed down to a common key. For instance, the sequences `bcbbfd` and `deddhd` are in fact an exact match since they describe the same chord variations regardless of their local key, which differ by a major second. Practically, this can be seen as translating all words such that they start with the symbol `a`.

4.2 Seed determination

The heuristic alignment strongly relies on the selection of meaningful parts. A key issue for the method is to determine a seed both *sensitive* enough to correctly index alignments between cover songs, and *specific* enough to index as few spurious hits as possible. The first parameter for optimizing the sensitivity/specificity tradeoff is the length of the seed. In the following, given a seed length N, we denote by Δ^N the set of all possible words of length N over Δ.

4.2.1 Sensitivity evaluation

Assessing the sensitivity of a seed can be done by analyzing the alignments of similar sequences. Let N be a seed length. We must determine how many alignments actually contain seeds of size N, *i.e.* how many alignments contain at least one run of N matches. Thus, we first computed on both *TSD* and *MSD* local alignments between cover songs using the local transposition variant described in 3.2.2. By tracing back the alignments, we were able to identify exact runs of matches. An alignment is considered validated if

TSD		*MSD*	
Word	%	Word	%
aaaaaaa	6.36	aaaaaaa	14.1
ahhhhhh	0.55	aaaaaah	1.45
aaaaaah	0.53	aaaaaaf	1.31
affffff	0.49	affffff	1.13
aaaaaaf	0.46	ahhhhhh	0.98

Table 2. The five most probable words in *TSD* and *MSD* and their frequency of occurrence (in % of the words in each database) for $N = 7$ and a 12-letter alphabet.

it can be indexed, *i.e.* if at least one long enough run is found. The second and fourth columns of Tab. 1-(i) show the probability Pr[false negative] that an alignment can not be indexed by the method, as a function of the seed length, on both of the datasets.

4.2.2 *Specificity evaluation*

To evaluate the specificity of a seed, we need to estimate the probability of finding two identical runs in unrelated audio sequences. Practically, for a given word w over Δ^N, we count the number of occurrences of w in the database of unrelated sequences (no cover songs). This computation is stored in a list L and repeated for each possible word. In the end, $L[j]$ contains the number of instances of a particular word, and $\sum_i L[i]$ is the total number of N-long words in the dataset. The probability of finding *one* word w in a random chunk of the database is given by $\frac{L[j]}{\sum_i L[i]}$, where j is the index in L corresponding to w. Subsequently, the overall probability of finding *two* identical words in the database is given by $Pr[\text{false positive}] = \frac{1}{(\sum_i L[i])^2} \sum_j L[j]^2$. The third and fifth columns of Tab. 1-(i) provides the Pr[false positive] computed for each seed length and for both of the datasets, as percentages.

4.2.3 *Word distribution*

Table 2 shows the five most probable words in both datasets for a fixed seed length of 7. The most probable bin is the mono-symbolic word. Following bins correspond to frequent intervals in tonal music: up-by-fifth (a→h) or down-by-fifth (a→f). In both of the datasets, mono-symbolic words occur far more often than other words, representing 6.36% and 14.1% of the words in *TSD* and *MSD*, respectively. Thus, mono-symbolic words are likely to be responsible for many false positive hits while not capturing very sensitive regions in true alignments. We hence re-evaluated in Tab. 1-(ii) the sensitivity and specificity tradeoff on both datasets without indexing mono-symbolic words. As a result, for a seed length of 7 symbols, specificity is increased by a factor between 8 and 10, while sensitivity is reasonably decreased by around 3% in both datasets.

5. RESULTS AND DISCUSSION

Statistical results emphasize a significant difference between sequences in both datasets. First, cover song alignments share much fewer words in *MSD* than in *TSD*. For instance, for a seed size of 7, only 41.1% of the cover song alignments in *MSD* share common multi-symbolic words, as compared to 85.1% in *TSD*. Subsequently, the characterization of cover songs in *MSD* should be more difficult than in *TSD*. Moreover, false negative rates suggest that the distributions of words between both datasets are different, hence it would not be relevant to put them in common for evaluation purpose.

To test the relevance of our system in comparison with alignment methods, we implemented the alignment techniques described in 3.2.2. Since indexing is computed in a transposition invariant manner, we tested local alignments with the accurate local transposition variant [1]. We evaluated the identification technique using the Mean of Average Precision (MAP), the standard metric for cover song retrieval evaluation [8, 16, 17].

From every cover class in *TSD*, we computed alignments of each member to the rest of the class and to confusing songs. Average MAP values are presented in Tab. 3-(i), and detailed among 8 cover classes of *TSD* in Fig. 2. We experimented with the same approach on *MSD*, more precisely on a subset that made the alignment computation practicable. This subset MSD_{2k} was formed by randomly choosing 30 cover song classes and 2000 confusing songs from *MSD*. We discovered that the identification method did not extend, as indicated by the MAP scores (Tab. 3-(ii)). We identify two possible reasons: 1) Cover songs are particularly different from each other in *MSD*; 2) *MSD* chroma features are not as suitable as *TSD* features for sequence alignment. To isolate the second possibility, we recomputed *TSD* with the audio features used in *MSD*, via the EchoNest API[6]. We repeated the same cover identification experiment on this new dataset $\widetilde{TSD}$ and obtained MAP scores indicated in Tab. 3-(iii). The significant drop in MAP scores between *TSD* and $\widetilde{TSD}$ evaluations suggest that *MSD* chroma features do not make for alignable sequences. This result, already infered in [4], is substantiated by the high false negative rate found in *MSD* sequences (Tab. 1). We think this is due to critical implementation differences in chroma features between both datasets such as chroma dimension, temporal filtering and segment synchronism, which may not be adapted to standard alignment techniques, as highlighted in [16] for instance.

We implemented the indexing strategy described in 4 on *TSD* and *MSD* sequences. Logically, previous results made *MSD* heuristic alignments ineffective. *MSD* is to be considered here only as a computation performance indicator. From Tab. 1, we chose the seed length $N = 7$, that features a reasonable false negative rate of 11.7%, eliminating the most dissimilar cover songs, for a low false positive rate of 0.44% that guarantees high performance with few spurious hits. For each dataset, we built a table hashing every N words in sequences and storing their positions. Then, for each query, all matching songs were filtered using either $\mathcal{F}_1$ or $\mathcal{F}_1$ and then $\mathcal{F}_2$. Resulting MAP scores are given in 3-(iv) and (v). Highly depending on the dataset, our scores are not intended to be compared to state-of-the-art results. Their relevance lies in the comparison between basic method and heuristic alignments. As shown in Fig. 2, BLAST filters seem to slightly decrease the accuracy of the cover identification. Combining both filters seems effective for quickly identifying most covers. Indeed, the overall accuracy of BLAST method on *TSD* reaches a MAP score of 30.11%, corresponding to a loss of 14.71% as compared to an accurate sequence alignment.

[6] http://developer.echonest.com/

	Method	Dataset	MAP (%)	Runtime (s/query)
(i)	Alignment	*TSD*	44.82	129
(ii)		MSD_{2k}	5.71	388
(iii)		$\widetilde{TSD}$	7.20	273
(iv)		*MSD*	-	193,765
(v)	BLAST-$\{\mathcal{F}_1\}$	*TSD*	18.06	0.24
(vi)		*MSD*	-	12.20
(vii)	BLAST-$\{\mathcal{F}_1, \mathcal{F}_2\}$	*TSD*	30.11	0.33
(viii)		*MSD*	-	16.9

Table 3. MAP results and computing times for cover identification on *TSD* and *MSD*.

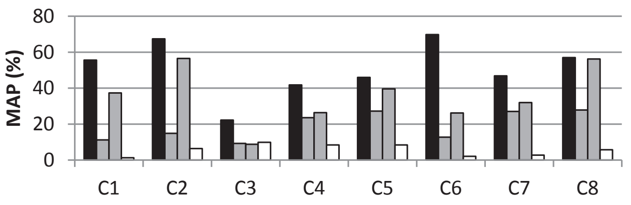


Figure 2. MAP scores obtained on 8 cover classes of *TSD*. Black: alignment, grey: BLAST-$\{\mathcal{F}_1\}$, BLAST-$\{\mathcal{F}_1, \mathcal{F}_2\}$, white: alignment with *MSD* chroma features ($\widetilde{TSD}$).

5.1 Computational efficiency

Due to the very high number of entries in the hash table, we implemented an efficient memory key/value lookup system in C. Building the index from chord sequences required about 16 minutes for the whole *MSD* on our server[7] . The average querying runtimes for each approach are given in Tab. 3. Note that we did not use parallel computing in this study. As expected, alignments imply slow computation, resulting in about 129 seconds per query on *TSD* and 388 seconds per query on MSD_{2k}. By counting the number of symbols in the whole *MSD* dataset, we infer that approximately 53 hours would be required to compute alignments. Note that removing the local transposition variant would speed-up by a factor of 12, still inadequate for practical computation. Thanks to BLAST heuristics, computation is drastically improved with around 12.2 seconds per query on *MSD* (0.24 seconds on *TSD*) with the coarse filter $\mathcal{F}_1$, and 16.9 seconds per query with both filters on *MSD* (0.33 seconds on *TSD*), on average.

6. CONCLUSION

We presented a new method for practical cover identification on large scale datasets. Inspired by bioinformatics heuristics, we applied BLAST to audio features and investigated the distribution of music sequences. Results obtained on our dataset suggest a reasonable loss of accuracy of the retrieval system in exchange for a substantial gain in computing time: estimating cover songs of a 3 minutes feature sequence can be acheived in less than 15 seconds in a million song database. We see this outcome as a significant step towards the practical search for approximate patterns in large datasets. Another main result of our study, although quite unexpected, is the apparent limitation of *MSD* chroma features regarding sequence alignment. Future studies on the *MSD* involving alignment techniques should investigate further the distribution of chroma sequences, and maybe combine them to other data (*e.g.* loudness, timbre). Future work will be particularly focused on enhancing the sensitivity of the identification of cover songs, considering for instance spaced or variable length seeds for BLAST.

7. REFERENCES

[1] J. Allali, P. Ferraro, P. Hanna, and C. Iliopoulos. Local transpositions in alignment of polyphonic musical sequences. In *String Processing and Information Retrieval*, volume 4726, pages 26–38. Springer Berlin / Heidelberg, 2007.

[2] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, D.J. Lipman, et al. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[3] S.F. Altschul, T.L. Madden, A.A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D.J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, 25(17):3389–3402, 1997.

[4] T. Bertin-Mahieux and D.P.W. Ellis. Large-scale cover song recognition using hashed chroma landmarks. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 117–120, 2011.

[5] T. Bertin-Mahieux, D.P.W. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proc. of the 12th International Conference on Music Information Retrieval*, 2011.

[6] 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, 2008.

[7] R.B. Dannenberg and N. Hu. Pattern discovery techniques for music audio. In *Proc. of the 3rd International Conference on Music Information Retrieval*, pages 63–70, 2002.

[8] J.S. Downie, M. Bay, A.F. Ehmann, and M.C. Jones. Audio cover song identification: Mirex 2006-2007 results and analyses. In *Int. Symp. on Music Information Retrieval (ISMIR)*, pages 468–473, 2008.

[9] E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, pages 63–100, 2006.

[10] D. Gusfield. *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press, pages 215–253, 1997.

[11] T. Jehan. *Creating Music by Listening*. PhD thesis, Massachusetts Institute of Technology, pages 57–59, 2005.

[12] J. Kilian and H.H. Hoos. Musicblast - gapped sequence alignment for mir. In *Proc. of the 5th International Conference on Music Information Retrieval*, pages 38–41, 2004.

[13] J.F. Kilian. *Inferring Score Level Musical Information From Low-Level Musical Data*. PhD thesis, Darmstadt University of Technology, pages 54–60, 2004.

[14] F. Kurth and M. Müller. Efficient index-based audio matching. *IEEE Trans. on Audio, Speech, and Language Processing*, 16(2):382–395, 2008.

[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. *Journal of Molecular Biology*, 48(3):443–453, 1970.

[16] J. Serrà, E. Gómez, and P. Herrera. *Audio cover song identification and similarity: background, approaches, evaluation, and beyond*, volume 274 of *Studies in Computational Intelligence*, chapter 14, pages 307–332. 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*, 16:1138–1151, 2008.

[18] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[7] Hardware: Intel Xeon X5675 @ 3.07GHz, 12M cache, 32GB RAM

A CROSS-CULTURAL STUDY OF MUSIC MOOD PERCEPTION BETWEEN AMERICAN AND CHINESE LISTENERS

Xiao Hu
Faculty of Education
The University of Hong Kong
xiaoxhu@hku.hk

Jin Ha Lee
The Information School
University of Washington
jinhalee@uw.edu

ABSTRACT

Music mood has been recognized as an important access point for music and many online music services support browsing by mood. However, how people judge music mood has not been well studied in the Music Information Retrieval (MIR) domain. In particular, people's cultural background is often assumed to be an important factor in music mood perception, but this assumption has not been verified by empirical studies. This paper reports on a study comparing mood judgments on a set of 30 songs by American and Chinese people. Results show that mood judgments do indeed differ between American and Chinese respondents. Furthermore, respondents' mood judgments tended to agree more with other respondents from the same culture than those from the other group. Both the song characteristics (e.g., genre, lyrical or instrumental) and the non-cultural background of the respondents (e.g., age, gender, familiarity with the songs) were analyzed to further examine the difference in mood judgments. Findings of this study help further our understanding on how cultural background affects mood perception. Also discussed in this paper are implications for designing MIR systems for cross-cultural music mood classification and recommendation.

1. INTRODUCTION

The number of studies on music mood has been increasing in the Music Information Retrieval (MIR) domain as many perceive music mood as a potential feature for organizing and recommending music. However, previous research asking people to provide mood tags for short music clips found that it is a highly subjective feature and the vocabulary of music mood varies widely among users [8]. In addition to the features inherent in music itself there are a number of features that can affect how people determine the mood of music (e.g., their current state of mind, life events). We believe that one important factor is the cultural context of the user. However, currently there are no cross-cultural studies that specifically compare how people perceive and determine the mood of music in the MIR domain.

In this study, we explore if users from China and the United States perceive music mood in different ways. We chose to compare these cultures for several reasons. First, China is a dominant Eastern culture while the United States is a dominant Western culture. Second, although the influence of American pop culture is gradually increasing in China, due to historical and political factors, Chinese people are far less affected by Western culture as compared to people from other East Asian countries such as Korea or Japan [13]. Third, one of the authors is fluent in Chinese and English, which is important as translating the mood labels while preserving the subtle nuances can be challenging for non-native speakers.

2. LITERATURE REVIEW

2.1 Cross-Cultural Studies in Music Psychology

There are a number of studies in music psychology on comparing responses on Western and non-Western music from Western and non-Western listeners but many of them focused on aspects such as memorability of music (e.g., [9]) or perception of complexity (e.g., [2]) rather than perception of music moods.

Balkwillm and Thompson [1] are among the first investigating whether judgments on music mood can transcend cultural boundaries. They recruited 30 Western listeners to judge 12 Hindustani raga excerpts in order to see if people can identify the intended emotion in music from an unfamiliar tonal system. Their findings showed that the emotions of joy, sadness and anger (but not peace) were identifiable by the listeners and the emotion judgments were significantly related to psychophysical characteristics of the pieces. However, their study did not compare judgments of people from different cultures.

Gregory and Varney [5] compared mood descriptors applied to Indian classical, Western classical, and new age music by Indian and European listeners and revealed "many subtle differences in affective response" in mood descriptors. They suggested that "the affective response to music is determined more by cultural tradition than by the inherent qualities of the music (p.47)." More recently, Wong et al. [14] found that Indian and Western listeners showed in-culture bias when judging the tension in Western and Indian music. However, Fritz et al. [4] found that native African (Mafa) listeners could recognize three basic emotions (happy, sad, scared/fearful) expressed in

Western music with above chance accuracy and suggested that, “the expression of these basic emotions in Western music can be recognized universally (pp.253).” The conflicting results in these studies highlight the need for more empirical research on cross-cultural music mood perception. In addition, these studies generally focused on classical and/or ethnic music whereas our study focuses on popular music.

2.2 Cross-Cultural Studies in Music Information Retrieval

In the MIR domain, there are few studies that examined cross-cultural aspects related to how users interact with and search for music. Lee et al. [7] collected music related questions from Q&A websites based in North America (i.e., Google Answers) and Korea (Naver Knowledge-iN) and did a comparative analysis. They found that Korean users experienced a number of challenges in cross-cultural/multilingual music searches: 1) they often failed to provide bibliographic metadata such as composer, performer or title in their queries, 2) they had difficulties in using Western music genres and instead relied on association-based concepts (i.e. where the music was used), and, 3) they had difficulties in using and transliterating lyrics information. The authors suggest that new access points for accommodating cross-cultural/multilingual music searching including associate metadata (i.e., usage) are necessary.

Nettamo et al. [11] also conducted a cross-cultural study of mobile music retrieval, management, and consumption behaviors of people in New York vs. Hong Kong. They found several differences in how music was being managed, shared, and used. For example, New Yorkers sought music information through various channels including blogs, websites, magazines, etc., and mood and context of use were factors affecting how they generated playlists. Hong Kong users, on the other hand, sought for music only through limited channels such as ranking websites or through friends, and did not use playlists at all.

Both of these studies indicate that there were in fact differences between users from different cultures with regards to their music related behaviors. However, none of these studies examined how they perceive music mood which is the gap this work is attempting to bridge.

3. STUDY DESIGN

This study focuses on the following set of research questions:

1. Do Americans and Chinese have different perceptions of mood on the same set of popular songs? In other words, do people from the same culture tend to agree more with each other?
2. Do some moods tend to be more agreeable among people from certain culture?
3. Do characteristics of the songs (i.e., genre, vocal or instrumental) affect the difference on mood judgments between Americans and Chinese?
4. Do users’ non-cultural background (i.e., gender, age, and familiarity with the songs) affect the difference on mood judgments between Americans and Chinese?

To answer these questions, we created an online survey in which each user listened to thirty 30-second music clips and selected the most appropriate mood cluster among the five given clusters for each piece. We used the same mood clusters that are used in MIREX[1] Audio Mood Classification (AMC) Task [6] (reprinted in Table 1). Users can choose “other” if they think none of the mood clusters is applicable to the music piece.

Cluster 1 (C_1)	passionate, rousing, confident, boisterous, rowdy
Cluster 2 (C_2)	rollicking, cheerful, fun, sweet, amiable/ good natured
Cluster 3 (C_3)	literate, poignant, wistful, bittersweet, autumnal, brooding
Cluster 4 (C_4)	humorous, silly, campy, quirky, whimsical, witty, wry
Cluster 5 (C_5)	aggressive, fiery, tense/anxious, intense, volatile, visceral

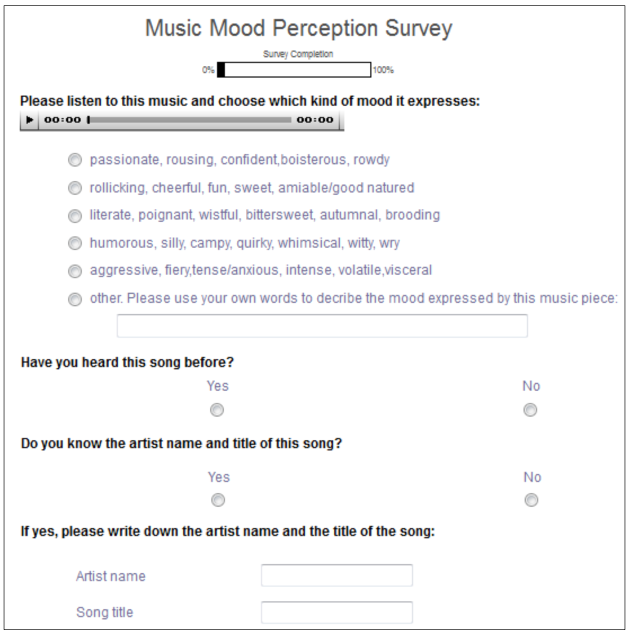
Table 1. Five Mood Clusters used in MIREX [6].

Half of the 30 pieces were selected from the MIREX AMC task test collection with the help of IMIRSEL. This AMC test collection was created based on the APM (Associated Production Music)[2] collection, and covers a variety of different music genres. The mood of each piece in the AMC test collection was judged by three MIREX evaluators whose cultural backgrounds ranged from Europe, America and Asia [6]. In order to avoid including the “obvious” examples that received high agreement from evaluators in the test data set, we selected the songs for which there was greater disagreement among the MIREX evaluators. As these pieces were instrumental without a vocal part, we balanced our test set by drawing the other half of the 30 pieces from the USPOP collection [3] and ensured that they all had vocal components. The songs from the USPOP collection were chosen similarly to the APM songs; we selected songs that had greater disagreement in the mood judgments from six IMIRSEL members (c.f., [6]).

We recruited two user groups for the survey: people who were raised in Mainland China and considered themselves “Chinese”, and people who were raised in the United States, and considered themselves “American”. All subjects were recruited from large universities in the U.S. The survey was deployed in both Chinese (Mandarin) and English; all the mood labels in English were translated into Chinese by the first author for the Chinese survey. In both surveys, we asked the users if they had heard the

[1]Music Information Retrieval Evaluation eXchange is the annual evaluation campaign for various music information retrieval algorithms hosted by the International Music Information Retrieval Systems Evaluation Lab (IMIRSEL) at the University of Illinois at Urbana-Champaign.
[2] http://www.apmmusic.com/pages/aboutapm.html

music clip before and if they could name the artist and the song title in order to gauge their familiarity with the songs. Figure 1 shows the screenshot of the online survey in English.



Music Mood Perception Survey

Survey Completion

0% 100%

Please listen to this music and choose which kind of mood it expresses:

00:00 00:00

- passionate, rousing, confident,boisterous, rowdy
- rollicking, cheerful, fun, sweet, amiable/good natured
- literate, poignant, wistful, bittersweet, autumnal, brooding
- humorous, silly, campy, quirky, whimsical, witty, wry
- aggressive, fiery,tense/anxious, intense, volatile,visceral
- other. Please use your own words to decribe the mood expressed by this music piece:

Have you heard this song before?

Yes No

Do you know the artist name and title of this song?

Yes No

If yes, please write down the artist name and the title of the song:

Artist name

Song title

Figure 1. Survey interface for American listeners

4. DATA AND DISCUSSION

4.1 Overview

There were a total of 55 responses from Chinese and 45 from Americans; however, not all responses were complete. 31 listeners completed the Chinese and American surveys, respectively, for a total of 62 complete responses. Table 2 shows the demographic information of the selected respondents. Among the Chinese respondents; 23 of them had been living in the U.S. for less than 2 years, 3 had been in the U.S. for 3-5 years, 1 for 6-8 years, and 4 for 9-11 years.

Cultural background	Age			Gender	
	Min	Max	Avg.	Male	Female
American	22	55	31.8	6	25
Chinese	19	46	26.2	10	21

Table 2. Demographics of survey respondents

In the following data analysis, chi square (χ^2) statistics are used to test whether distributions of two categorical variables (e.g., cultural background, mood categories) are independent from each other [12] unless noted otherwise. In the following subsections, we will answer each research question based on analysis of the survey responses.

4.2 Difference Between Cultural Groups

4.2.1 Mood Judgments on All Songs

Figure 2 shows the distribution of mood judgments of the two groups on all 30 songs. Chinese users selected mood Cluster 1 (*passionate, rousing, confident, boisterous, rowdy*) and Cluster 3 (*literate, poignant, wistful, bittersweet, autumnal, brooding*) more often than Americans whereas Americans chose Cluster 2 (*rollicking, cheerful, fun, sweet, amiable/good natured*) more often than Chinese. Both groups had similar numbers of judgments on Cluster 4 (*humorous, silly, campy, quirky, whimsical, witty, wry*) and Cluster 5 (*aggressive, fiery, tense/anxious, intense, volatile, visceral*). More Americans than Chinese chose the "other" option. A chi square test indicates that listeners' selection of mood clusters significantly depends on the cultural group they belong to ($\chi^2 = 73.64$, df = 5, $p < 0.0001$). In other words, there was a significant difference between Americans and Chinese when examining their mood judgments as a whole. A follow-up Tukey multiple comparison test [15] showed that judgments of the two cultural groups on the "other" cluster were significantly different from those on all the other clusters (at $p < 0.05$), judgments on Cluster 2 were significantly different from those on Cluster 1 and 3 (at $p < 0.05$), and there were no significant differences between judgments on other pairs of mood clusters.

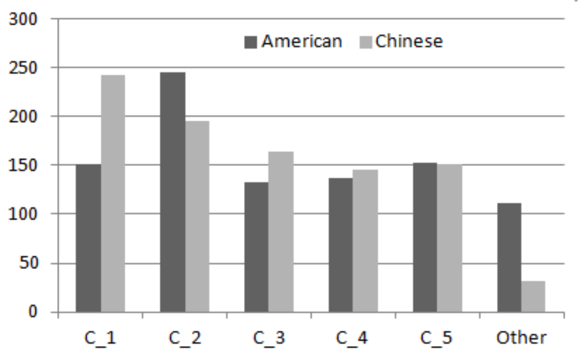


Figure 2. Distribution of mood judgments of the two groups

4.2.2 Agreement on Mood Judgments

In order to find out whether listeners from the same cultural group would agree more with each other than with listeners from another cultural group, we calculated the level of agreement on mood judgments among individual listeners. For categorical data such as the mood judgments, agreement is typically calculated based on the Sokal-Michener coefficient, which is a ratio of the number of pairs with the same values and the total number of variables (songs in this case) [12]. For instance, if two listeners *i* and *j* had the same mood judgment on 15 of the 30 songs, the agreement ratio between them will be 0.5. Table 3 shows the average agreement ratio among pairs of listeners within and across cultural groups in this study. Within each cultural group, users show 0.35 agreement rate. However, across cultural groups the agreement rate drops to 0.30. A non-pair wise *t*-test was conducted to test the significance of the difference on agreement ratio within each cultural group and across cultural groups. Both tests revealed a statistically significant difference. Therefore, our data support the hypothesis that listeners tended to agree more with others from the same cultural background than those from another cultural background.

	American	Chinese	T statistics	*p*-value
American	0.35	0.30	11.44	<0.001
Chinese	0.30	0.35	12.24	<0.001

Table 3. Average agreement ratio within and between cultural groups

4.3 Mood Clusters vs. Cultural Groups

We also investigated which mood clusters received higher agreement from people in each cultural group. We examined all pairs of responses from each group. Since there were 31 responses from each group, there were a total of 465 pairs of responses within each group. Each response had 30 mood judgments, thus there were 13,950 pairs of judgments in each group. Between the two cultural groups, there are 31 * 31 = 961 pairs of responses and 961 * 30 = 28,830 pairs of judgments. Table 4 lists the number of agreed pairs of judgments on each mood cluster within each cultural group and across cultural groups. It shows that American listeners agreed more on Cluster 2 and 5 while Chinese listeners agreed more on Cluster 1 and 3. The difference between the two groups is statistically significant ($\chi^2 = 668$, df = 5, $p < 0.0001$).

	C_1	C_2	C_3	C_4	C_5	Other	Total
American	706	1477	778	587	1094	270	4912
Chinese	1355	995	1203	443	894	11	4901
Across	1704	2122	1713	881	1999	131	8550

Table 4. Number of agreed pairs of judgments across mood clusters

4.4 Song Characteristics vs. Cultural Groups

Half of the test songs were instrumental and the other half were vocal. The two cultural groups showed significant difference in judging the mood for both instrumental ($\chi^2 = 88.09$, df = 5, $p < 0.0001$) and vocal songs ($\chi^2 = 28.98$, df = 5, $p < 0.0001$). Table 5 shows the agreement ratios among all judgment pairs on instrumental and vocal songs. Lyrics definitely seem to help achieve a higher agreement for Americans while they have essentially no effect on Chinese (it should be noted that all lyrics were in English). In addition, cross-culturally the two groups were more likely to provide different judgments on instrumental pieces than vocal ones. As discussed in [8], even if Chinese listeners cannot comprehend the lyrics as well as American listeners, the delivery of the singer may still affect how they determine the mood of the song.

	Instrumental	Vocal	All
American	0.28	0.41	0.35
Chinese	0.36	0.35	0.35
Across	0.25	0.34	0.30

Table 5. Agreement ratio on instrumental vs. vocal songs

We also looked at the genres of the songs as provided by APM and USPOP. Table 6 shows the genre distribution of the songs as well as instrumental vs. vocal information. Dance and Easy-listening songs were all instrumental while songs in the remaining genres were mostly vocal. For each of the five genres, mood judgments were significantly dependent on cultural groups ($\chi^2 = 21.91 \sim 46.68$, df = 5, $p < 0.001$). Table 6 also shows the agreement ratios across genres. As it can be seen, Americans agreed more on Pop songs whereas Chinese agreed more on songs in Other and Easy-listening. Cross-cultural agreement levels are generally lower than those within cultural groups. Among all the genres, Dance and Easy-listening songs had the least cross-cultural agreement.

	Dance	Easy-listening	Pop	Rock	Other	Total
Instru.	4	5	2	1	3	15
Vocal	0	0	5	7	3	15
American	0.30	0.29	0.46	0.35	0.31	0.35
Chinese	0.29	0.38	0.32	0.35	0.41	0.35
Across	0.22	0.28	0.33	0.31	0.30	0.30

Table 6. Song distribution and agreement ratio across genres

4.5 Listener Characteristics vs. Cultural Groups

The aggregated mood judgments across songs and cultural groups were still statistically significant when we consider the gender of the listeners (i.e., Chinese male vs. US male, $\chi^2 = 18.28$, df = 5, $p = 0.0026$; Chinese female vs. US female, $\chi^2 = 52.83$, df = 5, $p < 0.0001$).

As Table 2 shows, the Chinese respondents in this study were generally younger than the American respondents. To minimize the possible influence of age on mood judgments, we compared the answers from listeners of the same age range (22-46 years old) in both culture groups (24 Chinese and 28 American). The mood judgments of two cultural groups were still significantly different ($\chi^2 = 61.85$, df = 5, p < 0.0001).

In this study, a listener's familiarity with a song is measured by their answers to two questions: 1) whether he or she had heard the song before; and 2) whether he or she can identify the artist name and song title. A "no" answer to both questions indicates low familiarity, a "yes" to both questions indicates high familiarity, and a "yes" and a "no" indicates medium familiarity. The reason for using these two questions instead of directly asking the listeners their level of familiarity is because people may have different interpretations on song familiarity. Some people might consider a song familiar if it invokes any memory while other people might not think it is familiar unless they could actually sing part of the song. The two questions are objective, and thus are easier to answer and avoid personal biases. Table 7 shows the distribution of the level of familiarity across Americans and Chinese. As the test songs were Western songs, it is not surprising that American listeners were more familiar with the songs than Chinese listeners.

In order to see whether the level of familiarity has an effect on mood judgment agreement, we calculated the agreement ratio with various combinations of familiarity levels in each cultural group as well as across cultural groups. Each cell in Table 8 shows the agreement ratio among all judgment pairs with corresponding familiarity levels.

	Unfamiliar	Medium	Familiar	N/A	Total
American	617	120	192	1	930
Chinese	836	75	14	5	930

Table 7. Distribution of the level of familiarity

		Unfamiliar	Medium	Familiar
American	Unfamiliar	0.32	0.36	0.36
	Medium	-	0.43	0.44
	Familiar	-	-	0.44
Chinese	Unfamiliar	0.35	0.37	0.23
	Medium	-	0.36	0.22
	Familiar	-	-	0.24
Across	Unfamiliar	0.28	0.31	0.29
	Medium	-	0.37	0.33
	Familiar	-	-	0.24

Table 8. Agreement ratio across different levels of familiarity

For American listeners, being familiar with the songs did improve the odds of agreeing. However, medium and high familiarity did not appear to have much effect on agreement. For Chinese listeners, having heard the songs before (medium familiarity) slightly increased the agreement ratio, but high familiarity with the songs appears to actually decrease agreement, which is unintuitive. We suspect that this might be due to the sparseness of the samples: there were only 14 (out of 930) cases where the Chinese listeners were highly familiar with a song [Table 7]. It may also indicate that a mere identification of title and/or artist name from Chinese listeners does not imply that they understood what the song was about. Table 8 also shows that the agreement ratios between Americans and Chinese are lower than those of Americans.

5. DISCUSSION

Our analysis suggests that there is in fact a significant difference between how Americans and Chinese perceived music mood. From the total number of mood judgments across mood clusters, Chinese listeners chose Cluster 1 more often than American listeners. We conjecture that this difference may be attributed to the differences between Chinese and Western cultures. In Chinese culture, people tend to restrain the expression of feelings and Chinese people are generally more introverted compared to Western people [9], and thus may be more likely to think a Western music piece is "passionate," "rousing," or "boisterous" (Cluster 1). Previous research also found that Chinese value low-arousal positive affect (e.g., calm) whereas Westerners value high-arousal positive affect [11]. This may help explain the higher responses on Cluster 3 for Chinese listeners and Cluster 2 for American listeners. When the mood of the song is not clear, people may end up selecting moods that they generally prefer since they are more likely to focus on those moods; in other words, they hear what they want to hear.

It is also interesting to see fewer judgments of "other" among Chinese users. This may be related to the collectivism commonly seen in Eastern cultures and the individualism in Western cultures. Chinese listeners used one of the given five mood clusters 96.6% of the time. However, American listeners disagreed with the presented mood clusters more often, using the "other" option for 11.9% of their judgments, more than 3 times as often as Chinese.

These findings have implications for designing MIR systems for people with different cultural backgrounds. For example, a mood classification system may be designed so that it treats songs with mixed moods in a different way: categorizing them into Cluster 2 for Americans vs. Cluster 1 or 3 for Chinese reflecting their expectations. In addition, while we generally think it is more user-friendly to let users browse for music with different moods than asking users to search with their own mood terms, this would appear to be even more critical for Chinese listeners, as they seem to prefer using given organizational structures rather than providing their own input (via the "other" option as shown in Figure 1).

Among all the 30 test songs, the one with the highest disagreement between Americans and Chinese was *Got to get you into my life* by The Beatles. Figure 3 shows the judgment distribution across mood clusters for this song. Most of the Americans answered Cluster 2 while the Chinese' answers were spread out across multiple mood clusters with most answers in Cluster 4. A closer look at the data revealed that none of the Chinese listeners had listened to this song prior to this survey whereas 29 out of 31 American listeners had listened to this song and 19 of them were familiar enough to this song that they could name the artist. Cultural background evidently played an important role in mood judgments on this song. The Beatles are a symbol of the Western pop culture. Western listeners had probably been influenced by their background knowledge of the song and the band that this song *should* express a rollicking and cheerful (Cluster 2) or passionate (Cluster 1) mood. It is also possible that they were able to provide a mood judgment based on the whole song rather than the 30 seconds clip provided in the survey. In contrast, the Chinese listeners had no prior influence on how *others* had felt about this song or rest of the Beatles' songs, thus their answers were not as consistent as those from the American listeners.

Lyrics seem to affect how the Americans judge the mood of the songs, but not the Chinese [Table 5]. Although most of the Chinese listeners in this study could understand English, the music pieces in the survey are probably too short (30 seconds) for them to fully comprehend the lyrics and use them in mood judgments. Genre also affects people's agreement on music moods, but in different ways for different cultural groups. The fact that Americans agreed more on Pop songs is possibly related to their being familiar with the songs.

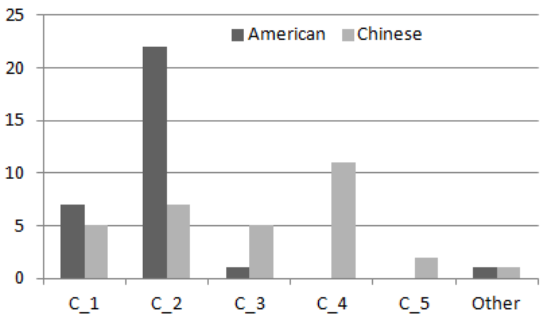


Figure 3. Mood judgment distributions for *Got to get you into my life*

As previously discussed, listeners' familiarity with the songs affects the level of agreement on mood judgments, but the influence is much stronger within the same cultural group. This makes it challenging to build an MIR system for users with cultural backgrounds that are different from the particular culture the music is from. For cross-cultural MIR systems, perhaps more flexibility should be provided to users. It may help to allow users to provide annotations so that they can complement the given "correct" mood labels. In such systems, it will be possible for users to assign multiple mood labels to a song, change a song's mood labels, or add alternative labels to the songs.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented a study comparing mood judgments on a common set of Western music pieces by American and Chinese listeners. Listeners from the two cultural groups indeed have different mood judgments and they tended to agree more with users from the same cultural group. Some genres seemed to be more difficult to reach user agreement across two groups, although further studies with a larger music samples should be conducted to validate the result. The cultural difference persists even when we consider the age and gender. The listeners' familiarity with the songs had a positive influence on the agreement level among users from the same cultural background of the songs. Findings of this study not only help further our understanding on how cultural background affects mood perception, but also have implications for designing cross-cultural MIR systems.

It should be noted that the Chinese respondents in this study have lived in the U.S., but there were not enough data to analyze the influence of this factor on their music mood judgments. In our future study, we will collect responses from Chinese people in China and compare the results. We will also investigate why people assign tracks to certain mood clusters by conducting in-depth interviews. In addition, we plan to increase the diversity of our user group by including users from countries other than China and United States such as Korea. Although Korea also represents non-Western culture, and Chinese and Korean cultures historically share a great deal of similarities, Korea is much more heavily influenced by American pop culture than China. Thus, comparing user groups from these countries may provide insights into how the exposure to other pop culture can affect the way people perceive the mood of music.

7. REFERENCES

[1] L. Balkwillm and W. F. Thompson: "A Cross-cultural investigation of the perception of emotion in music: psychophysical and cultural cues," *Music Perception*, Vol. 17, No. 1, pp. 43-64, 1999.

[2] T. Eerola, T. Himberg, P. Toiviainen and J. Louhivuori: Perceived complexity of Western and African folk melodies by Western and African listeners, *Psychology of Music*, Vol. 34, No. 3, pp. 337-371, 2006.

[3] D. Ellis, A. Berenzweig and B. Whitman: "The USPOP 2002 Pop Music Data Set," Retrieved from http://labrosa.ee.columbia.edu/projects/musicsim/uspop2002.html, 2003.

[4] T. Fritz, S. Jentschke, N. Gosselin, D. Sammler, I. Peretz, R. Turner, A. D. Friederici and S. Koelsch: "Universal recognition of three basic emotions in music," *Current Biology*,Vol.19, pp.573–576, 2009.

[5] A. H. Gregory and N. Varney: "Cross-cultural comparisons in the affective response to music," *Psychology of Music*, Vol. 24, pp. 47-52, 1996.

[6] X. Hu, J. S. Downie, C. Laurier, M. Bay and A. F. Ehmann: "The 2007 MIREX Audio Mood Classification task: Lessons learned," *Proceedings of the 9th International Society for Music Information Retrieval (ISMIR) Conference*, pp. 462-467, 2008.

[7] J. H. Lee, J. S. Downie, and S. J. Cunningham: "Challenges in cross-cultural/multilingual music information seeking," *Proc. of ISMIR*, pp. 1-7, 2005.

[8] J. H. Lee, T. Hill, and L. Work: "What does music mood mean for real users?" *Proceedings of the iConference*, 2012.

[9] S. J. Morrison, S. M. Demorest and L. A. Stambaugh: "Enculturation Effects in Music Cognition: The Role of Age and Music Complexity," *Journal of Research in Music Education*, Vol. 56, No. 2 pp. 118-129, 2008.

[10] R. R. McCrae, P. T. Costa, Jr. and M. Yik: "Universal aspects of Chinese personality structure," In M. H. Bond (Ed.) *The handbook of Chinese psychology*. Hong Kong: Oxford University Press, pp.189-207, 1996.

[11] E. Nettamo, M. Norhamo, and J. Häkkilä: "A cross-cultural study of mobile music: Retrieval, management and consumption," *Proceedings of OzCHI 2006*, pp. 87-94, 2006.

[12] R. R. Sokal and C. D. Michener: "A statistical method for evaluating systematic relationships," *University of Kansas Science Bulletin*, Vol. 38, pp. 1409–1438, 1958.

[13] J. L. Tsai, B. Knutson and H. H. Fung: "Cultural variation in affect valuation," *Journal of Personality and Social Psychology*, Vol.90, No.2, pp. 288–307, 2006.

[14] P. C. M. Wong, A. K. Roy and E. H. Margulis: "Bimusicalism: The implicit dual enculturation of cognitive and affective systems," *Music Percept*, Vol. 27, No. 2, pp. 81–88, 2009.

[15] J. H. Zar: *Biostatistical Analysis*, Fourth Edition, Prentice Hall, 1999.

SHORTEST PATH TECHNIQUES FOR ANNOTATION AND RETRIEVAL OF ENVIRONMENTAL SOUNDS

Brandon Mechtley
Arizona State University
Computer Science (SCIDSE)
bmechtley@asu.edu

Perry Cook
Princeton University
Computer Science and Music
prc@cs.princeton.edu

Andreas Spanias
Arizona State University
Electrical Engineering (SECEE)
spanias@asu.edu

ABSTRACT

Many techniques for text-based retrieval and automatic annotation of music and sound effects rely on learning with explicit generalization, training individual classifiers for each tag. Non-parametric approaches, where queries are individually compared to training instances, can provide added flexibility, both in terms of robustness to shifts in database content and support for foreign queries, such as concepts not yet included in the database. In this paper, we build upon prior work in designing an ontological framework for annotation and retrieval of environmental sounds, where shortest paths are used to navigate a network containing edges that represent content-based similarity, semantic similarity, and user tagging data. We evaluate novel techniques for ordering query results using weights of both shortest paths and minimum cost paths of specified lengths, pruning outbound edges by nodes' K nearest neighbors, and adjusting edge weights depending on type (acoustic, semantic, or user tagging). We evaluate these methods both through traditional cross-validation and through simulation of live systems containing a complete collection of sounds and tags but incomplete tagging data.

1. INTRODUCTION

1.1 Multiclass and non-parametric retrieval

Many techniques for text-based retrieval or classification of audio signals are parametric in nature, relying on explicit generalization, where individual classifiers are created for each label. For example, classification systems have been built for automatic record reviews [18], onomatopoetic labels [9], and genre [17], emotion [10], and instrumentation [5,7] identification. These systems make use of techniques such as one-versus-all discrimination [18], training each label with a support vector machine (SVM) classifier [1, 9], and learning a separate gaussian mixture model (GMM) for each label [16, 18].

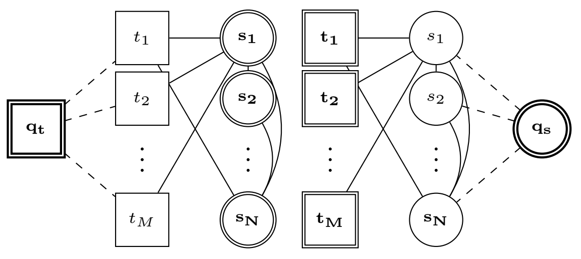
These multiclass methods benefit from constant query time complexity independent of the number of training instances, in that it is only necessary for each query (such as a sound, in the case of annotation) to be measured against each classifier. For specific, relatively stationary label domains, such as musical genres, this can be seen as a great benefit, especially when the number of sounds greatly exceeds the number of labels. However, there are many cases where non-parametric models can provide additional flexibility and robustness. One such case involves the presence of multiple types of information beyond acoustic feature vectors and annotations. For example, [19] and [13] describe methods where similarity between semantic concepts can assist in retrieval and annotation using tags not yet seen in training data. In large-scale systems with more complete tag sets, this may be less of a problem, but in live databases with incomplete tagging where no large-scale training database exists beyond user activity, as in the case of Freesound [1] , retrieval results can often come up empty.

Non-parametric (also known as similarity- or instance-based [6,11]) schemes compare each query to instances in a live database rather than having distinct training and production / evaluation stages. For example, [2], [3], and [12] use K-nearest-neighbors retrieval, where unlabeled sounds are annotated with tags belonging to their nearest neighbors in an acoustic feature space. [15] and [2] build two separate hierarchical cluster models—one for retrieval and one for annotation.

1.2 Associative retrieval

Graph-based techniques are often used for search in semantic and other associative networks. One technique that has seen much use is spreading activation. In spreading activation, an initial node (a query) is labeled with some weight, and this weight is spread to neighboring nodes with some decay. Spreading activation has been used in information retrieval applications, where nodes correspond to documents and terms [4]. Shortest paths are also of interest in associative retrieval. [20] introduces a graph-based framework where sounds are connected to tags through user activity and sounds are fully connected via acoustic similarity estimated by an HMM-based query-by-example algorithm described in [21]. New queries are immediately connected to other sounds or tags (either through acoustic or semantic similarity via the WordNet::Similarity library [14]), and shortest path distances using all nodes are

[1] Freesound: http://freesound.org/



(a) Text-based retrieval (b) Annotation

Figure 1: Two different possible query tasks with a single retrieval network. q_s and q_t represent a sound query and a tag query, respectively, and $t_1, t_2, ..., t_M \in T$ and $s_1, s_2, ..., s_N \in S$ represent tag and sound nodes already in the database. The query node and the subset of nodes over which the user is querying is marked in bold, and on-demand edge weights between the query and its respective class of nodes (sounds or tags) are marked with dashed lines. Note that S forms a clique.

used to rank retrieval results.

The results of [19] demonstrate that including semantic similarity to account for tags foreign to the training set can assist in annotation and text-based retrieval both for a random subset from the Freesound library, where tags are associated to sounds in a binary manner, and a smaller comprehensive user study, where each sound is tagged by multiple users. In this paper, we build upon this graph-based technique and perform a more in-depth study of its properties. Namely, we seek to a) evaluate the system using both traditional cross-validation and simulations of real-world systems with complete sound and tag sets but incomplete tagging data, b) demonstrate the effectiveness of adding a shortest-path algorithm to any existing tag-based query system, regardless of the presence of acoustic and semantic similarity measures, c) improve shortest-path retrieval performance by pruning network edges to nodes' K nearest neighbors, and d) explore the impact of assigning different weights to the importance of acoustic, semantic, and user-provided information.

2. SHORTEST PATH RETRIEVAL

2.1 Network structure

Formally, the network structure for retrieval and annotation takes the form of a weighted, undirected graph, $G = (V, E)$, where $V = S \cup T$ and S and T represent sets of sound and tag nodes. The graph edges, E, can be partitioned into three disjoint subsets, $E_{SS} \subseteq S \times S$, $E_{ST} \subseteq S \times T$, and $E_{TT} \subseteq T \times T$, representing acoustic, user-provided, and semantic information. The weighting function is denoted by $w : E \rightarrow \mathbb{R}^+$. This type of network structure can be adapted to different domains, such as music or even text documents, but for the sake of this paper, we focus on the task of retrieving and annotating environmental sounds. We therefore assume that sounds take the form of short audio clips representing individual sonic events. For more discussion on the concept of a "sound event," see the related discussion in [21]. In the following sections, we will discuss how the weights for E_{SS}, E_{ST}, and E_{TT} are calculated.

2.1.1 Sound-to-sound weights (E_{SS})

Sound-to-sound weights can be computed by comparing the acoustic content of each sound. For this task, we use the Sirens library [2] . For detailed information of how sound similarity is computed in Sirens and an evaluation of its performance, see [21]. A summary is as follows:

Sirens begins with acoustic feature extraction, where six features are calculated on overlapping 40ms Hamming windows hopped every 20ms. The feature trajectory for a sound file is given by $Y_{1:T}^{(1:F)}$, where $Y_t^{(i)}$ is the i-th feature's value at frame t.

The six features used by Sirens include *loudness*, the dB-scaled RMS level over time; *temporal sparsity*, the ratio of l^∞ and and l^1 norms calculated over all windowed RMS levels in a one-second interval; *spectral sparsity*, the ratio of l^∞ and l^1 norms calculated over short-time Fourier transform (STFT) magnitudes; *spectral centroid*, the bark-weighted average spectral content of a sound at any point in time; *transient index*, the l^2 norm of the difference of Mel frequency cepstral coefficients (MFCC) between consecutive frames; and *harmonicity*, a probabilistic measure of whether or not the signal comes from a harmonic source. This feature set was developed for a broad range of environmental sounds rather than any specific class of sounds and with a focus on ecological validity such that the features would best relate to qualities of human audition by using perceptual scalings of spectral content [21].

To compare sounds, [21] describes a method of estimating $L(s_i, s_j) = -\log P(Y_{1:T}^{(1:F)}(s_i)|\lambda^{(1:F)}(s_j))$, the log-likelihood of the feature trajectory of sound s_i being generated from the hidden Markov model (HMM), $\lambda^{(1:F)}(s_j)$, built to approximate the simple (i.e. constant, linear, or quadratic) feature trends of sound s_j. For retrieval in our undirected graph, however, it is helpful to have a semi-metric between sounds that is symmetric and nonnegative. In [8], a semi-metric that holds these properties is given:

$$\begin{aligned} w(s_i, s_j) =& L(s_i, s_i) + L(s_j, s_j) - \\ & L(s_i, s_j) - L(s_j, s_i). \end{aligned} \tag{1}$$

2.1.2 Sound-to-tag weights (E_{ST})

Letting $U_{|S|\times|T|}$ be a votes matrix where U_{ij} is equal to the number of users who have tagged sound s_i with tag t_j, we can compute the joint probability of s_i and t_j as

$$P(s_i, t_j) = \frac{U_{ij}}{\sum_{k,l} U_{kl}} \tag{2}$$

$$w(s_i, t_j) = -\log P(s_i, t_j). \tag{3}$$

2.1.3 Tag-to-tag weights (E_{TT})

The results from [13] and [19] demonstrate that semantic similarity (tag-to-tag edges) obtained from WordNet::-

[2] Sirens (Segmentation, Indexing, and Retrieval of Environmental Sounds): http://github.com/plant/sirens

Similarity [14] scores can be useful when performing text-based retrieval queries using tags not in the database or annotating sounds with these foreign tags. In general, however, it was found that including tag-to-tag links between in-network tags can hinder performance, so we have excluded these links in this paper, as we are chiefly interested in studying the effects of different shortest-path retrieval strategies rather than the source of the weights themselves.

2.2 Shortest path retrieval

Given the structure of the graph, $G(V, E)$ and its weights, w, as defined in the previous section, we rank search results according to their shortest path lengths from the query, q, to the target, t, in ascending order:

$$w^*(q,t) = \min_{P=\langle q,\ldots,t\rangle} \sum_{i=1}^{|P|-1} w(P_i, P_{i+1}), \tag{4}$$

3. NETWORK MODIFICATIONS

3.1 Depth-ordered retrieval

Shortest paths may sometimes hinder retrieval results in cases where they provide discursive paths that rely on numerous relations. For example, if sound-to-tag weights are trained with ground truth data obtained from user studies or extensive user activity, it would be desirable to only use these direct paths and visit no other nodes rather than second-guessing users (who, for the purpose of evaluation, we often assume are experts).

In these cases, we can form a list, L, of positive integers representing desired path depths, with an optional final element, $*$, representing shortest paths of any depth. Any targets unconnected to the query will be returned at the end of the list in random order. For example, $L = (2, *)$ will prioritize minimum-cost direct edges between the query and targets first, only using shortest paths as a last resort in the absence of direct edges. $L = (2, 3, *)$ will first return all direct edges, then shortest paths containing only three nodes, and finally all shortest paths. For the case of $L = (2, 3, ...)$, where depths are in monotonically increasing order, this algorithm performs similarly to a breadth-first search. In Section 4.3, we will discuss the relative performance of depth orderings.

3.2 Edge pruning

Shortest-path retrieval can be quite computationally expensive. In the worst case, Dijkstra's algorithm has $O(|E| + |V|\log|V|)$ time complexity. As our graph is quite well-connected (for large numbers of sounds), we can assume the complexity of performing a single query is $O(|V|^2)$, as $|E| = O(|V|^2)$ when the number of sounds greatly exceeds the number of tags. In these cases, it can be beneficial to limit search to only a node's K nearest neighbors, giving a complexity of $O(K|V| + |V|\log|V|) = O(|V|\log|V|)$. [3] In Section 4.4, we will discuss the effects of K nearest neighbor pruning, where G is converted to a directed graph with inbound and outbound edges identical to the original undirected edges and all but the K lowest-weight outbound edges are removed.

[3] Using spectral clustering to cluster sounds, as in [20], we can even improve this to $O(\log|V|)$ complexity.

3.3 Weighting edge classes

Lastly, it should be noted that the ranking of search results can be quite sensitive to variations in weighting between the different classes of edges, E_{SS}, E_{ST}, and E_{TT}, as each assumes a different probabilistic model. If one class has particularly low weights, its edges may be used more frequently than edges of other classes. In Section 4.5, we examine the effects of setting class-specific weights, γ_C:

$$\begin{aligned} w_\gamma(n_1, n_2) &= \gamma_C w(n_1, n_1) \\ \forall (n_1, n_2) \in E_C, \forall E_C &\in \{E_{SS}, E_{ST}, E_{TT}\} \end{aligned} \tag{5}$$

4. RESULTS AND DISCUSSION

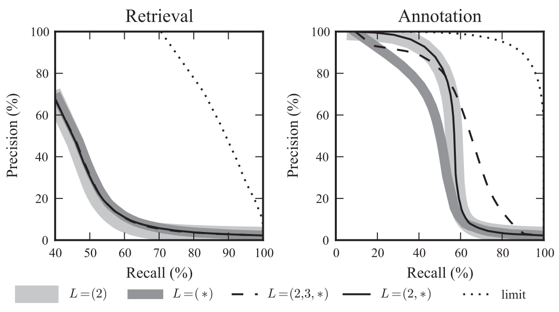
4.1 Training data

To evaluate the text-based retrieval and annotation performance of our various modifications to the shortest-path retrieval method, we use two datasets that link sounds to conceptual tags. The first dataset, *Soundwalks*, is a collection of 178 2-60s sound events manually segmented from one of four outdoor recording sessions recorded at 44.1KHz. Sound environments include light rail stops, a stadium during a football game, a skatepark, and a walkway on a college campus. To obtain tagging data, an online user study was performed where each user was asked to provide several semantic tags for 10 randomly selected sound events by freely typing terms separated by commas. Terms that could not be found in the WordNet taxonomy were ignored. With a total of 90 responses, each sound was tagged an average of 4.62 times. In [19] and [13], only the most popular 88 tags were used for evaluation, but we have used the entire set of 612 tags to more accurately study the system's performance. The second dataset, *Freesound*, was obtained from user activity on the website *Freesound.org*. 2046 sounds were randomly selected from the set of all sounds 2-60s in duration and containing at least one of any of the 50 most popular tags on the site. Each sound is associated with 3-8 tags, and each of the 377 total tags used is associated with at least 5 sounds. Note that on *Freesound*, tags are only associated to sounds in a binary manner, so all nonzero sound-to-tag weights are equal. Both datasets have been used to test the performance of the system a) against a slightly more reliable ground truth, in the case *Soundwalks*, where each sound file has been tagged by 4-5 users, and b) against a larger collection of sounds, as in *Freesound.*

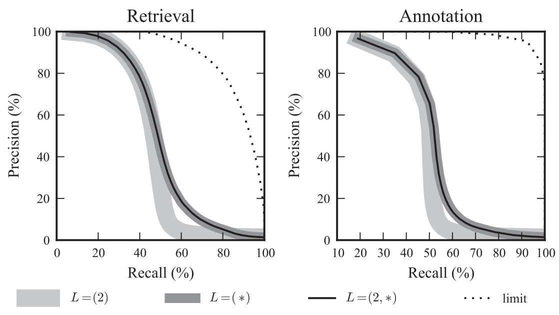
4.2 Evaluation methodology

4.2.1 Cross-validation versus incomplete tagging data

For multiclass retrieval, where classifiers are trained for each search term, evaluation procedures typically involve



(a) Soundwalks Precision/Recall



(b) Freesound Precision/Recall

	Retrieval		Annotation	
L	MAP	MAROC	MAP	MAROC
(2)	0.5505	0.7615	0.5920	0.7814
$(*)$	**0.5751**	**0.7910**	0.5230	0.7833
$(2,*)$	0.5673	0.7887	0.5964	0.7919
$(2,3,*)$	0.5678	0.7849	**0.6590**	**0.8824**

(c) Soundwalks MAP/MAROC

	Retrieval		Annotation	
L	MAP	MAROC	MAP	MAROC
(2)	0.5200	0.7504	0.5237	0.7506
$(*)$	0.5594	0.8610	**0.5628**	**0.8403**
$(2,*)$	**0.5618**	**0.8623**	0.5618	0.8399

(d) Freesound MAP/MAROC

Figure 2: Performance metrics for text-based retrieval and annotation of sounds, respectively. Data is averaged across $n = 50$ trials with half the tagging data missing. Curves are labeled according to the order of path lengths used in sorting results, where $*$ denotes all shortest paths. $Limit$ is the absolute best performance possible with the dataset.

cross-validation, where the set of sounds and their associated tags are split into several (e.g. 10) random non-overlapping subsets, the classifiers are trained with only one subset, and the remaining sounds are used as queries to test the performance of the trained classifiers. With a sufficiently large training dataset, performance results should converge to give a picture of the expected performance in a production setting.

In [19], [20], and [13], this technique was employed for shortest-path retrieval. For the cases of retrieval and annotation using sounds and tags not present in the training data (thereby testing the usefulness of both acoustic and semantic similarity), sounds and tags were split into 2 and 5 subsets, respectively, each combination thereof (one of $2 \times 5 = 10$) being used to build the network. For annotation, out-of-network sound queries were independently introduced to the network by computing their similarity to all other sounds in the network, and out-of-network tags were connected only to the in-network tags. Query performance was tested by querying each out-of-network sound against out-of-network tags. For retrieval, tag queries were connected independently to other tags, and out-of-network sounds were connected only to in-network sounds.

However, this method of cross-validation may not be entirely appropriate for shortest-path retrieval, as there is no distinct training phase (it is non-parametric). Rather than having only sounds and tags as training data, acoustic and semantic similarities *between* training instances must be considered. For this reason, we have chosen to implement a different evaluation strategy to compare techniques. In this strategy, we simulate a database where the set of sounds and tags are complete (there are no cross-validation splits), but only a random subset of the user tagging data is available. Specifically, for each association between a sound and a tag (for which there may be many for a single sound-tag pair in the *Soundwalks* dataset), we remove it with 50% probability. For annotation, every sound is used to query the entire set of tags, and for retrieval, every tag is used to query the entire set of sounds. Relevance results are then averaged over each query and over 50-100 trials with different tagging data. This simulation is perhaps more appropriate than the networks built for cross-validation, as we can examine how using shortest-path retrieval can help make up for sparse tagging data, which is oftentimes present in online tagging systems.

4.2.2 *Performance measures*

Each query returns an ordered list of nodes (tags for annotation and sounds for retrieval), sorted by path length in ascending order. An item in this list is said to be *relevant* if it is connected to the query at least once in the original user tagging data. Using this list of relevance for each item returned, we can compute mean *precision*, the percentage of items returned that are relevant as more items are returned, and mean *recall*, the percentage of all relevant items that have been returned. Plotting precision as a function of recall is a useful way of comparing different schemes. Additionally, one can compute summary statistics including *mean average precision* (MAP), the mean of precision values at the points where each relevant item is returned, and *mean area under the receiver operator characteristic* (MAROC), the integral of the curve produced by plotting the ratio of true positives versus false positives.

4.3 Depth-ordered retrieval

In Figure 2, we examine the effects of a) using shortest paths versus retrieving items based only on their tags (as most tag-based search strategies do) and b) using differ-

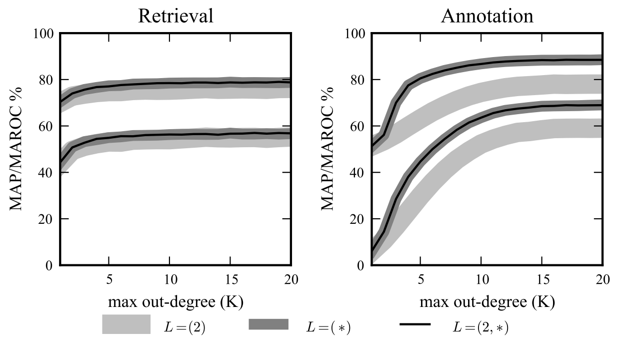


Figure 3: Effects of pruning outbound edges to nodes' K nearest neighbors using the *Soundwalks* dataset. Both MAP (lower) and MAROC (upper) values are plotted as a function of K and averaged across $n = 100$ trials with half the tagging data missing.

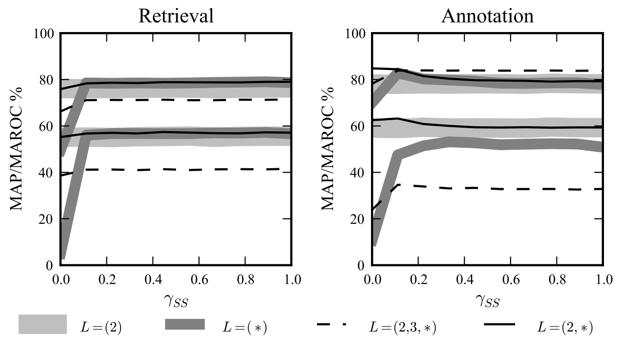


Figure 4: Effects of varying γ_{SS}, the global weight multiplier for sound-to-sound edges using the *Soundwalks* dataset. Both MAP (lower) and MAROC (upper) values are plotted as a function of γ_{SS} and averaged over $n = 50$ trials with half tagging data missing.

ent depth ordering strategies. $L = (2)$ corresponds to the case where only direct tag-to-tag links are used for retrieval (the baseline case, given no acoustic similarity information), $L = (*)$ represents the case of ranking results based on the lengths of their shortest paths, and $L = (2,*)$ and $L = (2,3,*)$ represent the cases of returning minimum 2- and 3-node paths before resorting to shortest paths. Results are shown for both the *Freesound* and *Soundwalks* datasets. $Limit$ represents the theoretical upper limit on performance imposed by the dataset, where the ground truth user tagging data itself is used to order results, analagous to $UpperBnd$ from [16]. Acoustic links were included for the *Soundwalks* dataset but not the *Freesound* dataset, in order to study the effects of using shortest-path retrieval as a drop-in method in an existing system.

From these plots, we can see that, in some cases, as in annotation on the *Soundwalks* dataset (Figure 2b), using shortest paths performs worse than the baseline case, likely because known sound-to-tag links are being circumvented in favor of paths that use acoustic similarity. However, $L = (*)$ seems to perform marginally better than $L = (2)$ for the case of retrieval. To account for this difference, we can see that prioritizing direct links, as in $L = (2,*)$, performs best. $L = (2,3,*)$ is a special case, as it produces higher MAP/MAROC, corresponding to its better performance in the last 75% of results, but it initially performs quite a bit poorer at annotation, which may be undesirable (if, say, we were to annotate with only those tags that score highest).

For the *Freesound* dataset, for which we provided no sound-to-sound links, we can see that the $L = (*)$, and optionally $L = (2,*)$, methods can assist in ordering the last half of results. This improvement is likely because, for annotation (and analagously for text-based retrieval), a sound can be annotated with additional tags from those sounds it shares a few tags with. Of course, in some use cases, this increase in performance may not be worth the extra query time. Note that $L = (2,3,*)$ would behave the same as $L = (2,*)$ in this case, as no sound-to-tag paths with an odd number of nodes exist in a network containing no sound-to-sound edges.

4.4 Pruning

To test the effects of limiting search to nodes' K nearest neighbors, we first constructed a network as described in 4.2.1 using the *Soundwalks* dataset, with sound-to-sound and sound-to-tag links, but with half the tagging data missing. For $K \in \{1, 2, ..., 20\}$, we then annotated with each sound and retrieved with each tag, testing relevance against the original tagging data. For each value of K, we averaged performance metrics over 50 trials for a total of 1000 trials per query type. As shown in Figure 3, it is only when $K < 10$ that significant losses in MAP/MAROC can be seen, suggesting that edge pruning can drastically improve query time without having significant effects on performance, as $10 \ll |E|$.

4.5 Weighting edge classes

Figure 4 demonstrates that, for the *Soundwalks* dataset, there is a clear shift in performance at $\gamma_{SS} \approx 0.2$. For $\gamma_{SS} < 0.2$, acoustic weights are used as the primary source of similarity information at the expense of known tagging data. For the case of annotation, there appears to be a slight increase in MAROC for $L = (*)$ at this point. For $L = (2,*)$, there is a slight increase in performance for $\gamma_{SS} < 0.2$, which suggests that the system performs slightly better when tagging data is used for direct links, but acoustic similarity, rather than cooccurrence of tags, is primarily relied on when no direct, 2-node links exist.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have experimented with several modifications of a shortest-path retrieval algorithm, where acoustic similarities between sounds are used in conjunction with user tagging data for the purpose of annotation and text-based retrieval. Specifically, we have demonstrated that:

1. giving priority to direct, 2-node paths before resorting to shortest path lengths can greatly improve annotation and retrieval accuracy,

2. pruning edges searched to nodes' K nearest neighbors can reduce query complexity from $O(|V|^2)$ to $O(|V|\log|V|)$ for values of K as low as 10, and

3. relative weighting between edge classes (sound-to-sound versus sound-to-tag) influences retrieval results only slightly but indicate when certain types of similarity information are best used.

While the query time of this approach will likely be slower than similar parametric classification approaches for any database where the number of desired classifiers (tags) is much less than the number of sounds, this approach can still be very useful for smaller datasets, where acoustic similarity between sounds and co-occurrence of tags can help make up for sparse tagging data, as shown in the results for the *Freesound* dataset.

In [20], a method where spectral clustering is used to create cluster nodes that reduce the number of sound-to-sound edges is discussed. In addition to actually improving query accuracy, query complexity is greatly reduced. Combined with the methods of depth-ordered search and pruning we have introduced in this paper, pre-processing sound-to-sound edges in this way could achieve time complexity as low as $O(\log|V|)$ during queries.

Additionally, [13] and [19] discuss the effects of the presence of tag-to-tag edges. While it was shown in [19] that tag-to-tag edges between in-network tags tend to hinder performance, using different edge class weights (γ_{SS} and γ_{TT}) and depth-ordered search could create a situation where tag-to-tag edges can improve results.

6. ACKNOWLEDGEMENTS

Special thanks are given to Gordon Wichern and our reviewers for many helpful comments and the contributors to and developers of *Freesound.org* for providing a wonderful source of reusable, tagged sounds. This material is based upon work supported by the National Science Foundation under grant NSF IGERT DGE-05-04647 and NSF Net Centric I/UCRC - ASU SenSIP Site Award 1035086.

7. REFERENCES

[1] Luke Barrington, Mehrdad Yazdani, Douglas Turnbull, and Gert Lanckriet. Combining feature kernels for semantic music retrieval. In *International Conference on Music Information Retrieval*, pages 614–619, 2008.

[2] P. Cano and M. Koppenberger. Automatic sound annotation. In *IEEE Workshop on Machine Learning for Signal Processing*, pages 391–400, 2004.

[3] P. Cano, M. Koppenberger, S. Le Groux, J. Ricard, P.Herrera, and N. Wack. Nearest-neighbor generic sound classification with a wordnet-based taxonomy. In *The 116th AES Convention*, Berlin, Germany, 2004.

[4] Fabio Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11:453–482, 1997.

[5] S. Essid, G. Richard, and B. David. Inferring efficient hierarchical taxonomies for music information retrieval tasks: Application to musical instruments. In *International Conference on Music Information Retrieval*, pages 324–328, 2005.

[6] M. Goto and K. Hirata. Recent studies on music information processing. *Acoustical Science and Technology*, 25(6), 24.

[7] P. Herrera-Boyer, G. Peeters, and S. Dubnov. Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(1):3–21, 2003.

[8] B. H. Huang and L. R. Rabiner. A probabilistic distance measure for hidden markov models. *AT&T Technical Journal*, 64(2):1251–1270, 1985.

[9] S. Kim, S. Narayanan, and S. Sundaram. Acoustic topic model for audio information retrieval. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 37–40, New Paltz, NY, 2009.

[10] T. Li and M. Ogihara. Detecting emotion in music. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 239–240, Baltimore, MD, 2003.

[11] O. Celma M. Sordo, C. Laurier. Annotating music collections: How content-based similarity helps to propagate labels. In *International Conference on Music Information Retrieval*, Vienna, Austra, 2007.

[12] E. Martinez, O. Celma, M. Sordo, B. de Jong, and X. Serra. Extending the folksonomies of freesound.org using content-based audio analysis. In *SMC*, Porto, Portugal, 2009.

[13] B. Mechtley, G. Wichern, H. Thornburg, and A. S. Spanias. Combining semantic, social, and acoustic similarity for retrieval of environmental sounds. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Dallas, Texas, 2010.

[14] T. Pederson, S. Patwardhan, and J. Michelizzi. Wordnet::similarity: measuring the relatedness of concepts. In *Innovative Applications of Artificial Intelligence Conference*, pages 1024–1025, Cambridge, Massachusetts, 2004. AAAI Press.

[15] M. Slaney. Semantic-audio retrieval. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages IV–1408–IV–1411, 2002.

[16] 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.

[17] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[18] B. Whitman and D. Ellis. Automatic record reviews. In *International Conference on Music Information Retrieval*, pages 470–477, 2004.

[19] G. Wichern, B. Mechtley, A. Fink, H. Thornburg, and A. Spanias. An ontological framework for retrieving environmental sounds using semantics and acoustic content. *EURASIP Journal on Audio, Speech, and Music Processing*, 2010.

[20] G. Wichern, H. Thornburg, and A. Spanias. Unifying semantic and content-based approaches for retrieval of environmental sounds. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 13–16, New Paltz, NY, 2009.

[21] G. Wichern, J. Xue, H. Thornburg, B. Mechtley, and A. Spanias. Segmentation, indexing, and retrieval for environmental and natural sounds. *IEEE Transactions on Audio, Speech and Language Processing*, 18(3):688–707, 2010.

MULTIVARIATE AUTOREGRESSIVE MIXTURE MODELS FOR MUSIC AUTO-TAGGING

Emanuele Coviello
University of California, San Diego
ecoviell@ucsd.edu

Yonatan Vaizman
University of California, San Diego
yvaizman@eng.ucsd.edu

Antoni B. Chan
City University of Hong Kong
abchan@cityu.edu.hk

Gert R.G. Lanckriet
University of California, San Diego
gert@ece.ucsd.edu

ABSTRACT

We propose the multivariate autoregressive model for content based music auto-tagging. At the song level our approach leverages the multivariate *autoregressive mixture* (*ARM*) model, a generative time-series model for audio, which assumes each feature vector in an audio fragment is a linear function of previous feature vectors. To tackle tag-model estimation, we propose an efficient hierarchical EM algorithm for *ARM*s (HEM-ARM), which summarizes the acoustic information common to the *ARM*s modeling the individual songs associated with a tag. We compare the *ARM* model with the recently proposed dynamic texture mixture (DTM) model. We hence investigate the relative merits of different modeling choices for music time-series: i) the flexibility of selecting higher memory order in *ARM*, ii) the capability of DTM to learn specific frequency basis for each particular tag and iii) the effect of the hidden layer of the DT versus the time efficiency of learning and inference with fully observable *AR* components. Finally, we experiment with a support vector machine (SVM) approach that classifies songs based on a kernel calculated on the frequency responses of the corresponding song *ARM*s. We show that the proposed approach outperforms SVMs trained on a different kernel function, based on a competing generative model.

1. INTRODUCTION

Browsing and discovery of new music can largely benefit from semantic search engines for music, which represent songs within a vocabulary of semantic tags, i.e., words or short phrases describing songs' attributes. By just typing the desired tags as in a standard text search engines (e.g., Bing or Google), users can find the music they desire.

E.C. and Y.V. contributed equally to this work. Y.V. was at ICNC, Hebrew University during this work. E.C., A.B.C. and G.R.G.L. acknowledge support from Google, Inc. E.C. and G.R.G.L. acknowledge support from Qualcomm, Inc, Yahoo!, Inc., the Hellman Fellowship Program, the Sloan Foundation, and NSF Grants CCF-0830535 and IIS-1054960. A.B.C. was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China [9041552 (CityU 110610)]. This research was supported in part by the UCSD FWGrid Project, NSF Research Infrastructure Grant Number EIA-0303622.

Historically, attempts to map songs onto a semantic vocabulary have initially relied on available metadata (e.g. artist names, genre annotation, critical reviews), or manual labeling from expert human annotators and social networks. More recently, distributed human computation games, such as TagATune [15] and HerdIt [2], have attempted to scale up manual labeling to larger collections by recruiting non-expert users through engaging or rewarding games. However, these efforts have so far covered only a small portion of the songs available in modern music collections. [1] This motivates the development of content-based auto-tagging systems, i.e., intelligent algorithms that, by analyzing and understanding the acoustic content of songs, can automatically index them with semantic tags.

1.1 Related work

A large number of content-based auto-taggers are trained on a database of songs annotated with respect to a semantic vocabulary following a common scheme. First, a time series of low-level spectral features (e.g., Mel Frequency Cepstral Coefficients (MFCCs)) is extracted from each song in the database. Then, for each tag, a representative statistical model is fine tuned to capture the most predictive patterns common to the songs annotated with that tag. Once a new song is available, the auto-tagger uses the learned tag-models to process its low-level features and produces a vector of tag-affinities. The tag-affinities are then mapped onto a semantic multinomial (SMN), which represents the song within the semantic vocabulary.

A variety of auto-taggers, based on either generative models [13, 19, 23, 24] or discriminative models [4, 9, 11, 16, 21, 26], rely on a Bag-of-Features (BoF) representation of the spectral content of songs, which ignores temporal dynamics by treating all feature vectors as independent. While augmenting the spectral features with their first and second instantaneous derivatives has represented a common choice to enrich the BoF representation with temporal information (e.g. [3, 23]), more principled solutions have been implemented in recently proposed auto-taggers. The dynamic texture mixture (DTM) treats short fragments of audio as the output of a linear dynamical system [7]. The multi-scale learning algorithm in [12] leverages several pooling functions for summarization of the features over time. The Bag-of-Systems (BoS) approach represents

[1] Pandora annotated catalog and TagATune labeled clips represent less than 5% and 0.15% of the iTunes' collection, respectively.

songs with a codebook of time-series models [10]. The multivariate autoregressive (AR) model was used in [17] in a semi-parametric approach for genre classification of short musical snippets. In [20] various methods for temporal integration, including the AR model, were examined for musical instrument classification.

1.2 Original contribution

In this paper we introduce the autoregressive mixture (ARM) model [1] for automatic music annotation and retrieval. We first model each song as an ARM, estimated from a collection of audio-fragments extracted from the song. Note that this is different from estimating single AR models from *individual* audio clips as done in [17], since each mixture component of a song-level ARM models the music content of *several* (perceptually similar) audio fragments.

In order to model tag-level distributions as ARMs, we propose a novel efficient hierarchical expectation maximization algorithm for ARMs (HEM-ARM). Starting from all the song-ARMs that are relevant for a specific tag, the proposed algorithm summarizes the common music content by clustering similar AR components together, and learning a tag-ARM model with fewer components. We compare our HEM-ARM with previous auto-taggers that used GMMs [23] and DTMs [7], to model tag-level distribution, in tandem with an efficient HEM algorithm for learning. In particular, we obtain that HEM-DTM generally performs better than HEM-ARM (e.g., the annotation F-scores are 0.264, 0.254, respectively). However, relative to HEM-DTM, our HEM-ARM has significantly lower time requirements, both for training (two orders of magnitude) and for annotation (one order of magnitude). These results are explained by the differences in the *graphical* structures of the models. The DT model has an observed layer (which models the spectral content) and a hidden layer (that encodes the temporal dynamics). As a consequence, using DTMs can learn different frequency bases that better adapt to specific tags, but requires marginalization over the hidden variables — and hence delays — at each training iteration and for inference at annotation. On the opposite, the AR is a fully observable model. Hence, training and annotation can be implemented efficiently by computing sufficient statistics for each song a single time.

In addition, once songs are modeled with ARMs, we investigate a kernel-SVM method upon these song-ARMs for semantic retrieval, similar to the work done in [3] over GMMs and in [17] over single ARs. We test several kernel functions, some of which represent each song by the quantized frequency responses (QFR) of its AR components.

The remainder of this paper is organized as follows. In section 2 we present the autoregressive (mixture) model, and in section 3 we derive the hierarchical EM algorithm for ARMs. In Section 4 we present our kernel-SVM approach. In Section 5 we report our experiments.

2. THE AUTOREGRESSIVE MIXTURE MODEL

In this section we present the autoregressive (AR) model and the autoregressive mixture (ARM) model for music time series.

2.1 The AR model

A multivariate autoregressive (AR) model is a generative time-series model for audio fragments. Given a time series of T $d-$dimensional feature vectors $x_{1:T} \in \mathbb{R}^{d \times T}$, the AR model assumes each audio feature x_t at time t is a linear combination of the previous p audio features. Specifically, the AR model is described by the equation

$$x_t = \sum_{j=1}^{p} A_j x_{t-j} + \nu_t \qquad (1)$$

where $\{A_j\}_{j=1}^{p}$ are p transition matrices of dimension $d \times d$. ν_t is a driving noise process and is i.i.d. zero-mean Gaussian distributed, i.e., $\nu_t \sim \mathcal{N}(0, Q)$, where $Q \in \mathbb{R}^{d \times d}$ is a covariance matrix. The initial condition is specified by $x_1 \sim \mathcal{N}(\mu, S)$, where $S \in \mathbb{R}^{d \times d}$ is a covariance matrix. We can express (1) in a vectorial form:

$$x_t = \widetilde{A} x_{t-p}^{t-1} + \nu_t \qquad (2)$$

where $\widetilde{A} = [A_1 \dots A_p] \in \mathbb{R}^{d \times dp}$ and $x_a^b = [x_b' \dots x_a']' \in \mathbb{R}^{dp \times 1}$. Note that, for convenience, we assume $x_t = 0$ for $t \in \{-p+1, \dots, 0\}$, and hence assume that x_1 triggered the generation of the whole time series. An AR model is hence parametrized by $\Theta = \{\mu, S, \widetilde{A}, Q\}$. The likelihood of a sequence $x_{1:T}$ is

$$p(x_{1:T}|\Theta) = \mathcal{N}(x_1|\mu, S) \prod_{t=2}^{T} \mathcal{N}(x_t| \sum_{j=1}^{p} A_j x_{t-j}, Q) \qquad (3)$$

where $\mathcal{N}(\cdot|\mu, \Sigma)$ is the pdf of a Gaussian distribution with mean μ and covariance matrix Σ.

The parameters of an AR model can be estimated from a time-series $x_{1:T}$ with various optimization criteria [17,18].

2.2 The ARM model

An ARM model treats a group of audio fragments as samples from K AR models. Specifically, for a given sequence, an assignment variable $z \sim \text{categorical}(\pi_1, \cdots \pi_K)$ selects one of the K AR models, where the i^{th} AR model is selected with probability π_i. Each mixture component is specified by the parameters $\Theta_i = \{\mu^i, S^i, \widetilde{A}^i, Q^i\}$, and the ARM model is specified by $\Theta = \{\pi_i, \Theta_i\}_{i=1}^{K}$. Whereas a single AR model suffices to describe an individual audio fragment, the ARM model is a more appropriate modeling choice for an entire song. This is motivated by the observation that a song usually shows significant structural variations within its duration, and hence multiple AR components are necessary to model the heterogeneous sections.

The likelihood of an audio fragment $x_{1:T}$ under an ARM model is

$$p(x_{1:T}|\Theta) = \sum_{i=1}^{K} \pi_i p(x_{1:T}|z = i, \Theta_i), \qquad (4)$$

where the likelihood of $x_{1:T}$ under the i^{th} AR component $p(x_{1:T}|z = i, \Theta_i)$ is given by (3).

The parameters of an ARM model can be estimated from a collection of audio-fragments using the expectation maximization (EM) algorithm [8], which is an iterative procedure that alternates between estimating the assignment variables given the current estimate of the parameters, and re-estimating the parameters based on the estimated assignment variables.

3. THE HEM ALGORITHM FOR ARM MODELS

In this paper we proposed to model tag distributions as ARM models. One way to estimate a tag-level ARM model is to run the EM algorithm directly on all the audio fragments extracted from the relevant songs. However, this approach would require excessive memory and computation time, to store all the input audio-sequences in RAM and to compute their likelihood at each iteration. In order to avoid this computational bottleneck, we propose a novel hierarchical EM algorithm for ARM models (HEM-ARM), which allows to learn ARM models using an efficient hierarchical estimation procedure. In a first stage, intermediate ARM models are estimated in parallel for each song, using the EM algorithm for ARMs on the song's audio fragments. Then, the HEM-ARM algorithm estimates the final model by summarizing the common information represented in the relevant song-ARMs. This is achieved by aggregating together all the relevant song-ARMs into a single big ARM model, and clustering similar AR models together to form the final tag-level ARM model.

At a high level, the HEM algorithm consists in maximum likelihood estimation of the ARM tag model from virtual samples distributed according to the song ARM models. However, since using the virtual samples can be approximated with a marginalization over the song ARM distribution (for the law of large numbers, see (8)), the estimation is carried out in an efficient manner that requires only knowledge of the parameters of the song models without the need of generating actual samples. The HEM algorithm was originally proposed by Vasconcelos and Lipmann [25] to reduce a GMM with a large number of mixture components to a compact GMM with fewer components, and extended to DTMs by Chan et al. [5]. The HEM algorithm has been successfully applied to the estimation of GMM tag-distribution [23] and DTM tag-distribution [7]. We now derive the HEM algorithm for ARMs.

3.1 Derivation of the HEM for ARMs

Formally, let $\Theta^{(s)} = \{\pi_i^{(s)}, \Theta_i^{(s)}\}_{i=1}^{K^{(s)}}$ be an ARM model with $K^{(s)}$ components, which pools together the ARM models of all the songs relevant for a tag. The goal of the HEM-ARM algorithm is to learn a tag-level ARM model $\Theta^{(t)} = \{\pi_j^{(t)}, \Theta_j^{(t)}\}_{j=1}^{K^{(t)}}$ with fewer components (i.e., $K^{(t)} < K^{(s)}$), that represents $\Theta^{(s)}$ well. The likelihood of the tag ARM $\Theta^{(t)}$ is given by (4).

The HEM algorithm uses a set of N *virtual* samples generated from the base model $\Theta^{(s)}$, where the $N_i = N\pi_i^{(s)}$ samples $X_i = \{x_{1:\tau}^{(i,m)}\}_{m=1}^{N_i}$ are from the i^{th} component, i.e., $x_{1:\tau}^{(i,m)} \sim \Theta_i^{(s)}$. We assume that samples within each X_i are assigned to the same component of the tag model $\Theta^{(t)}$, and we denote the entire set of virtual samples with $X = \{X_i\}_{i=1}^{K^{(s)}}$.

The log likelihood of the incomplete data under $\Theta^{(t)}$ is

$$\begin{aligned} \log p(X|\Theta^{(t)}) &= \log \prod_{i=1}^{K^{(s)}} p(X_i|\Theta^{(t)}) \\ &= \log \prod_{i=1}^{K^{(s)}} \sum_{j=1}^{K^{(t)}} \pi_j^{(t)} p(X_i|\Theta_j^{(t)}). \end{aligned} \tag{5}$$

The HEM algorithm consists of the maximum likelihood estimation of the parameters of $\Theta^{(t)}$ from (5). Since (5) involves marginalizing over the hidden assignment variables $z_i^{(s)} \in \{1, \dots, K^{(t)}\}$, its maximization can be solved with the EM algorithm. Hence, we introduce an indicator variable $\mathbf{z}_{i,j}$ for when the virtual audio sample set X_i is assigned to the j^{th} component of $\Theta^{(t)}$, i.e., when $z_i^{(s)} = j$. The complete data log-likelihood is then:

$$\begin{aligned} &\log p(X, Z|\Theta^{(t)}) = \\ &= \sum_{i=1}^{K^{(s)}} \sum_{j=1}^{K^{(t)}} \mathbf{z}_{i,j} \log \pi_j^{(t)} + \mathbf{z}_{i,j} \log p(X_i|\Theta_j^{(t)}) \end{aligned} \tag{6}$$

The $\mathcal{Q}$-function is obtained by taking the conditional expectation of (6) with respect to Z, and the dependency on the virtual samples is removed by using the law of large numbers, i.e.,

$$\log p(X_i|\Theta_j^{(t)}) = N_i \frac{1}{N_i} \sum_{m=1}^{N_i} \log p(x_{1:\tau}^{(i,m)}|\Theta_j^{(t)}) \tag{7}$$

$$\approx N_i \, \mathbb{E}_{x_{1:\tau}|\Theta_i^{(s)}} \left[\log p(x_{1:\tau}|\Theta_j^{(t)})\right]. \tag{8}$$

Note that (8) can be computed using the chain rule of the expected log-likelihood and (1) to break the expectation

$$\mathbb{E}_{x_{1:\tau}|\Theta_i^{(s)}} \left[\log p(x_{1:\tau}|\Theta_j^{(t)})\right] = \tag{9}$$

$$= \sum_{t=1}^{\tau} \mathbb{E}_{x_{1:t}|\Theta_i^{(s)}} \left[\log p(x_t|x_{1:t-1}, \Theta_j^{(t)})\right] \tag{10}$$

$$= \sum_{t=1}^{\tau} \mathbb{E}_{x_{1:t}|\Theta_i^{(s)}} \left[\log p(x_t|x_{t-p:t-1}, \Theta_j^{(t)})\right] \tag{11}$$

$$\begin{aligned} = \sum_{t=1}^{\tau} \mathbb{E}_{x_{1:t-1}|\Theta_i^{(s)}} \Big[\mathbb{E}_{x_t|x_{t-p:t-1},\Theta_i^{(s)}} \Big[\\ \log p(x_t|x_{t-p:t-1}, \Theta_j^{(t)})\Big]\Big] \end{aligned} \tag{12}$$

The inner expectation in (12) is the expected log-likelihood of a Gaussian, and its closed form solution depends on the first and second order statistics of $x_{t-p,t-1} \sim \Theta_i^{(s)}$. The outer expectation involves the computation of the expected first and second order statistics of $\Theta_i^{(s)}$, which can be carried out with the recursion presented in Algorithm 1. Note that, since the AR model $\Theta_j^{(t)}$ has no hidden variables, the computation of the expected sufficient statistics in Algorithm 1 is independent of $\Theta_j^{(t)}$, and hence needs to be executed only once for each input component $\Theta_i^{(s)}$.

Algorithm 1 Expected sufficient statistics

```
1: Input: song-level AR model $\Theta_i^{(s)} = \{\mu, S, \widetilde{A}, Q\}$, length of virtual samples $\tau$.
2: Compute expected sufficient statistics for $t = 1, \dots, \tau - 1$:
     $\tilde{E}_1^{(i)} = \mathbb{E}_{x_1 \Theta_i^{(s)}} [x_1 x_1'] = \mu\mu' + S$
     $\hat{E}_1^{(i)} = \mathbb{E}_{x_{-p+1:1} \Theta_i^{(s)}} \left[ x^1_{-p+1} {x^1_{-p+1}}' \right] = \left[ \begin{array}{cc} \tilde{E}_1^{(i)} & 0_{d \times (d-1)p} \\ 0_{(d-1)p \times d} & 0_{(d-1)p \times (d-1)p} \end{array} \right]$
     For $t = 1, \dots, \tau - 1$
         $\hat{E}_t^{(i)} = \mathbb{E}_{x_{1:t} | \Theta_i^{(s)}} \left[ x^t_{t-p+1} {x^t_{t-p+1}}' \right] = \left[ \begin{array}{cc} \widetilde{A} \hat{E}_{t-1}^{(i)} \widetilde{A}' + Q & A \hat{E}_{t-1}^{(i)} \\ \hat{E}_{t-1}^{(i)} \widetilde{A}' & \hat{E}_{t-1}^{(i)} \end{array} \right]_{(1:dp, 1:dp)}$
     Endfor
3: Compute expected sufficient statistics:
     $\hat{E}^{(i)} = \sum_{t=1}^{\tau-1} \hat{E}_t^{(i)}$    (13)
4: Output: expected sufficient statistics: $\hat{E}^{(i)}$.
```

If hidden variables are present (which is the case for the DT components of the DTM model, but not for the AR model), computing the expected sufficient statistics of a song component $\Theta_i^{(s)}$ involves marginalizing over the hidden variables of $\Theta_j^{(t)}$, and hence needs to be repeated at every iteration for each $j = 1, \dots, K^{(t)}$.

The E-step of the HEM consists of computing of the expected sufficient statistics in Algorithm 1, the assignments variables in (14) and (15), and the cumulative expected sufficient statistics in (16). The M-step maximizes the $\mathcal{Q}$-function with respect to $\Theta^{(t)}$, giving the updates in (17). The full HEM-ARM scheme is presented in Algorithm 2.

4. KERNEL-SVM APPROACH

We then used a semi-parametric approach that leverages the ARM model at the song level, and kernel support vector machine (SVM) for retrieval. In particular, we first model each song as an ARM using the EM algorithm. Then, for each tag, we learn a binary SVM classifier over the train set, based on a notion of similarity between ARM models defined in terms on their proximity in parameter space. Finally, following [3], we use the SVMs' decision values as the relevance of a song for a tag, and use it for retrieval of test songs based on one-tag queries.

Since the AR parameters lie on a non-linear manifold, naïvely treating them as Euclidean vectors would not necessary produce a correct similarity score. Hence, in the remainder of this section, we present several kernel functions based on more appropriate similarity scores between autoregressive (mixture) models. In previous work, Meng and Shawe-Taylor [17] specialize the Probability Product Kernel [14] to the AR case, which depends non-linearly on the AR parameters, and is define as:

$$\mathcal{K}_{\text{AR}}(\Theta_a, \Theta_b) = \int_{x_{1:p}} (p(x_{1:p}|\Theta_a) p(x_{1:p}|\Theta_b))^\rho, \tag{18}$$

where $\rho = 0.5$ corresponds to the Battaccharyya affinity. Since a song-ARM is associated with several AR components, for retrieval we collect a decision value for each AR component, and then rank the songs according to the average of the corresponding decision values (PPK-AR). Note that we compute PPK between individual AR components of the song-ARMs. This is different from [17], which uses *single* ARs on individual audio snippets.

Algorithm 2 HEM algorithm for ARM

```
1: Input: combined song-level ARM $\{\pi_i^{(s)}, \Theta_i^{(s)}\}_{i=1}^{K^{(s)}}$, number of virtual samples $N$.
2: Compute cumulative expected sufficient statistics $\hat{E}^{(i)}$ for each $\Theta_i^{(s)}$ using Algorithm 1
3: Initialize tag-level ARM, $\{\pi_j^{(t)}, \Theta_j^{(t)}\}_{j=1}^{K^{(t)}}$.
4: repeat
5:     {E-step}
6:     Compute expected log-likelihood for each $\Theta_i^{(s)}$ and $\Theta_j^{(t)}$:
         $\ell_{i|j} = \mathbb{E}_{x_{1:\tau}|\Theta_i^{(s)}} [\log p(x_{1:\tau}|\Theta_j^{(t)})]$
         $= -\frac{d\tau}{2} \log 2\pi - \frac{1}{2} \log |S_j^{(t)}|$
         $-\frac{1}{2} \text{trace} {S_j^{(t)}}^{-1} [S_i^{(s)} + (\mu_j^{(t)} - \mu_i^{(s)})' (\mu_j^{(t)} - \mu_i^{(s)})]$
         $-\frac{\tau-1}{2} \text{trace} {Q_j^{(t)}}^{-1} Q_i^{(s)} - \frac{\tau-1}{2} \log |Q_j^{(t)}|$
         $-\frac{1}{2} \text{trace} [{Q_j^{(t)}}^{-1} (\widetilde{A}_j^{(t)} - \widetilde{A}_i^{(s)}) \hat{E}^{(i)} (\widetilde{A}_j^{(t)} - \widetilde{A}_i^{(s)})']$
7:     Compute assignment probability and weighting:
         $\hat{\mathbf{z}}_{i,j} = \frac{\pi_j^{(t)} \exp\left(N_i \ell_{i|j}\right)}{\sum_{j'=1}^{K^{(t)}} \pi_{j'}^{(t)} \exp\left(N_i \ell_{i|j'}\right)}$    (14)
         $\hat{\mathbf{w}}_{i,j} = \hat{\mathbf{z}}_{i,j} N_i = \hat{\mathbf{z}}_{i,j} \pi_i^{(s)} N$    (15)
8:     Computed aggregated expectations for each $\hat{\Theta}_j^{(t)}$:
         $\hat{N}_j = \sum_i \hat{\mathbf{z}}_{i,j}$,    $\hat{M}_j = \sum_i \hat{\mathbf{w}}_{i,j}$,
         $\hat{S}_j = \sum_i \hat{\mathbf{w}}_{i,j} [S_i^{(s)} + \mu_i^{(s)} (\mu_i^{(s)})']$    $\hat{m}_j = \sum_i \hat{\mathbf{w}}_{i,j} \mu_i^{(s)}$
         $\hat{V}_j = \sum_i \hat{\mathbf{w}}_{i,j} A_i^{(s)} \hat{E}^{(i)} (\widetilde{A}_i^{(s)})'$    $\hat{P}_j = \sum_i \hat{\mathbf{w}}_{i,j} \hat{E}^{(i)}$    (16)
         $\hat{R}_j = \sum_i \hat{\mathbf{w}}_{i,j} \hat{E}^{(i)} (\widetilde{A}_i^{(s)})'$    $\hat{Q}_j = \sum_i \hat{\mathbf{w}}_{i,j} Q_i^{(s)}$
9:     {M-step}
10:    Recompute parameters for each component $\hat{\Theta}_j^{(t)}$:
         $\widetilde{A}_j^* = \hat{R}_j' \hat{P}_j^{-1}$    $Q_j^* = \frac{1}{(\tau-1)\hat{M}_j} (\hat{V}_i - A_j^* \hat{R}_j + \hat{Q}_j)$,
         $\mu_j^* = \frac{1}{\hat{M}_j} \hat{m}_j$,    $S_j^* = \frac{1}{\hat{M}_j} \hat{S}_j - \mu_j^* (\mu_j^*)'$,    (17)
         $\pi_j^* = \frac{\sum_{i=1}^{K^{(s)}} \hat{\mathbf{z}}_{i,j}}{K^{(s)}}$.
11: until convergence
12: Output: tag-level ARM $\{\pi_j^{(t)}, \Theta_j^{(t)}\}_{j=1}^{K^{(t)}}$.
```

In addition, we experiment SVM classification in tandem with a probability product kernel between auotoregressive *mixture* models (PPK-ARM). Following an approximation by Jebara et al. [14], the PPK-ARM can be computed from the PPK between individual components as

$$\mathcal{K}_{\text{ARM}}(\Theta^{(1)}, \Theta^{(2)}) = \sum_{a=1}^{K_s} \sum_{b=1}^{K_s} (\pi_a^{(1)} \pi_b^{(2)})^\rho \mathcal{K}_{\text{AR}}(\Theta_a^{(1)}, \Theta_b^{(2)}). \tag{19}$$

We finally propose a novel descriptor of AR models based on their frequency responses, and compute a kernel between these descriptors. Since an AR is a linear time invariant (LTI) system, its dynamics can be characterized by a transfer function defined as:

$$H(s) = (I_d - \sum_{j=1}^{p} A_j s^{-j})^{-1} \in \mathbb{C}^{d\times d} \qquad (20)$$

where $s \in \mathbb{C}$ is a complex number and I_d is the d dimensional identity matrix. The transfer function describes the cross influences of each pair of components of the audio feature vectors. In particular, we sample the transfer function at 200 equally spaced points on the unit circle, and then sum the the absolute values of these matrices over 30 linearly spaced frequency bins, to get a representation of the system's frequency response. By concatenating the AR's μ parameter and the log values of these 30 frequency response matrices, we get a descriptor $\Delta \in \mathbb{R}^{(d+30d^2)\times 1}$, which we call quantized frequency response (QFR). Finally, we use a SVM over QFRs based on cosine-similarity (CS) kernel and radial basis function (RBF) kernel. [2]

5. EXPERIMENTS

5.1 Data

We performed automatic music annotation on the CAL500 dataset (details in [23] and references therein), which is a collection of 502 popular Western songs by as many different artists, and provides binary annotations with respect to a vocabulary of semantic tags. In our experiments we consider the 97 tags associated to at least 30 songs in CAL500 (11 genre, 14 instrumentation, 25 acoustic quality, 6 vocal characteristics, 35 mood and 6 usage tags).

The acoustic content of a song (resampled at $22,050Hz$) is represented by computing a time-series of 34-bin Mel-frequency spectral (MFS) features, extracted over half overlapping windows of 92 msec of audio signal, i.e., every $\sim$ 46 msec. Following the insight in recent work of Hamel et al. [12], MFS features where further projected on the first $d = 20$ principal components, which we estimated over the MFSs collected from the $10,870$ songs in the CAL10K dataset [22].

Song level ARMs were learned with $K = 4$ components and memory of $p = 5$ steps, from a dense sampling of audio fragments of length $T = 125$ (i.e., approximately $6s$), extracted with 80% overlap.

5.2 Results with HEM-ARM

For each tag, all the relevant song ARMs were pooled together to form a big ARM, and a tag-level ARM with $K^{(t)} = 8$ components was learned with the HEM-ARM algorithm (with $N = 1000$ virtual samples of length $\tau = 10$). To reduce the effects of low likelihood in high dimension, for annotation we smooth the likelihood (3) by $T \cdot d \cdot p$. We compare our HEM-ARM with hierarchically trained Gaussian mixture models (HEM-GMM) [23] and dynamic texture mixture models (HEM-DTM) [7].

On the test set, a novel test song is annotated with the 10 most likely tags, corresponding to the peaks in its semantic multinomial. Retrieval given a one tag query involves rank ordering all songs with respect to the corresponding entry in their semantic multinomials. Annotation is measured with average per-tag precision (P), recall (R), and f-score (F). Retrieval is measured by per-tag area under the ROC (AROC), mean average precision (MAP), and precision at the first 10 retrieved objects (P@10). Refer to [23] for a detailed definition of the metrics. All reported metrics are the result of 5 fold-cross validation.

	annotation			**retrieval**			**time**	
	P	R	F	AROC	MAP	P@10	train	test
HEM-ARM	0.468	0.203	0.254	0.696	0.421	0.412	$198m$	$41m$
HEM-DTM	0.446	0.217	0.264	0.708	0.446	0.460	$424h$	$482m$
HEM-GMM	0.474	0.205	0.213	0.686	0.417	0.425	$41m$	$38m$

Table 1. Annotation and retrieval on CAL500, for HEM-ARM, HEM-DTM and HEM-GMM.

In Table 1 we report annotation and retrieval results for HEM-ARM, HEM-DTM and HEM-GMM. In addition, we register the total time for the training stage, which consist in the estimation of the 97 tag models over the 5 folds (and also includes the estimation of the 502 song-level models), as well as for the test stage, i.e., the automatic-annotation of the 502 songs with the 97 tags.

From Table 1 we note that the advantages of the proposed HEM-ARM relative to HEM-DTM are in terms of *computation efficiency*. While HEM-DTM performs better than HEM-ARM on each metric (except on annotation precision where HEM-ARM is better), HEM-ARM has significantly lower time requirements. Specifically, the training time for our HEM-ARM is two orders of magnitude lower than that for HEM-DTM. Similarly, our auto-tagger requires approximately 42 minutes for the test-stage, while the auto-tagger based on DTMs requires 482 minutes to accomplish the same task. These results are explained by comparing the *graphical structures* of the AR and DT models. While the AR is a fully observable model, the DT consists of an observed layer, which model the spectral content, and a hidden layer that encodes the temporal dynamics. Hence, DTMs have the advantage of learning different frequency basis to best represent specific tags [7]. However, the computation of the (expected) sufficient statistics with respect to each DT component requires marginalization of the hidden variables (see [5]). Hence, during training, it needs to be executed at each iteration of the learning algorithms for each input datum (i.e., audio-fragments for EM, and DTs for HEM) and for each individual component of the learned model; during annotation, it needs to be repeated for each audio-fragment and each DT component of the tag models. On the opposite, the corresponding statistics for ARMs involve no marginalization of hidden variables. Hence, during training, they need to be computed only a single time for each input datum (i.e., audio-fragments for EM, and ARs for HEM). In addition, during annotation, the sufficient statistics can be collected a single time for each song.

Finally, HEM-ARM performs favorably relative to HEM-GMM (which does not model temporal dynamics), while still requiring limited computation times. Since learning and inference are performed efficiently, HEM-ARM can leverage higher order memories to model temporal dynam-

[2] The CS kernel is defined as $\mathcal{K}(a, b) = a'b/\sqrt{||a||_2||b||_2}$. The RBF kernel is defines as $\mathcal{K}(a, b) = \exp\{-||a - b||_2^2/\sigma\}$. We set the bandwidth σ of the RBF kernel to the descriptor dimension $dim(\Delta)$.

p	1	2	3	4	5	6	7	8	9
F-score	0.234	0.247	0.252	0.255	0.254	0.245	0.244	0.242	0.237
AROC	0.650	0.672	0.686	0.693	0.696	0.695	0.694	0.695	0.694

Table 2. Annotation (F-score) and retrieval (AROC) performance of HEM-ARM, for different memories $p \in [1{:}9]$.

kernel		AROC	MAP	P@10	train	test
ARM based	PPK-ARM	0.717	0.448	0.459	$233m$	$194m$
	PPK-AR	0.727	0.463	0.484	$287m$	$194m$
	QFR-CS	0.717	0.461	0.479	$125m$	$65m$
	QFR-RBF	0.723	0.469	0.488	$137m$	$74m$
GMM-PPK [3]		0.696	0.436	0.454		
MFCC-PPK-AR [17]		0.706	0.447	0.463		

Table 3. Retrieval for the kernel-SVM approach. Including train and test times

ics, without incurring in large delays. In particular, in Table 2 we plot annotation (F-score) and retrieval (AROC) performance as a a function of the memory p of the AR models. Performance are fairly stable for $p = 4, 5, 6$. Shorter memories (e.g., $p = 1, 2, 3$) do not suffices to capture the interesting dynamics, while too large values deteriorate annotation performance.

5.3 Results with kernel-SVM.

We implemented the kernel-SVM approach as described in Section 4. In particular, we learned song-ARMs with $K = 4$ components and memory $p = 5$, estimated from the $d = 20$ dimensional PCA-MFS features. We then computed the QFR-CS and QFR-RBF kernels based on the QFR descriptors, the PPK kernel between ARM (PPK-ARM), and the PPK kernel between individual AR components (PPK-AR). For comparison, we also considered PPK similarity between song-GMMs estimated on MFCC features [3] (GMM-PPK) and PPK similarity between single AR models estimated on the MFCC features of entire songs as proposed in [17] (MFCC-PPK-AR). We used the LibSVM software package [6] for the SVM, with all parameters selected using validation on the training set.

Retrieval scores are reported in Table 3, and are result of 5-fold cross validation. We note that these results are generally superior to those in Table 1, since the SVM is a discriminative algorithm and hence tends to be more robust on strongly labeled datasets such as CAL500. In particular, the best performance was registered with the QFR-RBF and PPK-AR systems (score differences between them are not statistically significant). In addition, PPK similarity on ARMs (PPK-ARM) proves less performant, suggesting that the approximation in (19) may be not enough accurate for our task. Finally, PPK similarity on GMM performs the worst, since it does not leverage termporal dynamics, and MFCC-AR-PPK, which doesn't leverage mixtures, is also significantly behind.

6. DISCUSSION

In this paper we have proposed the ARM model for music auto-tagging. We have derived a hierarchical EM algorithm for efficiently learning tag ARMs. We have showed that our HEM-ARM can estimate tag models significantly more efficiently than HEM-DTM, at the price of a small reduction in performance. We have also successfully tested a kernel-SVM approach based on several similarity functions based on the ARM model.

7. REFERENCES

[1] A. Agarwal and B. Triggs. Tracking articulated motion using a mixture of autoregressive models. In *ECCV - Lecture notes in computer science*, pages 54–65, 2004.

[2] L. Barrington, D. O'Malley, D. Turnbull, and G. Lanckriet. User-centered design of a social game to tag music. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, pages 7–10. ACM, 2009.

[3] L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet. Combining feature kernels for semantic music retrieval. *Proc. ISMIR 2008*, pages 723–728, 2008.

[4] 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.

[5] A.B. Chan, E. Coviello, and G. Lanckriet. Clustering dynamic textures with the hierarchical EM algorithm. In *Proc. IEEE CVPR*, 2010.

[6] C.C. Chang and C.J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[7] 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.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[9] 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.

[10] K. Ellis, E. Coviello, and G. Lanckriet. Semantic annotation and retrieval of music using a bag of systems representation. In *ISMIR*, 2011.

[11] A. Flexer, F. Gouyon, S. Dixon, and G. Widmer. Probabilistic combination of features for music classification. In *Proc. ISMIR*, pages 111–114, 2006.

[12] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. ISMIR, 2011.

[13] M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In *Proc. ISMIR*, pages 369–374, 2009.

[14] T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *The Journal of Machine Learning Research*, 5:819–844, 2004.

[15] 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, 2009.

[16] M.I. Mandel and D.P.W. Ellis. Multiple-instance learning for music information retrieval. In *Proc. ISMIR*, pages 577–582, 2008.

[17] A. Meng and J. Shawe-Taylor. An investigation of feature models for music genre classification using the support vector classifier. In *Proc. ISMIR*, pages 604–609, 2005.

[18] A. Neumaier and T. Schneider. Estimation of parameters and eigenmodes of multivariate autoregressive models. *ACM Transactions on Mathematical Software (TOMS)*, 27(1):27–57, 2001.

[19] J. Reed and C.H. Lee. A study on music genre classification based on universal acoustic models. In *Proc. ISMIR*, pages 89–94, 2006.

[20] C. Joder S. Essid and G. Richard. Temporal integration for audio classification with application to musical instrument classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(1):174–186, 2009.

[21] M. Slaney, K. Weinberger, and W. White. Learning a metric for music similarity. In *Proc. ISMIR*, pages 313–318, 2008.

[22] Derek Tingle, Youngmoo E. Kim, and Douglas Turnbull. Exploring automatic music annotation with "acoustically-objectiv" tags. In *Proc. MIR*, New York, NY, USA, 2010.

[23] 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.

[24] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.

[25] N. Vasconcelos and A. Lippman. Learning mixture hierarchies. In *Advances in Neural Information Processing Systems*, pages 606–612, 1998.

[26] B. Whitman and D. Ellis. Automatic record reviews. In *Proc. ISMIR*, 2004.

BUILDING MUSICALLY-RELEVANT AUDIO FEATURES THROUGH MULTIPLE TIMESCALE REPRESENTATIONS

Philippe Hamel, Yoshua Bengio
DIRO, Université de Montréal
Montréal, Québec, Canada
{hamelphi,bengioy}@iro.umontreal.ca

Douglas Eck
Google Inc.
Mountain View, CA, USA
deck@google.com

ABSTRACT

Low-level aspects of music audio such as timbre, loudness and pitch, can be relatively well modelled by features extracted from short-time windows. Higher-level aspects such as melody, harmony, phrasing and rhythm, on the other hand, are salient only at larger timescales and require a better representation of time dynamics. For various music information retrieval tasks, one would benefit from modelling both low and high level aspects in a unified feature extraction framework. By combining adaptive features computed at different timescales, short-timescale events are put in context by detecting longer timescale features. In this paper, we describe a method to obtain such multi-scale features and evaluate its effectiveness for automatic tag annotation.

1. INTRODUCTION

Frame-level representations of music audio are omnipresent in the music information retrieval (MIR) field. Spectrograms, mel-frequency cepstral coefficients (MFCC), chromagrams and stabilized auditory images (SAI) are just a few examples of features that are typically computed over short frames. It has been shown that using frame-level features aggregated over time windows on the scale of a few seconds yields better results on various MIR tasks [2] than applying learning algorithms directly on frame-level features. However, the aggregation of frame-level features, also known as the bag-of-frames approach, does not model the temporal structure of the audio beyond the timescale of the frames. A simple method to get some information about short-time dynamics is to use the derivatives of the frame-level features. However, this method does not yield a representation that can model much longer temporal structure. Some alternative techniques to the bag-of-frames approach inspired by speech processing rely on the modelization of the temporal structure with models such as HMMs [12]. A representation that could jointly model the short-term spectral structure and long-term temporal structure of music audio would certainly improve MIR systems.

In this paper, we take a step to improve the bag-of-frames approach by combining a set of features computed over different timescales. The idea is that longer timescale features, by modelling temporal structure, will give some context to the shorter timescale features which model spectral structure. The combination of multiple timescales could yield a general representation of the music audio that would be useful to solve various MIR tasks relying on audio features. In particular, we will show that a simple classifier trained over a multi-scale spectral representation of music audio obtains state-of-the-art performance on the task of automatic tag annotation. The multi-timescale representation that we introduce in this paper has the advantage of being a general purpose scalable method that requires no prior knowledge of the spectral or temporal structure of music audio.

The paper is divided as follows. First, in Section 2, we describe the current state of the research on multi-scale representations. Then, in Section 3, we describe our experimental setup. In Section 4 we discuss our results. Finally, we conclude in Section 5.

2. MULTI-SCALE REPRESENTATIONS

Using representations at multiple scales allows much flexibility to model the structure of the data. Multi-scale representations offer a natural way to jointly model local and global aspects, without having prior knowledge about the local and global structures.

The idea of considering multiple scales is not new. It has been applied widely in the machine vision field. For example, pyramid representations [3] and convolutional networks [8] are just a few examples of multi-scale representations.

Recently, the MIR community as shown interest in taking advantage of multi-scale representations. Here are a few examples of recent work that has been done on multi-scale representation of music audio. Multi-scale spectro-temporal features inspired by the auditory cortex have been proposed in [11]. These features are used to discriminate speech from non-speech audio in a small dataset. In [10], structural change of harmonic, rhythmic and timbral features are computed at different timescales. This representation is used to build meaningful visualizations, although it has not been applied to music audio classifica-

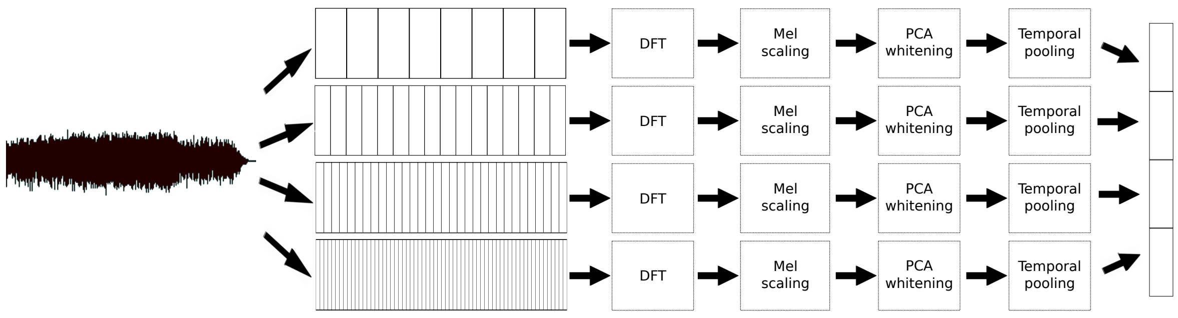


Figure 1: PMSCs are computed in parallel at different timescales.

tion. In [5], boosting is applied on features at different timescales to optimize music classification. Although the validity of this method is demonstrated, it does not obtain state-of-the-art results on the CAL500 dataset. Learning features jointly at different timescales obtains state-of-the-art performance for automatic tag annotation [6]. However this model still depends on a bag of short timescale frames to build the long timescale representation, limiting the potential to model temporal dynamics. Deep convolutional networks have been applied to genre recognition in [9]. The authors show that classification performance for genre recognition and artist identification can be improved by using an unsupervised deep convolutional representation instead of raw MFCC features. Unfortunately, the results presented in this work are not comparable to other work in the field. In [1], scattering representations of MFCCs have been shown to improve music genre classification. The performance reported are comparable to other results reported on the same dataset. A bag-of-system approach have been proposed in [4] to combine models at various time resolutions.

3. EXPERIMENTAL SETUP

We used the TagATune dataset [7] in our experiments. TagATune is the largest dataset for music annotation available for research that provides audio files. It contains over 20,000 30-second audio clips sampled at 22050 Hz, and 160 tag categories. Our train, valid and test datasets contained 14660, 1629 and 6499 clips respectively.

We used the area under the ROC curve (AUC) averaged over tags (AUC-tag) as our main performance measure. We also use the AUC averaged over clips (AUC-clip) and precision at k for comparison with other models. For more details on these performance measures, see [6].

3.1 Multi-scale Principal Mel-Spectrum Components

In our experiments, we used Principal Mel-Spectrum Components (PMSCs) [6] as base features. PMSCs are general purpose spectral features for audio. They are obtained by computing the principal components of the mel-spectrum. PMSCs have shown great potential for the task of music tag annotation.

Moreover, it is quite simple to compute PMSCs at different timescales. The time length of the frame used to compute the discrete Fourier transform (DFT) determines the timescale of the features. To obtain multi-timescale features, we simply need to compute a set of PMSCs over frames of different lengths (Figure 1). The smallest DFT window we used was 1024 samples (46.4 ms). The size of the timescales grew exponentially in powers of 2 (1024, 2048, 4096, etc.).

We keep the same number of mel coefficients for all timescales. Thus, longer frames are more compressed by the mel-scaling, since the dimensionality of the output from the DFT is proportional to the frame's length. However, mel-scaling is more important for high frequency bins, while low-frequency bins are barely compressed by the mel-scaling. Fortunately, these high frequencies are already represented in shorter timescales where they are less compressed. In our experiments, we used 200 mel energy bands.

In our experiments, we found that using the log amplitude of the mel-spectrum yields better performance than using the amplitude.

PCA whitening is computed and applied independently on each timescale. In order to circumvent memory problems when computing the PCA, we limit the number of frame examples by randomly sub-sampling frames in the training set. We typically used around 75 000 frames to compute the PCA. It is also worth noting that we preserve all the principal components since we don't use PCA for dimensionality reduction, but rather to obtain a feature space with an approximate diagonal covariance matrix. The PCA whitening step decorrelates the features, which allows a more efficient temporal aggregation.

The principal components obtained for different timescales are shown in Figure 2. For each timescale, the first few principal components (those that account for the most variance in the data) tend to model global spectral shape. Subsequent components then model harmonic structure in the lower part of the mel-spectrum, and as we go up in the coefficients (and lower in the accounted variance), the components model structure in higher frequencies. It is interesting to notice the periodic structure in the components which shows how the harmonics are captured by the components. Also, if we compare components between timescales, we can observe that components tend to model a larger part of the mel-spectrum and exhibit more structure in the lower frequencies as we go higher in the frame size.

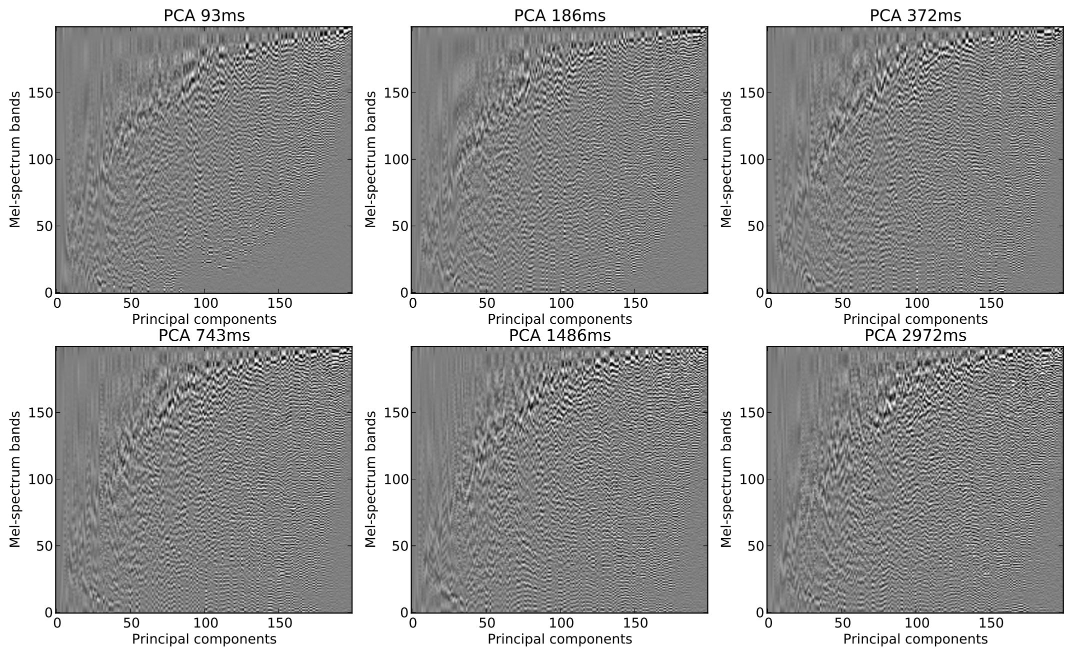


Figure 2: PCA Whitening matrices for different timescales. The first few principal components tend to model global spectral shape. Subsequent components then model harmonic structure in the lower part of the mel-spectrum, and as we go up in the coefficients, the components model structure in higher frequencies.

The next step consists of summarizing the features over a given time window by computing meaningful statistics. We refer to this step as temporal pooling. Following results from [6], we combined four pooling functions: mean, variance, maximum and minimum. These statistics are applied independently to each principal component through time and concatenated into a single feature vector for a given time window. In consequence, for each timescale we obtain a feature vector having four times the dimension of a single frame. Again, following results from [6], we fixed the pooling window at approximately 3 seconds for all experiments. Although, depending on how the frames were overlapped, this window length might vary for different timescales (see Section 4). The choice of the window length can be justified by the fact that 3 seconds would be enough for a human listener to label audio examples, but longer windows would give us less meaningful statistics for shorter timescales.

By concatenating the pooled features from each timescale, we obtain multi-timescale PMSCs (Figure 1).

3.2 Multi-Layer Perceptron

The classifier we used is similar as the pooled feature classifier (PFC) model presented in [6]. However, in our case, the input pooled feature vector will tend to be larger, since it is obtained by concatenating many timescales.

We used a one-hidden layer artificial neural network, also known as multi-layer perceptron (MLP), as the classifier for all experiments. We kept the size of the network constant at 1000 hidden units for all experiments. The number of parameters (weights) in the system varies depending on the dimensionality of the input.

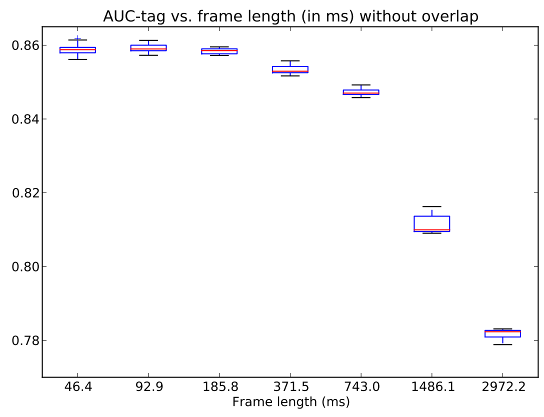
The input to the MLP is a multi-timescale PMSC representation a window of approximately 3 seconds of audio. In order to obtain tags for a full song in the test and validation phases, we simply average the MLP outputs over all windows from that song.

The MLP is well suited for multi-label classification like the music annotation task. The hidden layer acts as a latent representation that can model correlation between inputs as well as shared statistical structure between the conditional distributions associated with different targets (tags). This gives the MLP an advantage over other models such as the multi-class SVM, for which one would have to train a separate model for each tag. Also, the MLP scales sub-linearly in the number of examples, so it scales well to large datasets.

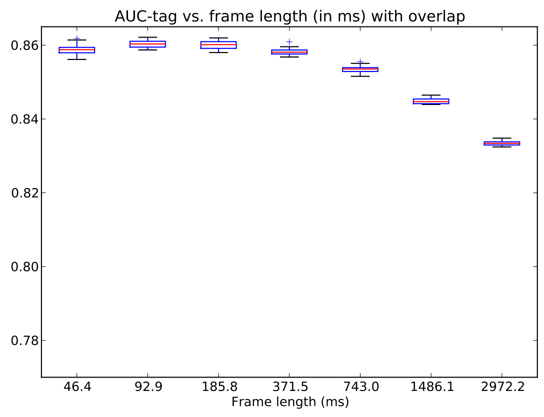
4. RESULTS

In our experiments, we evaluated the performance of different timescales individually, and their combination for the task of automatic tag annotation.

In our first experiment, for a given timescale, we did not overlap frames. In consequence, longer timescales have fewer frame examples. In the extreme case, the longest timescale is the size of the pooling window, meaning that the max, mean and min are all equal, and variance is zero. Obviously, this is not ideal. As we can see in Figure 3a, longer timescale perform worse than short timescales. However, we still see a significant advantage to using a com-

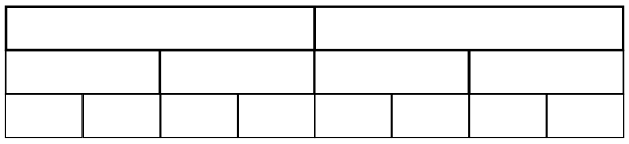


(a)

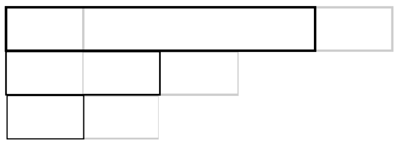


(b)

Figure 3: AUC-tag for single timescale features without overlap (a) and with overlap (b). Shorter timescales tend to perform better than longer timescales, and performance generally improve when using overlapped frames.



(a) no overlap



(b) overlap

Figure 4: Illustration of frames without overlap (a) and with overlap (b).

	Multi PMSCs	PMSCs	PMSCs + MTSL	MUSLSE
AUC-Tag	**0.870**	0.858	0.868	-
AUC-Clip	**0.949**	0.944	0.947	-
Precision at 3	**0.481**	0.467	0.470	0.476
Precision at 6	**0.339**	0.330	0.333	0.334
Precision at 9	**0.263**	0.257	0.260	0.259
Precision at 12	**0.216**	0.210	0.214	0.212
Precision at 15	**0.184**	0.179	0.182	0.181

Table 1: Performance of different automatic annotation models on the TagATune dataset

bination of timescales. In Figure 5a, we show the performance of multi-timescale features. We combined timescales incrementally, starting from the shortest one to the longest one. For example, the representation with two timescales combines 46.4ms and 92.9ms frames, the one with three timescales combines 46.4ms, 92.9ms and 185.8ms frames, etc.

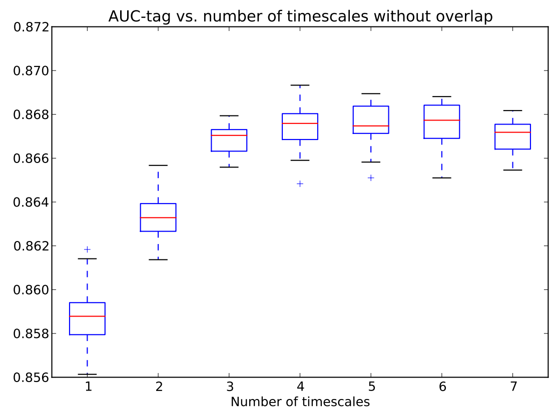
In order to obtain more examples for higher timescales, and yield more meaningful statistics for the temporal pooling, we considered using more overlapping between windows. In our second experiment, we used the same frame step for all timescales, corresponding to the smallest frame length, in this case, 46ms (Figure 4). We include all frames that start within the pooling window in the temporal pooling. This means that the longest timescale frames will overflow beyond the pooling window length up to almost twice the window length. Even though this method will give us the same number of frames to aggregate for each timescale, the longer timescales will still have much more redundancy than shorter timescales. Longer timescales perform significantly better with more overlap than without overlap, as we can see by comparing Figure 3a and 3b. The overlap also gives a boost of performance when combining timescales (Figure 5b).

In Table 1, we show the test performance of the model that obtained the best AUC-tag on the validation set. We compare with two other state-of-the-art models: Multi-timescale learning model (MTSL) [6] and Music Understanding by Semantic Large Scale Embedding MUSLSE [13]. The multi-timescale PMSCs trained with the MLP obtains the best performance on all measures. Moreover, this model is a lot faster to train than the MTSL. For the TagATune dataset, the training time would typically be a few hours for the MLP compared to a few days for the MTSL.

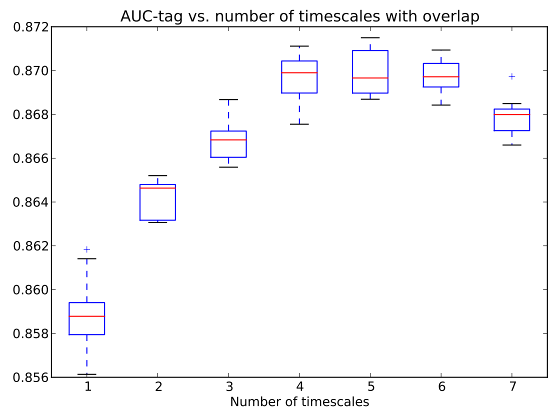
5. CONCLUSION

Multi-timescale PMSCs are general purpose features that aim at jointly modelling aspects salient at multiple timescales. We showed that, for the task of automatic tag annotation, using multi-timescale features gives an important boost in performance compared to using features computed over a single timescale. Moreover, with a simple classifier, we obtain state-of-the-art performance on the TagATune dataset.

Multi-timescale PMSCs could potentially improve the performance of more complex learning models such as MTSL or MUSLSE. They could most likely be useful for other music information retrieval tasks such as genre recognition, instrument recognition or music similarity as well.



(a)



(b)

Figure 5: AUC-tag in function of number of timescales used without overlap (a) and with overlap (b). The combination of timescales always include the shorter timescales. For example, the representation with 2 timescales combines 46.4ms and 92.9ms frames, the one with 3 timescales combines 46.4ms, 92.9ms and 185.8ms frames, etc.

Although the timescales used in these experiments are not long enough to model many aspects of the temporal structure of music, the combination of multiple timescales of analysis allows to model some mid-level temporal dynamics that are useful for music classification. It is also a improvement on the typical bag-of-frames approach. Even though we are still using frame level features, the concatenation of longer timescale representations puts short-time features in context.

In future work, it would be interesting to optimize the pooling window lengths independently for each timescale. This would allow longer timescale features to be aggregated over less redundant information and provide more relevant and stable statistics. It would also allow us to compute PMSCs over even larger timescales.

6. REFERENCES

[1] J. Andén and S. Mallat. Multiscale scattering for audio classification. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR'11)*, 2011.

[2] J. Bergstra. Algorithms for Classifying Recorded Music by Genre. Masters thesis, Université de Montréal, 2006.

[3] P. Burt and T. Adelson. The laplacian pyramid as a compact image code. *IEEE Trans. Communications*, 9:4(532–540), 1983.

[4] K. Ellis, E. Coviello, and G.R.G. Lanckriet. Semantic annotation and retrieval of music using a bag of systems representation. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR'11)*, 2011.

[5] R. Foucard, S. Essid, Lagrange M., and Richard G. Multi-scale temporal fusion by boosting for music classification. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR'11)*, 2011.

[6] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR'11)*, 2011.

[7] E. Law, K. West, M. Mandel, M. Bay, and J. S. Downie. Evaluation of algorithms using games: the case of music tagging. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR'09)*, 2009.

[8] 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.

[9] H. Lee, Y. Largman, P. Pham, and A. Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems (NIPS) 22.*, 2009.

[10] M. Mauch and M. Levy. Structural change on multiple time scales as a correlate of musical complexity. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR'11)*, 2011.

[11] N. Mesgarani, M. Slaney, and S. Shamma. Content-based audio classification based on multiscale spectro-temporal features. *IEEE Transaction on Speech and Audio Processing*, 2006.

[12] J. Reed and C.-H. Lee. On the importance of modeling temporal information in music tag annotation.

In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2009*, pages 1873–1876, 2009.

[13] J. Weston, S. Bengio, and P. Hamel. Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval. *Journal of New Music Research*, 2011.

A COMPARISON OF SOUND SEGREGATION TECHNIQUES FOR PREDOMINANT INSTRUMENT RECOGNITION IN MUSICAL AUDIO SIGNALS

Juan J. Bosch, Jordi Janer, Ferdinand Fuhrmann and Perfecto Herrera
Universitat Pompeu Fabra, Music Technology Group, Roc Boronat 138, Barcelona
juanjo.bosch@gmail.com, jordi.janer@upf.edu,
ferdinand.fuhrmann@gmail.com, perfecto.herrera@upf.edu

ABSTRACT

The authors address the identification of predominant music instruments in polytimbral audio by previously dividing the original signal into several streams. Several strategies are evaluated, ranging from low to high complexity with respect to the segregation algorithm and models used for classification. The dataset of interest is built from professionally produced recordings, which typically pose problems to state-of-art source separation algorithms. The recognition results are improved a 19% with a simple sound segregation pre-step using only panning information, in comparison to the original algorithm. In order to further improve the results, we evaluated the use of a complex source separation as a pre-step. The results showed that the performance was only enhanced if the recognition models are trained with the features extracted from the separated audio streams. In this way, the typical errors of state-of-art separation algorithms are acknowledged, and the performance of the original instrument recognition algorithm is improved in up to 32%.

1. INTRODUCTION

The amount of music available has dramatically increased in recent years. There is thus a clear need of effectively organizing and retrieving this content. Music Information Retrieval (MIR) is a research field dealing with the extraction of music content information, and can be used for such purposes. Instrumentation is a very useful description of musical data, since it can be exploited successfully in different forms; songs can be retrieved using the information about the presence of an instrument, and the identification of the musical genre is easier when knowledge about the instrumentation is available (e.g. a banjo makes the piece more likely to be country than classical music). Additionally, instrumentation is a key aspect for the perceived similarity in music [1].

Audio source separation deals with the recuperation of the original signals from the acoustical sources constituting an audio mixture by computational means. Even though there is still much room for improvement when applied to real world music, state-of-art separation algorithms can be used to, at least, increasing the presence of a source or a group of sources in a mixture, such as harmonic-percussive separation [10]. They can potentially be a useful pre-step to improve the results of MIR tasks, such as chord detection, melody extraction, etc.

The automatic recognition of instruments is usually based on timbre models or features such as MFCCs or MPEG-7 combined with statistical classifiers. An extensive review of approaches for isolated musical instrument classification can be found in [8], with several classification techniques, a number of instrumental categories below ten, and accuracies that reach up to 90%.

More recent works deal with instrument recognition in polytimbral musical signals, which is a more realistic and demanding problem. For instance, Tzanetakis focused on the detection of voice [12], while Essid [4] presented an approach using a taxonomy-based hierarchical classification, in which the classifiers were trained on combinations of instruments such as: piano, tenor sax, double bass and drums. Kitahara et al. [9] proposed several techniques to improve instrument recognition in duo and trio music by dealing with three issues: the feature variations caused by sound mixtures, the pitch dependency of timbres, and the use of musical context. With the proposed techniques, they achieved an 85.8% average recognition rate in trio music. Fuhrmann [5] proposed a method for automatic recognition of predominant instruments with Support Vector Machine (SVM) classifiers trained with features extracted from real musical audio signals. One of the problems identified in this system is that it often missed some of the labels in excerpts containing more than one predominant instrument.

The recognition of the instruments present in a mixture becomes more complex as the number of instruments increases. Reducing the number of instruments in the audio to be analyzed should thus help in the recognition of instruments, and the idea of using source separation as a pre-step has already been investigated in previous research. Heittola et al. [6] use Non-Negative Matrix Factorization (NMF) with a source filter model, based on previous work by Virtanen and Klapuri [13]. Klapuri's multipitch estimation is used in the separation, with aid of an optional streaming algorithm which organizes individual notes into sound sources. The Viterbi algorithm is then employed to find the most likely sequence of notes. The classifiers use MFCC's (with a 40 channel filter bank) along with their first time derivatives. A Gaussian Mixture Model (GMM) is used to model the instrument conditional densities of the features, and the parameters are estimated using the Expectation Maximization (EM) algorithm from the training material. A Maximum Likelihood classifier is then used for classification. The dataset was artificially created from the RWC dataset, with a maximum of six note polyphony, and 19 different pitched instruments, reaching a 59.1% F1-measure. Burred [3] also presents an instrument classification approach with a stereo blind source separation pre-step, using Gaussian likelihood as a timbre similarity measure. The reported accuracy reaches 86.7%, with polyphony of 2 instruments, and 5 classes. Results are significantly better than in monaural separation, with 79.8% accuracy.

In this paper, we address the combination of source separation and instrument recognition, in order to improve the identification of predominant instruments in professionally produced western music recordings. As opposed to audio data created by artificially mixing several instrument with no musical relation between them, this real world scenario adds more complexity to source separation algorithms, due to several reasons. First, in real world western music, instruments are harmonically related, and thus their spectral components usually share some of the frequencies. Furthermore, effects such as reverbs, delays, etc. make the separation much more difficult. On the other hand, such scenario allows the algorithms to take advantage of the spatial information present in stereophonic recordings.

2. METHOD

This section introduces the methodology proposed to investigate if the performance of an instrument recognition algorithm can be improved with a previous audio segregation step, as introduced in subsection 2.1. The dataset is described in subsection 2.2, and the evaluation methodology is introduced in subsection 2.3.

2.1 Audio segregation for instrument recognition

The algorithm used by Fuhrmann in [5] is considered as the baseline instrument recognition. It is conceived to output a set of labels corresponding to the predominant instruments in an excerpt of polytimbral music. Ten pitched instruments are used in this study: cello, clarinet, flute, acoustic guitar, electric guitar, organ, piano, saxophone, trumpet, violin, and additionally human singing voice. The original system uses SVM, which outputs probabilistic estimates for each of the modeled categories. As previously introduced, the main problem is that it sometimes misses some labels in excerpts with multiple instruments.

The hypothesis is that in order to enhance its performance, a previous step could be performed, separating input audio data into several streams. These streams are then separately processed by the instrument recognition algorithm, resulting in several sets of labels. The sets of labels are then combined and given as output labels. Several segregation methods are considered, as well as different strategies for the label combination, and also several models used for instrument recognition. Figure 1 illustrates the combination of a segregation process followed by the instrument recognition in each of the streams.

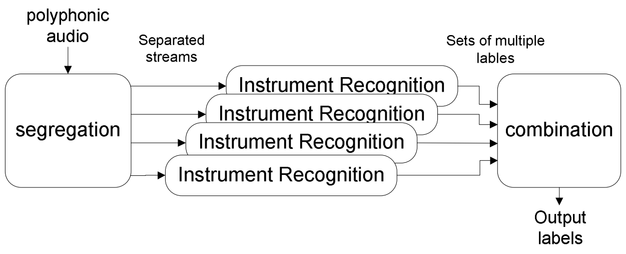


Figure 1. Generic flow diagram for the application of audio segregation as a previous step to the instrument recognition

We consider two different segregation methods. The first is FASST (A Flexible Audio Source Separation Framework), presented by Ozerov et al. [10]. It is based on structured source models, which allow the introduction of constrains according to the available prior knowledge about the separation problem. It aims at generalizing several existing source separation methods, and allows creating new ones. The second segregation method is a simple Left/Right-Mid/Side (LRMS) separation based on panning information, where M = L+R and S = L-R.

In this research, the FASST algorithm is used in a configuration which separates the polytimbral audio input into four streams: "drums", "bass", "melody", and "other" (dbmo). This is a default configuration provided with the FASST framework, and it fits our interest in the recognition of the predominant pitched instruments, as the classifier neither considers bass nor drums. After the separation, the "melody" stream would ideally contain the main instrument to be recognized, and the "other" stream would contain the rest of the instruments, with no presence of bass and percussive instruments. Recognition of the predominant instruments in these streams of audio should be easier than in the case of the original polytimbral mixture. However, there are limitations in most separation algorithms, especially when applied to real world music. They commonly create artifacts and errors in the separation, producing some leakage of instruments in streams where they should not be present. This could affect the recognition of instruments due to the changes the artifacts produce in timbre. In order to deal with these errors, we investigate if a classifier could learn how a source separation algorithm behaves, and acknowledge the errors by training models on the separated audio estimations. In simple words, the models would learn, with the features of the estimated "drums", "bass", "melody", "other" stream, when the predominant instrument of the audio is a cello, clarinet, flute, acoustic guitar, electric guitar, organ, piano, saxophone, trumpet, violin, or voice. We consider the use of different models for each of the separated streams, in order to allow the usage of a different set of (automatically selected) audio features, as well as different parameters for training the classifiers.

Finally, the strategy for the combination of the labels given as output by the individual instrument recognition models is also important. Two strategies are explored in our experiments: 1) selecting some of the classifiers' output only (e.g. only the sets of labels from the "melody" and "other" streams), and 2) requiring a degree of agreement (overlap) between all sets of labels. In the second strategy, output labels correspond to the ones present in more than N sets of labels predicted by the models.

2.2 Data

Two different datasets have been created for training and testing, based on the database originally compiled by Fuhrmann [6]. Firstly, the training dataset contains 6700 annotated excerpts of 3 seconds in which only one instrument is predominant. These data are unevenly distributed among the modeled categories, ranging from minimum 388 to a maximum of 778. A second training dataset is derived by separating the original one into dbmo streams with FASST. Secondly, the testing set consists of around 3000 excerpts annotated with one to five instruments. This set was created by dividing the original music pieces of the original database [6] into segments with the following properties: 1) the predominant instruments are the same in the whole excerpt, 2) the length is between 5 and 20 seconds, and 3) the excerpts are stereo. The first property allows us to disregard segmentation according to the instrumentation into the recognition evaluation, since the predominance of instruments typically changes amongst or even within sections of a music piece. The second property ensures that the instrument labeling process has enough information to output the labels with a certain confidence. The third property corresponds to the use case of interest: professionally produced music recordings, in stereo format.

2.3 Evaluation methodology

The evaluation method is based on comparing the output labels against the manually annotated ground-truth labels. Following the traditional information retrieval evaluation measures, we calculate: true positives (tp), true negatives (tn), false positives (fp) and false negatives (fn) for each of the instruments (labels). We consider L the closed set of labels $L = \{l_i\}$, with $i = 1...N$, N the number of instruments, and the dataset $X = \{x_i\}$, with $i = 1...M$, and M the number of excerpts. We define $\hat{Y} = \{\hat{y}_i\}$, with $i = 1...M$ as the set of ground-truth labels, and $Y = \{y_i\}$, with $i = 1...M$, and $y_i \subseteq L$, the set of predicted labels assigned to each instance i. Precision and recall are defined for each of the labels l in L as:

$$P_l = \frac{tp_l}{tp_l + fp_l} = \frac{\sum_{i=1}^{M} y_{l,i}\hat{y}_{l,i}}{\sum_{i=1}^{M} y_{l,i}} \tag{1}$$

$$R_l = \frac{tp_l}{tp_l + fn_l} = \frac{\sum_{i=1}^{M} y_{l,i}\hat{y}_{l,i}}{\sum_{i=1}^{M} \hat{y}_{l,i}} \tag{2}$$

where $y_{l,i}$ and $\hat{y}_{l,i}$ are boolean variables referring to instance i, which indicate the presence of the label l in the set of predicted labels, or in the set of ground-truth labels respectively. Additionally, we define the F1 as the harmonic mean between precision and recall:

$$F_l = \frac{2P_lR_l}{P_l + R_l} \tag{3}$$

We also define macro and micro averages of the previous metrics, in order to obtain more general performance metrics, which consider all labels. The macro is here understood as an unweighted average of the precision or recall taken separately for each label (average over labels).

$$P_{macro} = \frac{1}{|L|}\sum_{l=1}^{L} P_l\,,\; R_{macro} = \frac{1}{|L|}\sum_{l=1}^{L} R_l \tag{4}$$

On the other hand, the micro average is an average over instances, and thus, giving more weight to the labels with a higher number of instances:

$$P_{micro} = \frac{\sum_{l=1}^{L} tp_l}{\sum_{l=1}^{L}\left(tp_l + fp_l\right)} = \frac{\sum_{l=1}^{L}\sum_{i=1}^{M} y_{l,i}\hat{y}_{l,i}}{\sum_{l=1}^{L}\sum_{i=1}^{M} y_{l,i}} \tag{5}$$

$$R_{micro} = \frac{\sum_{l=1}^{L} tp_l}{\sum_{l=1}^{L}\left(tp_l + fn_l\right)} = \frac{\sum_{l=1}^{L}\sum_{i=1}^{M} y_{l,i}\hat{y}_{l,i}}{\sum_{l=1}^{L}\sum_{i=1}^{M} \hat{y}_{l,i}} \tag{6}$$

The macro and micro F1 are defined as the harmonic mean of respectively, the macro and micro averages.

$$F_{macro} = \frac{2P_{macro}R_{macro}}{P_{macro} + R_{macro}}\,,\; F_{micro} = \frac{2P_{micro}R_{micro}}{P_{micro} + R_{micro}} \tag{7}$$

The following section details the experiments performed according to the presented methodology.

3. EXPERIMENTS

We conducted five experiments to investigate the benefits of the segregation of the audio signal into different streams prior to the application of an instrument recognition algorithm. In the first four experiments, the SVM models used for the instrument recognition were trained with parameters that optimized the performance in Experiment 1: a polynomial kernel of degree 4 and a cost parameter = 0.1. In each experiment, we consider all combinations of sets of labels to find the best recognition performance. These are notated with the initials of the streams considered, e.g.: “Exp3:dbo” refers to the combination of the labels outputted from the recognition of the d (drums) + b (bass) + o (other) streams in Experiment 3. The combination strategy was initially the union of the labels predicted by each of the models. Then, in Experiment 5 we explored a partial overlap strategy, and we optimized recognition performance by tuning the parameters for each of the models.

3.1 Experiment 1: original algorithm

The original algorithm is employed without a previous separation step, as shown in Figure 2:

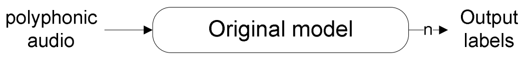


Figure 2: Original instrument recognition algorithm

The labels obtained in this experiment are named “n” for “no separation”. In this configuration, the stereo audio input is transformed into mono, by adding the left and right channels. We obtain the following micro averages: precision = 0.708, recall = 0.258 and F1 = 0.378.

3.2 Experiment 2: Left/Right-Mid/Side separation + original models

In this experiment, audio was segregated into four streams with l = Left, r = Right, n = l+r (Mid), and s = l-r (Side), and the original model was used for classification, as depicted in Figure 5.

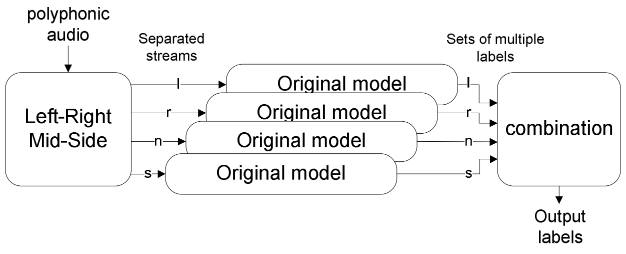


Figure 5. LRMS separation into lrns streams, used as input of the original instrument recognition models.

The label “n” is used for the “Mid” stream in order to be consistent with the notation in Experiment 1, also performed in the addition of the Left and Right channels.

Evaluation results showed that the best combination is with “Exp2:lrns”, obtaining a micro F1 = 0.451. This represents an absolute improvement of 7.3 percent points in the micro F1 with respect to the original algorithm “Exp1:n”, or in relative terms, a 19.3%. This is a considerable improvement, especially taking into account that this is a very simple segregation method which could even be performed in real time.

3.3 Experiment 3: FASST + original models

In this experiment, FASST separation into the bass (b), drums (d), melody (m) and other (o) streams is used, along with the original models for the instrument recognition, as shown in Figure 3.

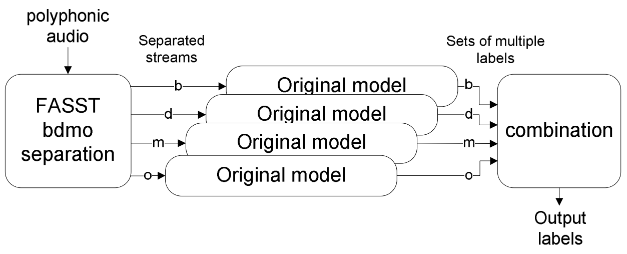


Figure 3. FASST separation into the drum, bass, melody and other streams, combined with the original instrument recognition models.

The evaluation showed that the original algorithm without source separation provides better results than any of the combinations of the labels obtained in Experiment 3. The best micro F1 (0.355) is obtained with a combination of all separated streams (“Exp3:dbmo”). In this case, recall (0.385) is better than with the original algorithm (“Exp1:n”), but precision is quite worse (0.330), so the F1-measure is lower. There is thus a decrease in performance when using source separation as a pre-step of the original instrument recognition models. This is probably due to the fact that the separation is not perfect, there is some energy of instruments in streams where there should not be present, and their timbre is modified. Additionally, the separation algorithm has the drawback of its complexity and execution time, which is above one minute per second of to-be-separated audio (Intel Core 2 Duo @ 2.4 GHz, 4 GB RAM with Windows XP – 32 bits).

3.4 Experiment 4: FASST + models trained with separated audio

In this experiment, the instrument recognition models have been trained with the dbmo audio streams obtained from separating the training dataset with FASST. Four different models have been created; one for each of the output streams of the FASST bdmo separation algorithm, as shown in Figure 4. A different set of features has been automatically selected for each of the SVM models during the training process.

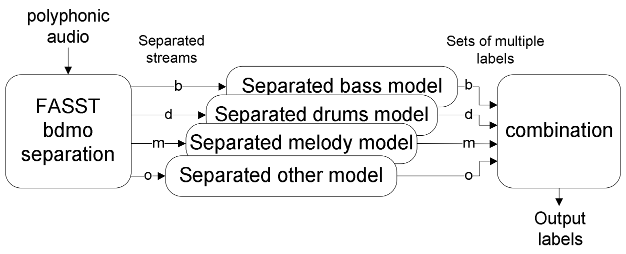


Figure 4. FASST separation into the drum, bass, melody and other streams, combined with the instrument recognition using models trained on the separated audio.

The evaluation showed that using the models trained on each of the streams of separated audio provides better results than using the original models, and better than the original algorithm without any sound segregation. The combination of the "m" and "o" labels already improves the results obtained in "Exp1:n", obtaining a micro F1 = 0.411. The best micro F1-measure (0.446) is obtained with the bdmo combination. If the "n" labels are additionally combined, the micro F1 increases to 0.480.

If we analyze the recognition results per instrument, the best are obtained with the voice, achieving 0.902 precision, 0.574 recall and a 0.701 F1-measure. Clarinet seems to be the most challenging instrument to be recognized, with a F1-measure = 0.113. A further observation is that there is a relation between the stream and the instruments which are better recognized. For instance, the recognition in the bass stream is better for instruments with low frequency content, such as the cello, which is not so well recognized in the rest of the streams.

3.5 Experiment 5: Optimizing the performance of FASST + models trained with separated audio

In this experiment, we aimed at improving the results obtained in Experiment 4 — FASST dbmo separation + models trained with separated audio. Different models are used for the recognition of each of the four audio streams, and thus it is possible to optimize the parameters of each of them. Additionally, we also investigate the requirement of a certain degree N of overlap in the combination of labels. The evaluation showed that if the value of N is increased, the precision increased as well, at the expense of a lower recall. With N = 0 (no overlap required), the obtained micro F1 is equal to 0.446. If N = 1, which is equivalent to outputting only the labels which had been predicted by at least two of the classifiers, we obtain the best precision found in all experiments (0.733), but the recall is considerably reduced (0.354), and the F1-measure is thus smaller. Therefore, the overall performance is considered to be worse when N increases.

As in all previous experiments, the minimum degree of overlap between labels was set to N=0 in Experiment 5, which provided the best results in terms of the F1-measure. The output labels were thus the union of all labels predicted by each of the models. On the other hand, the use of a different configuration for the training of each of the four models led to some improvements in the results, achieving a micro F1= 0.497. In order to further improve the results we tried combining the labels derived from both, source separation and panning-based segregation streams. The combination Exp5:dbmonslr achieved the best F1-measure from all experiments, equal to 0.503.

The most relevant results of all experiments are presented in Table 1.

	Mac Prec	Mac Rec	Mic Prec	Mic Rec	Mac F1	Mic F1
Exp1:n	**57.8**	24.9	**70.8**	25.8	34.9	37.8
Exp2:lrns	48.5	33.8	58.2	36.7	39.8	45.1
Exp3:dbmo	31.0	37.0	33.0	38.5	33.7	35.5
Exp4:dbmo	49.0	30.6	62.5	34.7	37.7	44.6
Exp4:dbmon	47.5	37.3	59.3	40.3	41.8	48.0
Exp5:dbmon	44.0	41.5	54.9	45.5	42.7	49.7
Exp5:dbmolrns	41.0	**45.5**	50.4	**50.1**	**43.2**	**50.3**

Table 1. Instrument recognition measures (in %). See text for details regarding the studied experimental methods and their acronyms

In the following section we analyze the obtained results, and compare the evaluated approaches.

4. DISCUSSION

The highest precision is obtained with the instrument recognition algorithm [5] by itself ("Exp1:n"), at the expense of having a low recall, which provides a medium F1-measure. In "Exp2:nslr" we considerably improve the results with a simple panning-based segregation, achieving a 19.2% relative increase in the micro-F1 with respect to the original algorithm. Experiment 3 makes use of the FASST dbmo separation as a pre-step to the instrument recognition. In this experiment, the precision drops, and the recognition performance is worse. After training the recognition models with source separated data, we obtain considerably better results in "Exp4:dbmo" compared to "Exp3:dbmo" and also "Exp1: n" in terms of F1-measure. With the aggregation of the sets of labels obtained with the original algorithm, we obtain a further increase in the performance in "Exp4:dbmon". The results from "Exp5:dbmon" show that it is possible to further improve the instrument recognition by tuning the parameters of each of the dbmo models. Finally "Exp5:dbmonslr" corresponds to the best results obtained in any of the automatic instrument recognition experiments, by combining "dbmo" sets of labels from the tuned models trained with separated streams, and the "Exp2:nslr" sets of labels obtained with the LRMS separation. The detailed results for all possible combination of labels and experiments can be found in [2]. The best micro F1-measure = 0.503, thanks to the recall gained by the combination of all labels. The micro F1-measure obtained with the original algorithm without segregation was 0.378, so we were able to improve 12.2 percent points, which represents a 32.3% relative to the initial value. It is interesting to note that the micro averages are better than the macro averages, since the majority of categories with the most frequent instances (e.g. voice) are more easily recognized than the rest.

5. CONCLUSIONS

We presented novel methods to improve the automatic recognition of predominant musical instruments, by its combination with audio segregation algorithms. A comparison with previous similar approaches is not straightforward, since the number of classes and datasets are different. However, if we compare the performance of the original algorithm with the best of our presented approaches combining source separation and instrument recognition, there is around 32% improvement of the micro F1-measure. The way in which the combination is made is very important to be able to improve the results of the algorithms: we found that the application of a source separation pre-step may not provide a better recognition of the instruments if the models do not consider the limitations and errors of the separation algorithms. Training the classification models with the different streams of separated audio has been found to be an effective strategy for acknowledging the typical source separation errors. This leads to a better performance, which can be further enhanced by tuning the parameters of each of the different models used in the instrument recognition. A drawback of the use of the proposed separation algorithm is its computational complexity. As a simple, fast and efficient alternative, we propose the decomposition of the stereophonic polytimbral audio into the left, right, mid and side streams, and the combination of the labels identified by the instrument recognition algorithms in each of the streams. This increased a 19.2% the performance of the predominant instrument recognition.

Acknowledgements: This research was partially supported by "La Caixa" Fellowship Program, and the following projects: Classical Planet: TSI-070100-2009-407 (MITYC), DRIMS: TIN2009-14247-C02-01 (MICINN) and MIRES: EC-FP7 ICT-2011.1.5 Networked Media and Search Systems, grant agreement No. 287711.

6. REFERENCES

[1] V. Alluri and P. Toiviainen, "Exploring perceptual and acoustical correlates of polyphonic timbre," *Music Perception*, vol. 27, no. 3, pp. 223–242, 2010.

[2] J.J. Bosch, "Synergies between Musical Source Separation and Instrument Recognition", Master's Thesis, Universitat Pompeu Fabra, 2011.

[3] J. J. Burred, "From sparse models to timbre learning: new methods for musical source separation," PhD Thesis, Technical University of Berlin, Berlin, 2008.

[4] S. Essid, G. Richard, and B. David, "Instrument recognition in polyphonic music based on automatic taxonomies," *IEEE Transactions On Audio Speech and Language Processing*, vol. 14, no. 1, pp. 68-80, 2006.

[5] 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.

[6] F. Fuhrmann and P. Herrera, "Polyphonic Instrument Recognition for exploring semantic Similarities in Music," in *Proc. of 13th Int. Conference on Digital Audio Effects DAFx10*, Graz Austria, 2010, pp. 1-8.

[7] 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.

[8] P. Herrera-Boyer, G. Peeters, and S. Dubnov, "Automatic Classification of Musical Instrument Sounds," *Journal of New Music Research*, vol. 32, no. 1, pp. 3-21, Mar. 2003.

[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 Advances in Signal Processing*, 2007.

[10] N. Ono, K. Miyamoto, H. Kameoka, J. Le Roux, Y. Uchiyama, E. Tsunoo, T. Nishomoto, and S. Sagayama, "Harmonic and percussive sound separation and its application to MIR-related tasks" in *Advances in Music Information Retrieval*. Vol. 274 Springer, 2010, pp. 213–236.

[11] A. Ozerov, E. Vincent, and F. Bimbot, "A general flexible framework for the handling of prior information in audio source separation", *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 4, pp. 1118 – 1133.

[12] G. Tzanetakis, "Song-specific bootstrapping of singing voice structure," in *Proc IEEE International Conference on Multimedia and Expo ICME*, 2004.

[13] T. Virtanen and A. Klapuri, "Analysis of polyphonic audio using source-filter model and non-negative matrix factorization," in *Advances in Models for Acoustic Processing*, Neural Information Processing Systems Workshop, 2006.

LEARNING SPARSE FEATURE REPRESENTATIONS FOR MUSIC ANNOTATION AND RETRIEVAL

Juhan Nam
CCRMA
Stanford University
juhan@ccrma.stanford.edu

Jorge Herrera
CCRMA
Stanford University
jorgeh@ccrma.stanford.edu

Malcolm Slaney
Yahoo! Research
Stanford University
malcolm@ieee.edu

Julius Smith
CCRMA
Stanford University
jos@ccrma.stanford.edu

ABSTRACT

We present a data-processing pipeline based on sparse feature learning and describe its applications to music annotation and retrieval. Content-based music annotation and retrieval systems process audio starting with features. While commonly used features, such as MFCC, are hand-crafted to extract characteristics of the audio in a succinct way, there is increasing interest in learning features automatically from data using unsupervised algorithms. We describe a systemic approach applying feature-learning algorithms to music data, in particular, focusing on a high-dimensional sparse-feature representation. Our experiments show that, using only a linear classifier, the newly learned features produce results on the CAL500 dataset comparable to state-of-the-art music annotation and retrieval systems.

1. INTRODUCTION

Automatic music annotation (a.k.a. music tagging) and retrieval are hot topics in the MIR community, as large collections of music are increasingly available. Therefore, tasks such as music discovery have become progressively harder for humans without the help of computers. Extensive research has been done on these topics [20], [11], [8] [5]. Also, different datasets have become standards to train and evaluate these automatic systems [19], [12].

Training for most automatic systems use audio content—in the form of audio features—as the input data. Traditionally well-known audio features, such as MFCC, chroma and spectral centroid, are used to train algorithms to perform the annotation and retrieval tasks. These "hand-crafted" features usually capture partial auditory characteristics in a highly condensed form, ignoring many details of the input data. While such engineered features have proven to be valuable, there is increasing interest in finding a better feature representation by learning from data in an unsupervised manner. Unsupervised learning is usually conducted either by mapping the input data into a high-dimensional sparse space or by means of deep learning. In this paper, we apply high-dimensional sparse feature-learning to short-term audio spectrograms and construct song-level features for music annotation and retrieval.

In summary, the contributions of this paper are as follows:

- We propose a data preprocessing method to make feature-learning algorithms more effective.
- We demonstrate that the feature-learning algorithms capture rich local timbral patterns of music, useful for discrimination.
- We show that song-level features constructed from the local features achieve results comparable to state-of-art algorithms on the CAL500 dataset using only a linear classifier, and furthermore outperform them with a nonlinear classifier.

1.1 Recent Work

Lee et. al. proposed a Convolutional Deep Belief Network (CDBN) and applied it to the audio spectrogram for music genre and artist classification [14]. Dieleman et. al. also employed the CDBN but on engineered features (EchoNest chroma and timbre features) for artist, genre and key recognition tasks [6]. Our approach is similar to these systems in that the input data is taken from multiple audio frames as an image patch and max-pooling is performed for scalable feature-learning. However, we perform feature learning with a high-dimensional single-layer network and the max-pooling separately after learning the features [2]. While this can limit the representational power, it allows faster and simpler training of the learning algorithms.

Henaff et. al. applied a sparse coding algorithm to a single frame of constant-Q transform spectrogram and aggregated them into a segment-level feature for music genre classification [10]. Likewise, Schlter et. al. compared Restricted Boltzmann Machine (RBM), mean-covariance RBM and DBN on similarity-based music classification [17]. Our approach is also similar to these pipelines. However, in our work we provide deeper insight on the learned features by showing how they are semantically relevant. In addition, we investigate the effect of sparsity and max-pooling on the performance.

Finally, Hamel et. al. showed that simply PCA-whitened spectrogram can provide good performance by combining different types of temporal pooling [9]. Our approach is

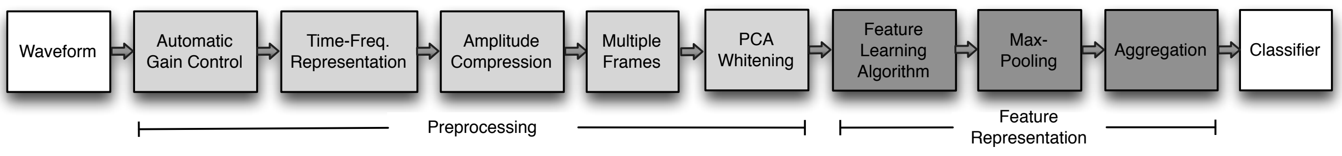


Figure 1: Data processing pipeline for feature representation. This takes an waveform as input and produces a song-level feature for the classifier.

quite different from this work because we encode the PCA-whitened spectrogram into a high-dimensional sparse space and extract features from it.

2. DATA PROCESSING PIPELINE

We perform the music annotation and retrieval tasks using the data processing pipeline shown in Figure 1. Each block in the pipeline is described in this section.

2.1 Preprocessing

Data preprocessing is a very important step to make features invariant to input scale and to reduce dimensionality. We perform several steps of the preprocessing.

2.1.1 Automatic Gain Control

Musical signals are dynamic by nature and each song file in a dataset has different overall volume due to different recording conditions. Thus, we first apply Automatic Gain Control (AGC) to normalize the local energy. In particular, we employ time-frequency AGC using Ellis' method [7]. The AGC first maps FFT magnitude to a small number of sub-bands, computes amplitude envelopes for each band, and uses them to create a time-frequency magnitude envelope over a linear-frequency scale. Then, it divides the original spectrogram by this time-frequency envelope. As a result, the AGC equalizes input signals so they have uniformly distributed spectra over frequency bins.

2.1.2 Time-frequency Representation

A time-frequency representation is an indispensable processing step for musical signals, which are characterized primarily by harmonic or non-harmonic elements. There are many choices of time-frequency representations, each one having different time/frequency resolutions and/or perceptual mappings. In this paper, we chose a mel-frequency spectrogram.

Our initial experiments—based on a spectrogram—showed that using multiple consecutive frames as an input unit for learning algorithms (which is analogous to taking a patch from an image) significantly improves performance over using single frames. However, the FFT size used for musical signals is usually large and thus concatenating multiple frames yields a very high-dimensional vector requiring expensive computation for learning algorithms. Using a moderate number of mel-frequency bins, instead of the straight FFT, preserves the audio content well enough, while significantly reducing the input dimension. We chose 128 mel-frequency bins, following Hamel's work [9], and will present results for various numbers of frames below.

2.1.3 Magnitude Compression

We compress the magnitude using an approximated log scale, $\log_{10}(1 + C|X(t, f)|)$, where $|X(t, f)|$ is the mel-frequency spectrogram and C controls the degree of compression [15]. In general, the linear magnitude of each bin has an exponential distribution. Scaling with a log function gives the magnitude a more Gaussian distribution. This enables the magnitude to be well-fitted with the ensuing PCA whitening, which has an implicit Gaussian assumption.

2.1.4 Multiple Frames

As previously discussed, we take multiple frames as an input unit for feature learning. This approach was used in the convolutional feature-learning algorithm [14]. In that work, however, the multiple frames are taken over the PCA-whitening space where PCA is performed on single frames. In our case, we apply the PCA to multiple consecutive frames for the reasons explained next.

2.1.5 PCA Whitening

PCA whitening is often used as a preprocessing step for independent component analysis or other learning algorithms that capture high-order dependency. It removes pairwise correlation in the input data domain and, as a result, reduces the data dimensionality. Note that the PCA captures short-term temporal correlation as well because it is performed on multiple frames (after vectorizing them).

2.2 Feature Representation

At this point, the input has been processed in a highly reconstructible way so that the underlying structure of the data can be discovered via feature-learning algorithms. In this section, we describe how such algorithms reveal the underlying structure.

2.2.1 Feature Learning Algorithm

We compare three feature-learning algorithms to encode the preprocessed data into high-dimensional feature vectors: K-means clustering, Sparse Coding and Sparse Restricted Boltzmann Machine.

K-means Clustering: K-means clustering learns K centroids from the input data and assigns the membership of a given input to one of the K centroids. In the representational point of view, this can be seen as a linear approximation to the input vectors, $x \approx Ds$, where D is a dictionary (centroids) and s is an extremely sparse vector that has all zeros but a single "1" that corresponds to the assigned centroid. We use the encoded vector, s, as learned features.

Sparse Coding (SC): Sparse coding is an algorithm to represent input data as a sparse linear combination of elements in a dictionary. The dictionary is learned using the L1-penalized sparse coding formulation. In our experiments, we optimize

$$\begin{aligned}\min_{D,s^{(i)}} \sum_i \left\| Ds^{(i)} - x^{(i)} \right\|_2^2 + \lambda \left\| s^{(i)} \right\|_1 \\ \text{subject to } \left\| D^{(j)} \right\|_2^2 = 1, \forall j\end{aligned} \quad (1)$$

using alternating minimization over the sparse codes $s^{(i)}$, and the dictionary D [3]. We use the absolute value of the sparse code s, as learned features.

Sparse Restricted Boltzmann Machine (sparse RBM): The Restricted Boltzmann Machine is a bipartite undirected graphical model that consists of visible nodes $\mathbf{x}$ and hidden nodes $\mathbf{h}$ [18]. The visible nodes represent input vectors and the hidden nodes represent the features learned by training the RBM. The joint probability for the hidden and visible nodes is defined in Eq. 2 when the visible notes are real-valued Gaussian units and the hidden notes are binary units. 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 nodes in either layer is observed (Eq. 3 and 4).

$$\begin{aligned}-\log P(\mathbf{x},\mathbf{h}) \propto E(\mathbf{x},\mathbf{h}) = \\ \frac{1}{2\sigma^2}\mathbf{x}^T\mathbf{x} - \frac{1}{\sigma^2}\left(\mathbf{c}^T\mathbf{x} + \mathbf{b}^T\mathbf{h} + \mathbf{h}^T W\mathbf{x}\right)\end{aligned} \quad (2)$$

$$p(h_j|\mathbf{x}) = sigmoid(\frac{1}{\sigma^2}(b_j + \mathbf{w}_j^T\mathbf{x})) \quad (3)$$

$$p(x_i|\mathbf{h}) = \mathcal{N}((c_i + \mathbf{w}_i^T\mathbf{h}), \sigma^2), \quad (4)$$

where σ^2 is a scaling factor, $\mathbf{b}$ and $\mathbf{c}$ are bias terms, and W is a weight matrix. The parameters are estimated by maximizing the log-likelihood of the visible nodes. This is performed by block Gibbs sampling between two layers, particularly, using contrastive-divergence learning rule which involves only a single step of Gibbs sampling.

We further regularize this model with sparsity by encouraging each hidden unit to have a pre-determined expected activation using a regularization penalty:

$$\lambda \sum_j (\rho - \frac{1}{m}(\sum_{k=1}^{m} \mathbf{E}[h_j|\mathbf{x}^k]))^2, \quad (5)$$

where $\{\mathbf{x}^1, ..., \mathbf{x}^m\}$ is the training set and ρ determines the target sparsity of the hidden unit activations [13].

Similar to K-means clustering and SC, we can interpret Eq. 4 as approximating input vectors, $\mathbf{x}$, with a linear combination of elements from dictionary W. That is, $\mathbf{x} \approx W\mathbf{h}$ (ignoring the bias term, $\mathbf{c}$). The advantage of RBM over the two algorithms is that the RBM has an explicit encoding scheme, $\mathbf{h} = sigmoid(\frac{1}{\sigma^2}(\mathbf{b} + W^T\mathbf{x})$ from Eq. 3. This enables much faster computation of learned features than SC.

2.2.2 *Pooling and Aggregation*

A song is a very long sequence of data. There are many ways to summarize the data over the entire song. A typical approach to construct a long-term feature is aggregating short-term features by computing statistical summaries over the whole song. However, summarizing all short-term feature over a song dilutes their local discriminative characteristics. Instead, we pool relevant features over smaller segments and then aggregate them by averaging over all the segments in a song.

Since the learned feature vectors are generally sparse and high-dimensional, we performed max-pooling over segments of the song. Max-pooling is an operation that takes the maximum value at each dimension over a pooled area. This is often used in the setting of convolutional neural networks to make features invariant to local transformation. In our experiments, it is used to reduce the smoothing effect of the averaging. In Section 4 we discuss how the pooling size is determined.

2.3 Classification

Music annotation is a multi-labeling problem. We tackle this by using multiple binary classifiers, each predicting the presence of an annotation word. The binary classifier also returns the distance from the decision boundary given a song-level feature. We used the distance as a confidence measure of relevance between a query word and a song for music retrieval.

2.3.1 *Linear SVM*

We use a linear SVM as a reference classifier to evaluate the song-level feature vectors learned by different settings of feature representation. We trained the linear SVM by minimizing the hinge loss given training data. By combining the hinge loss for multiple SVMs as a single objective, we trained them simultaneously, avoiding individual cross-validation for each SVM and thereby saving computation time [16].

2.3.2 *Neural Network*

We also applied a neural network to improve classification performance. For simple evaluation, we used a single hidden layer. However, instead of the cross-entropy, which is usually used as a cost function for a neural network, we employed the hinge loss from the linear SVM above, so that the penalty term is consistent between classifiers. That way, performance difference can be attributed only to the inclusion of the hidden layer.

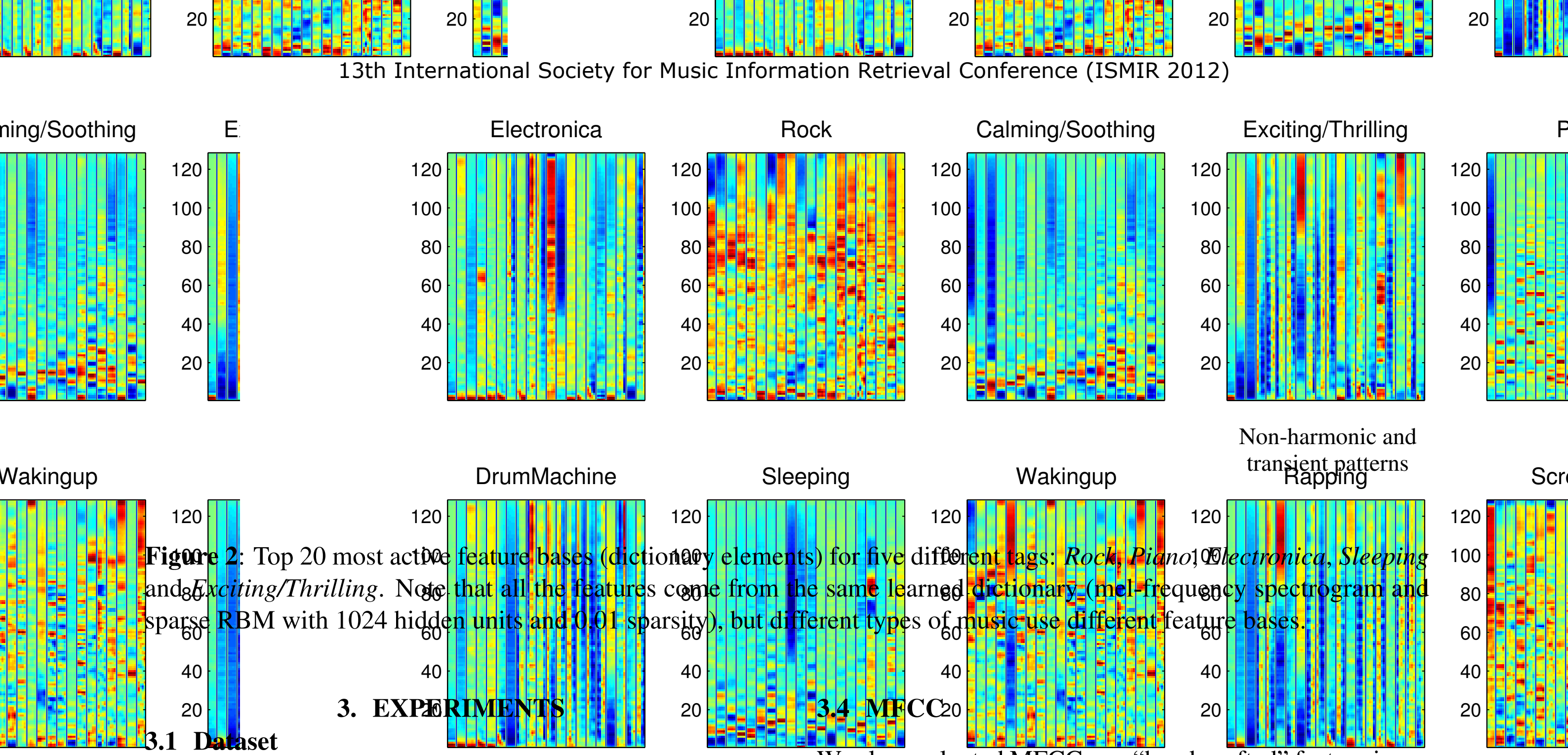


Figure 2: Top 20 most active feature bases (dictionary elements) for five different tags: *Rock*, *Piano*, *Electronica*, *Sleeping* and *Exciting/Thrilling*. Note that all the features come from the same learned dictionary (mel-frequency spectrogram and sparse RBM with 1024 hidden units and 0.01 sparsity), but different types of music use different feature bases.

3. EXPERIMENTS

3.1 Dataset

We evaluated our proposed feature representation on the CAL500 dataset [19]. This dataset contains 502 western songs, each of which was manually annotated with one or more tags out of 174 possibilities, grouped in 6 categories: *Mood*, *Genre*, *Instrument*, *Song*, *Usage*, and *Vocal*. In our experiments, we considered only 97 tags with at least 30 example songs, to be able to compare with results reported elsewhere [20], [4], [8] [5]. In order to apply the full path of our pipeline, we obtained MP3 files of the 502 songs and used the decoded waveforms.

3.2 Preprocessing Parameters

We first resampled the waveform data to 22.05kHz and applied the AGC using 10 sub-bands and attack/delay smoothing the envelope on each band. We computed an FFT with a 46ms Hann window and 50% overlap, which produces a 513 dimensional vector (up to half the sampling rate) for each frame. We then converted it to a mel-frequency spectrogram with 128 bins. In the magnitude compression, C was set to 10 (see section 2.1.3). For PCA whitening and feature learning steps, we sampled 100000 data examples, approximately 200 examples at random positions within each song. Each example is selected as a $128 \times n$ (n=1, 2, 4, 6, 8 and 10) patch from the mel-frequency spectrogram. Using PCA whitening, we reduced the dimensionality of the examples to 49, 80, 141, 202, 263 and 323 for each n by retaining 90% of the variance. Before the whitening, we added 0.01 to the variance for regularization.

3.3 Feature Representation Parameters

We used dictionary size (or hidden layer size) and sparsity (when applicable) as the primary feature-learning parameters. The dictionary size was varied over 128, 256, 512 and 1024. The sparsity parameter was set to $\rho = 0.01$, 0.02, 0.03, 0.05, 0.07 and 0.1 for sparse RBM and λ = 0.5, 1.0, 1.5 and 2.0 for sparse coding. Max-pooling was performed over segments of length 0.1, 0.25, 0.5, 1, 2, 4, 8, 16, 32 and 64 seconds.

3.4 MFCC

We also evaluated MFCC as a "hand-crafted" feature in order to compare it to our proposed feature representation. Instead of using the MFCC provided from the CAL500 dataset, we computed our own MFCC to match parameters as close as possible to the proposed feature. We used the same AGC and FFT parameters but 40 bins for mel-frequency spectrogram and then applied log and DCT. In addition, we formed a 39-dimensional feature vector by combining its delta and double delta and normalized it by making the MFCC have zero mean and unit variance. The MFCC was also fed into either the classifier directly or the feature-learning step.

3.5 Classifier Parameters

We first subtracted the mean and divided by the standard deviation of each song-level feature in the training set and then trained the classifiers with the features and hard annotation using 5-fold cross-validation. In the neural network, since the classifier is not our main concern, we simply fixed the hidden layer size to 512. After training, we adjusted the distance from the decision boundary using the diversity factor of 1.25, following the heuristic in [11].

4. EVALUATION AND DISCUSSION

4.1 Annotation and Retrieval Performance Metrics

The annotation task was evaluated using Precision, Recall and F-score, following previous work. Precision and Recall were computed based on the methods described by Turnbull [20]. The F-score was computed by first calculating individual F-scores for each tag and then averaging the individual F-scores, similarly to what was done by Ellis [8]. It should be noted that averaging individual F-scores tends to generate lower average F-score than computing the F-score from mean precision and recall values. As for the retrieval, we used the area under the receiver operating characteristic curve (AROC), mean average precision (MAP) and top-10 precision (P10) [8].

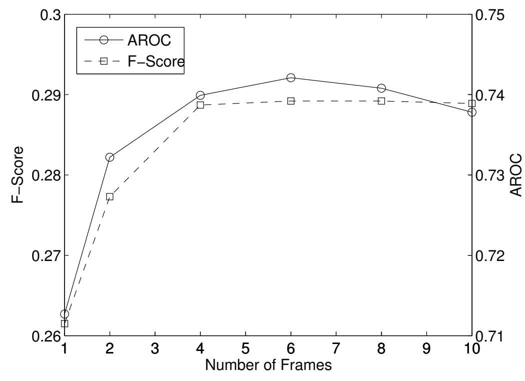


Figure 3: Effect of number of frames (Sparse RBM with 1024 hidden units)

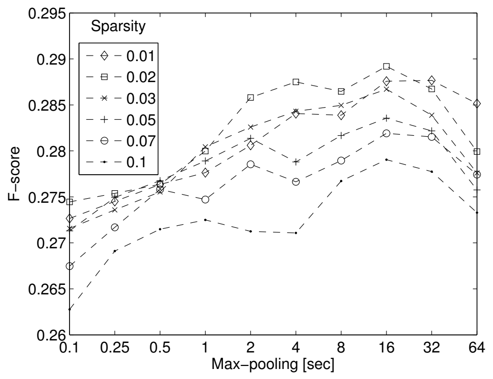


Figure 4: Effect of sparsity and max-pooling (Sparse RBM with 1024 hidden units)

4.2 Visualization

Figure 2 shows most active top-20 feature bases learned on the CAL500 for each tag. They are vividly distinguished by different timbral patterns, such as harmonic/non-harmonic, wide/narrow band, strong low/high-frequency content and steady/transient ones. This indicates the feature learning algorithm effectively maps input data to high-dimensional sparse feature vectors such that the feature vectors (hidden units in RBM) are "selectively" activated by given music.

4.3 Results and Discussion

We discuss the effect of parameters in the pipeline on the annotation and retrieval performance.

4.3.1 Number of Frames

Figure 3 plots F-score and AROC for different number of frames (patch size) taken from the mel-frequency spectrogram. It shows that the performance significantly increases between 1 and 4 frames and then saturates beyond 4 frames. It is interesting that the best results are achieved at 6 frames (about 0.16 second long). We think this is related to the representational power of the algorithm. That is, when the number of frames is small, the algorithm is capable of capturing the variation of input data. However, as the number of frames grows, the algorithm becomes incapable of representing the exponentially increasing variation, in particular, temporal variation.

	Annotation			**Retrieval**		
Data+Algorithm	Prec.	Recall	F-score	AROC	MAP	P10
With AGC						
MFCC only	0.399	0.223	0.242	0.713	0.446	0.467
MFCC+K-means	0.446	0.240	0.270	0.732	0.471	0.492
MFCC+SC	0.437	0.232	0.260	0.713	0.452	0.476
MFCC+SRBM	0.441	0.235	0.263	0.725	0.463	0.485
Mel-Spec+K-means	0.467	0.252	0.287	0.740	0.488	**0.520**
Mel-Spec+SC	0.468	0.252	0.286	0.734	0.482	0.507
Mel-Spec+SRBM	**0.479**	**0.257**	**0.289**	**0.741**	**0.489**	0.513
Without AGC						
MFCC only	0.399	0.222	0.239	0.712	0.444	0.460
MFCC+K-means	0.438	0.237	0.267	0.727	0.465	0.489
Mel-Spec+SRBM	0.449	0.244	0.274	0.727	0.477	0.503

Table 1: Comparison of the performance for different input data and feature learning algorithms. These results are all based on a linear SVM.

4.3.2 Sparsity and max-pooling size

Figure 4 plots F-score for a set of sparsity values and max-pooling sizes. It shows a clear trend that higher accuracy is achieved when the feature vectors are sparse (around 0.02) and max-pooled over segments of about 16 seconds.[1] These results indicate that the best discriminative power in song-level classification is achieved by capturing only a few important features over both timbral and temporal domains.

4.3.3 Input Data, Algorithms and AGC

Table 1 compares the best results on features learned on different types of input data and feature learning algorithms. As shown, the mel-frequency spectrogram significantly outperforms MFCC regardless of the algorithms. Among the feature learning algorithms, K-means and sparse RBM generally perform better than SC. In addition, the results show that the AGC significantly improves both annotation and retrieval performance, regardless of the input features.

4.3.4 Comparison to state-of-the-art algorithms

Table 2 compares our best results to those of state-of-the-art algorithms. They all use MFCC features as input data and represent them either using a Gaussian Mixture Model (GMM), as a bag of frames [20], or Dynamic Texture Mixture (DTM) [4]. They have progressively improved their performance by building on the previous systems, such as, in Bag of Systems (BoS) [8] or Decision Fusion (DF) *decision-fusion*. However, our best system trained with a *linear* SVM shows comparable results. In addition, with nonlinear neural-network classification, our system outperforms the prior algorithms in F-score and all retrieval metrics.

[1] We found that the average length of songs on the CAL500 dataset is approximately 250 seconds, which suggests that aggregating about 16 ($\approx$ 250/16) max-pooled feature vectors over an entire song is an optimal choice.

	Annotation			Retrieval		
Methods	Prec.	Recall	F-score	AROC	MAP	P10
HEM-GMM [20]	0.374	0.205	0.213	0.686	0.417	0.425
HEM-DTM [4]	0.446	0.217	0.264	0.708	0.446	0.460
BoS-DTM-GMM-LR [8]	0.434	**0.272**	0.281	0.748	0.493	0.508
DF-GMM-DTM [5]	**0.484**	0.230	0.291	0.730	0.470	0.487
DF-GMM-BST-DTM [5]	0.456	0.217	0.270	0.731	0.475	0.496
Proposed methods						
Mel-Spec+SRBM+SVM	0.479	0.257	0.289	0.741	0.489	0.513
Mel-Spec+SRBM+NN	0.473	0.258	**0.292**	**0.754**	**0.503**	**0.527**

Table 2: Performance comparison: state-of-the-art (top) and proposed methods (bottom).

5. CONCLUSION AND FUTURE WORK

We have presented a sparse feature representation method based on unsupervised feature-learning. This method was able to effectively capture many timbral patterns of music from minimally pre-processed data. Using a simple linear classifier, our method achieved results comparable to state-of-the-art algorithms for music annotation and retrieval tasks on the CAL500 dataset. Furthermore, our system outperformed them with a non-linear classifier.

To ensure the discriminative power of our proposed feature representation method, we need to evaluate it on larger datasets, such as, the Million Song Dataset [1] or Magnatagatune [12] and also for different classification tasks.

6. REFERENCES

[1] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *ISMIR*, 2011.

[2] A. Coates, H. Lee, and A. Ng. An analysis of single-layer networks in unsupervised feature learning. *Journal of Machine Learning Research*, 2011.

[3] A. Coates and A. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *ICML*, 2011.

[4] E. Coviello, A. Chan, and G. Lanckriet. Time series models for semantic music annotation. *IEEE Transactions on Audio, Speech, and Language Processing*, 2011.

[5] E. Coviello, R. Miotto, and G. Lanckriet. Combining content-based auto-taggers with decision-fusion. In *ISMIR*, 2011.

[6] S. Dieleman, P. Brakel, and B. Schrauwen. Audio-based music classification with a pretrained convolutional network. In *ISMIR*, 2011.

[7] D. Ellis. Time-frequency automatic gain control. web resource, available, `http://labrosa.ee.columbia.edu/matlab/tf_agc/`, 2010.

[8] K. Ellis, E. Coviello, and G. Lanckriet. Semantic annotation and retrieval of music using a bag of systems representation. In *ISMIR*, 2011.

[9] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *ISMIR*, 2011.

[10] H. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun. Unsupervised learning of sparse features for scalable audio classification. In *ISMIR*, 2011.

[11] M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In *ISMIR*, 2009.

[12] E. Law and L. 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.

[13] H. Lee, C. Ekanadham, and A. Ng. Sparse deep belief net model for visual area v2. *Advances in Neural Information Processing Systems*, 2007.

[14] H. Lee, P. Pham Y. Largman, and A.Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Advances in Neural Information Processing Systems*, 2009.

[15] M. Müller, D. Ellis, A. Klapuri, and G. Richard. Signal processing for music analysis. *IEEE Journal on Selected Topics in Signal Processing*, 2011.

[16] J. Nam, J. Ngiam, H. Lee, and M. Slaney. A classification-based polyphonic piano transcription approach using learned feature representation. In *ISMIR*, 2011.

[17] J. Schlter and C. Osendorfer. Music Similarity Estimation with the Mean-Covariance Restricted Boltzmann Machine. In *ICMLA*, 2011.

[18] P. Smolensky. *Information processing in dynamical systems:Foundation of harmony theory*. MIT Press, Cambridge, 1986.

[19] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Towards musical query-by-semantic description using the CAL500 data set. In *ACM Special Interest Group on Information Retrieval Conference*, 2007.

[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*, 2008.

SCORE ANALYZER: AUTOMATICALLY DETERMINING SCORES DIFFICULTY LEVEL FOR INSTRUMENTAL E-LEARNING

Véronique Sébastien, Henri Ralambondrainy, Olivier Sébastien, Noël Conruyt
IREMIA - Laboratoire d'Informatique et de Mathématiques, EA2525
University of Reunion Island, Saint-Denis, Reunion (FRANCE)
veronique.sebastien/henri.ralambondrainy/olivier.sebastien /noel.conruyt@univ-reunion.fr

ABSTRACT

Nowadays, huge sheet music collections exist on the Web, allowing people to access public domain scores for free. However, beginners may be lost in finding a score appropriate to their instrument level, and should often rely on themselves to start out on the chosen piece. In this instrumental e-Learning context, we propose a Score Analyzer prototype in order to automatically extract the difficulty level of a MusicXML piece and suggest advice thanks to a Musical Sign Base (MSB). To do so, we first review methods related to score performance information retrieval. We then identify seven criteria to characterize technical instrumental difficulties and propose methods to extract them from a MusicXML score. The relevance of these criteria is then evaluated through a Principal Components Analysis and compared to human estimations. Lastly we discuss the integration of this work to @-MUSE, a collaborative score annotation platform based on multimedia contents indexation.

1. INTRODUCTION

In the context of knowledge transmission, musical know-how presents specific features to be efficiently preserved and shared. Indeed, to play correctly and nicely an instrument, one should at the same time acquire physical (gestures, hands position, listening) and intellectual (music theory, score reading) skills. As such, conceiving a service to preserve, transmit and share musical know-how is a complex issue, as we deal with both music hearing faculties and artistic gestures production.

While more and more instrumental e-Learning services are proposed to music amateurs (Garage Band[1], Song2See[2], iScore[3]), few of them aims at sharing instrumental know-how on a large scale. Therefore, we propose to build a Musical Sign Base (MSB), grounded on the Sign Management methodology [1], in order to collect annotated performances (personal interpretations or stances) each related to a given musical work (class). This base can be used to compare various performances from music experts or students, and also to dynamically build new music lessons from the available content. To allow musicians to feed this base, we designed a collaborative score annotation platform: @-MUSE (@nnotation platform for MUSical Education). It allows users to illustrate abstract scores (notation) with dia content depicting advices, exercises or questions dexed on the piece (annotation) [2]. However, learners may want to be guided in their choice of a new piece to learn, and to obtain rapidly some starting recommendations to begin learning it on appropriate bases, before any teacher can annotate the piece. That is why, annotations created previously on similar pieces can be useful in this frame in order to depict basic information on the new piece.

To do so, we present in this paper a Score Analyzer prototype in order to automatically identify remarkable parts in a musical piece, from a performer viewpoint. For the time being, we choose to concentrate on the piano for several reasons: the authors are pianists and work in collaboration with piano and guitar experts from music conservatories, but also, the piano repertoire is extremely rich, both historically and technically. Indeed, we want our system to be able to manage not only basic know-how, but also advanced one, on virtuoso instrumental works.

In the first part of this work, we explore existing methods to automatically extract musicological and technical information from a digital score. For this knowledge to be relevant to performers, we base this study on the needs of a pianist who would discover a new piece, following the process generally used by piano teachers to introduce a new work to their students. We then propose seven criteria to characterize technical instrumental difficulties and give methods to extract them from a MusicXML score. The relevance of these criteria is then evaluated through a Principal Components Analysis (PCA) and compared to human estimations. Lastly we discuss the integration of this work to @-MUSE, our collaborative score annotation platform.

[1] http://www.apple.com/fr/ilife/garageband/
[2] http://www.songquito.com/index.php/en/
[3] http://rcmusic.ca/iscore-home-page

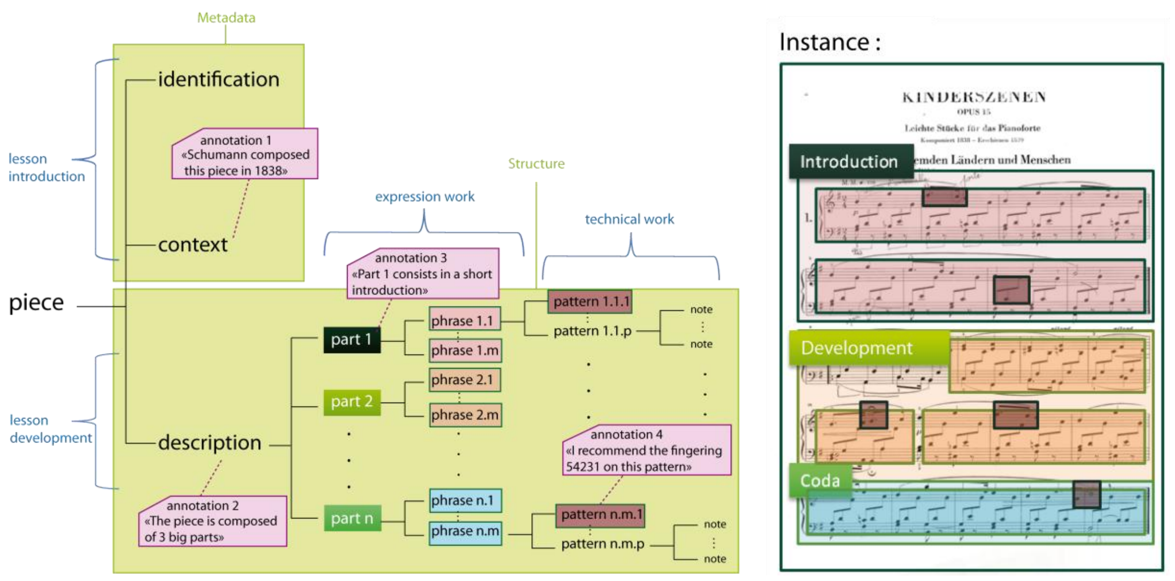


Figure 1. Generic model for musical pieces descriptive logics

2. MUSIC EDUCATION AND ARTIFICIAL INTELLIGENCE

The learning of an instrument generally consists in assimilating a basic repertoire to progress while enjoying playing real artistic compositions instead of only repeating scales mechanically, which can be boring and demotivating. Most of these technical points are directly dealt in the context of the considered pieces. This is why it is essential to select an appropriate corpus for the learner, and to quickly detect remarkable technical points in order to assimilate them, and then concentrate on higher-level considerations, such as expression and musicality. Pointing such features is generally the job of the teacher, until the learner is able to detect them by himself (self-regulation). In the frame of the @-MUSE project, our aim is to assist musicians in this procedure using descriptive logics adapted to each piece genre (baroque, classical, romantic, etc). Figure 1 details a generic model to instance each descriptive logic. It is derived from how teachers introduce new pieces to their students [4]. To extract the different necessary information, we use the standard MusicXML format [3] which describes scores logically, staff by staff, measure by measure, and lastly note by note (Figure 2).

As shown on Figure 1, the first step in our model consists in placing the musical work in its context (composer, period, form metadata). In our frame, it can

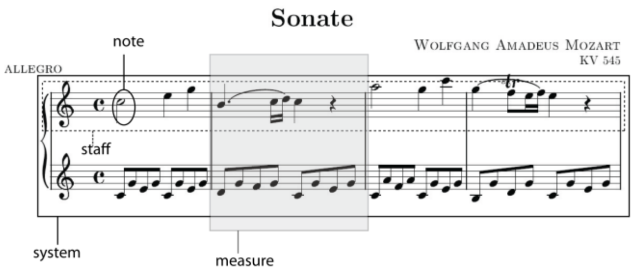


Figure 2. Musical score logical structure

be done using metadata such as title or composer, present in the MusicXML file. In addition, specialized music web services such as MusicBrainz[1] or Last.fm[2] can be queried to obtain more metadata to illustrate the piece (for instance, a portrait and biography of the composer, or an indication about the piece style). Several performances of the piece can be retrieved from video sharing websites in order to get a glimpse of how the piece should sound.

The second step is to analyze the global form of the piece. Most information about it exists within the piece title (i.e.: Sonata, Fugue, etc.). The challenge is thus to detect the main parts of the piece which characterize its form (i.e.: Introduction, part 1, part 2, Coda). Indeed, grasping its structure is essential to performers, as each part may sound totally differently (especially on advanced pieces). In our frame, this also enables a better indexation for annotations. To achieve that goal, we propose to rely on some of the characteristic tags within the MusicXML file. Indeed, score symbols such as direction texts (e.g. “meno mosso”), tempo and key modifications, double bars generally indicate the beginning of a new part within the piece. While this method seems quite "naïve", it gives acceptable results most of the time. Some exceptions may occur, especially on contemporary pieces, which present unconventional structures.

After indicating main parts of the piece, the teacher generally brings the attention of the learner on the remarkable rhythmic or harmonic patterns the piece is build on (if any), leading to more technical and detailed practice. In our work, discovering predefined patterns such as scales, arpeggios or trills may be done using a memory window of successive intervals. Indeed, scales will correspond to sequences of ascendant or descendant seconds, arpeggios to sequences of triples, etc. Each detected pattern can then be linked to a generic annotation explaining how to work on it. However, detecting more complex and

[1] http://musicbrainz.org, visited on the 10/04/2012.
[2] http://www.last.fm, visited on the 10/04/2012.

non-determined patterns remains a challenge, as it does not only involve rhythms and pitch features, but also polyphonic ones. Moreover, it does not present a unified definition of "similarity". Two fragments can be considered as "similar", without having the same pitches, but by possessing similar intervals (transposition). Several works exist on Musical Pattern Discovery. Among them, [5] presents a method based on time windows and define different types of patterns (abstract patterns, prefixes, patterns network). Still, each suggestion given by our system calls for a validation by a music professional.

In order to semantically annotate the detected structures, we need a musical form ontology. While the Music Ontology [6] is particularly fitted to the music industry, it lacks some concepts to be effective in music education. More specialized ontologies exist, such as the Symbolic Music Ontology (allowing to manipulate Voices and Motifs concepts), the Chord Ontology or the Neuma ontology (for Gregorian Music) [7], however, a real form taxonomy has yet to be built to manage the manipulation of concepts such as Sonata, Fugue, Theme or Coda.

The last step of our introduction lesson is to underline specific difficulties of the piece. This will allow us to both specify the global level of the piece, and to detect its technical difficulties measure by measure. To do so, we propose in what follows seven criteria to evaluate a piano piece difficulty.

3. CRITERIA DEFINITION AND RETRIEVAL

In Table 1, we propose seven criteria affecting the level of a piece for the piano and detail how they can be estimated from a MusicXML file. These criteria were defined on the base of pianists experiences, both professionals and amateurs. They may be applied to other instruments with some adaptations (see Instruments column in Table 1). Globally, a piano piece difficulty depends on its tempo, its fingering, its required hand displacements, as well as its harmonic, rhythmic and polyphonic features. Although we define each criteria separately, they affect each other in a complex manner. In particular, fingerings remain hard to extract from a score, as most MusicXML files do not contain this information. Indeed, while other criteria reside in the basic notation layer (notes pitch and duration), the fingering is from the annotation layer and directed at humans only (human performance information).

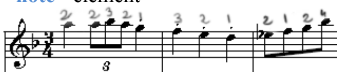
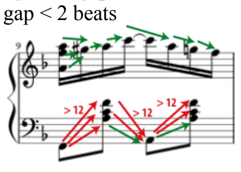
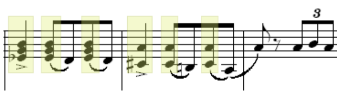
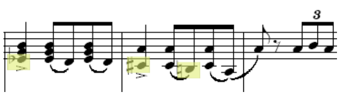
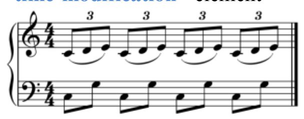
Performance difficulty criterion	Definition	MusicXML implementation	Instruments
Playing speed	The required fingers velocity to play the piece. Depends on the tempo and the shortest significant note value (i.e. a piece presenting a high tempo may contain only long values, and conversely, a piece with a low tempo may contain groups of short notes thus increasing the required fingers agility for the players)	<note><type> elements Tempo attribute in <sound> element	All
Fingering	Fingering: choice of finger and hand position on various instruments. Different notations exist according to the instrument. (ex: in piano: 1 = thumb, 2 = index finger, 3 = middle finger, etc.) Cost functions are used on intervals to extract the general fingering difficulty level See [8][8][9] for more detail.	<fingering> element within each <note> element 	All, requires adaptations in constraints and costs functions (some instruments do not use thumbs)
Hand Displacement	Ratio of hands displacements greater than an octave (12 semitones). Depends on the duration of the interval: if the duration exceeds 2 beats (i.e. 2 quarters in 4/4, 2 eights in 6/8), the displacements is not considered as difficult. The difficulty degree of the displacement evolves with its size (in pitch), its duration and its fingering	Combined <note> elements where <pitch> gap > 12 and <duration> gap < 2 beats 	All, requires adaptations depending on the instrument morphology
Polyphony	Chords ratio (aggregate of musical pitches simultaneously attacked) Polyphonic difficulties may increase with the number of notes played at the same time and their fingerings. Simultaneous voices (in a Fugue for instance) constitute special cases of polyphonic difficulties to treat.	<chord> element 	All (except for monophonic instruments, such as the flute)
Harmony	Ratio of differences from the piece main tonality. Characterized by the amount of accidental alterations.	<alter> and <accidental> elements 	All
Irregular Rhythm	Ratio of irregular polyrhythms (simultaneous sounding of two or more independent rhythms). Example: synchronizing a triplets over duplets	<time-modification> element 	All (except for monophonic instruments)
Length	The number of pages of the score. May also be measured in bars number to avoid dependency to the page layout.	new-page attributes or <measure> elements	All

Table 1. Performance difficulty criteria in piano practice

Several works present methods to automatically deduce fingerings on a given musical extract for piano ([8][9][10]). Most of them are based on dynamic programming. All possible fingers combinations are generated and evaluated, thanks to cost functions. The latter are determined by kinematic considerations. Some functions, even consider the player's hand size to adjust its results. Then, expensive (in term of effort) combinations are suppressed until only one remains, which will be displayed as the resulting fingering. While the result often differs from a fingering determined by a human expert, it remains largely playable and exploitable in the frame of an educational usage. However, few algorithms can process polyphonic extracts, and many other cases are ignored (i.e., left hand, finger substitutions, black and white keys alternation).

Even if more work is needed on this issue, the use of cost functions remains relevant as it is close from the process humans implicitly apply while working on a musical piece. Therefore, we use this method in our Score Analyzer prototype to translate extracted criteria into difficulty indicators (see part 5). But to do so, we need to study how our criteria discriminate a corpus of piano pieces, both objectively (through a components analysis) and subjectively (based on pianists experience).

4. PIANO SCORES CORPUS CLUSTERING

To study how our criteria discriminate scores, we realized a PCA on a sample of fifty piano pieces (Figure 5). The pieces were selected to be representative of a classical piano cursus in a French Music Conservatory. Most pieces concern intermediate to advanced players, fewer target beginners and virtuosi. Most MusicXML files were retrieved from online music notation communities such as MuseScore.com, Noteflight or the Werner Icking Music Archives. Some were generated from PDF files using the SmartScore™ OCR software.

The criteria defined in Table 1 were extracted on each piece. Displacements, chords and harmonic characteristics are distinguished whether they occur on the right (RH) or the left hand (LH). Fingerings were not exploited for the time being as work is in progress to deduce them from MusicXML files (see part 3). Our analysis thus counts 9 numeric variables (Figure 4), and 1 nominal variable (composer). Each ratio is calculated on the base of the total number of notes (e.g. harmonic criteria), or the total number of hands positions (e.g. displacements, chords) within the piece. A displacement is thus defined as a pair composed of two successive hand positions.

A correlations study (Figure 3) points out some links between variables. Some are musically natural (i.e. harmonyLH and harmonyRH, harmonic characteristics concern both hands). We also note a strong correlation (81%) between chordsLH and displacementsLH. This value could characterize accompaniments presenting an alternation of a low-pitched bass and a middle or high-pitched chord, thus inducing regular large displacements and chords at the left hand (ragtime, waltz). Lastly, the piece length can be linked to its playing speed, which characterizes advanced and virtuosi works, demanding an important fingers velocity on a long duration (stamina).

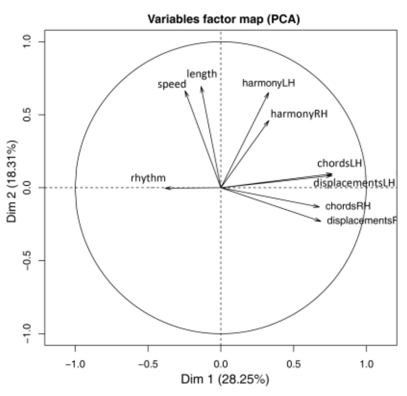


Figure 3. Variables correlation map

The PCA then gives an optimal projection of each piece in the 2D space of the first principal components. Figure 5 presents this projection as well as the three classes detected by the analysis. This clustering was realized through a hierarchical clustering using the Ward's method [11] on the first few principal components. The resulting tree is then cut according to its corresponding indices, in order to find an appropriate number of clusters. Lastly, this clustering is consolidated using a k-means algorithm. The first interpretation of these three classes validates the relevance of our criteria to estimate the difficulty level of a piano piece. Indeed, we notice that at least two of the classes naturally regroup pieces according to their level (class 1 and 2). A further observation backed by a Student test (variable means comparisons between the whole population and the clusters) gives a better interpretation of the classes. Class 1 mostly regroups pieces addressed to beginners (*Kinderszenen*, Schumann's *Choral*) and to intermediate musicians (Bach's *Invention*, *Sonatines*). The Student test confirms this tendency, as most variables remain below average for this class: few chords, displacements and pages, simple harmonies (C major or A minor). Yet, the tempo remains lively. Rhythmic difficulties are noticeable on intermediate pieces. They generally feature characteristic rhyth-

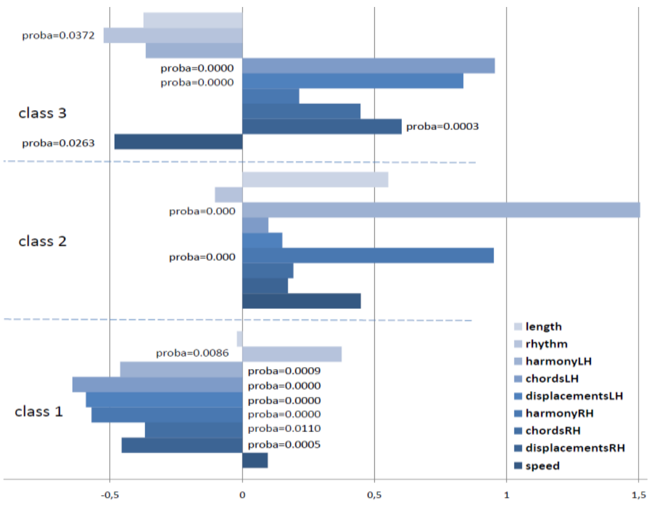
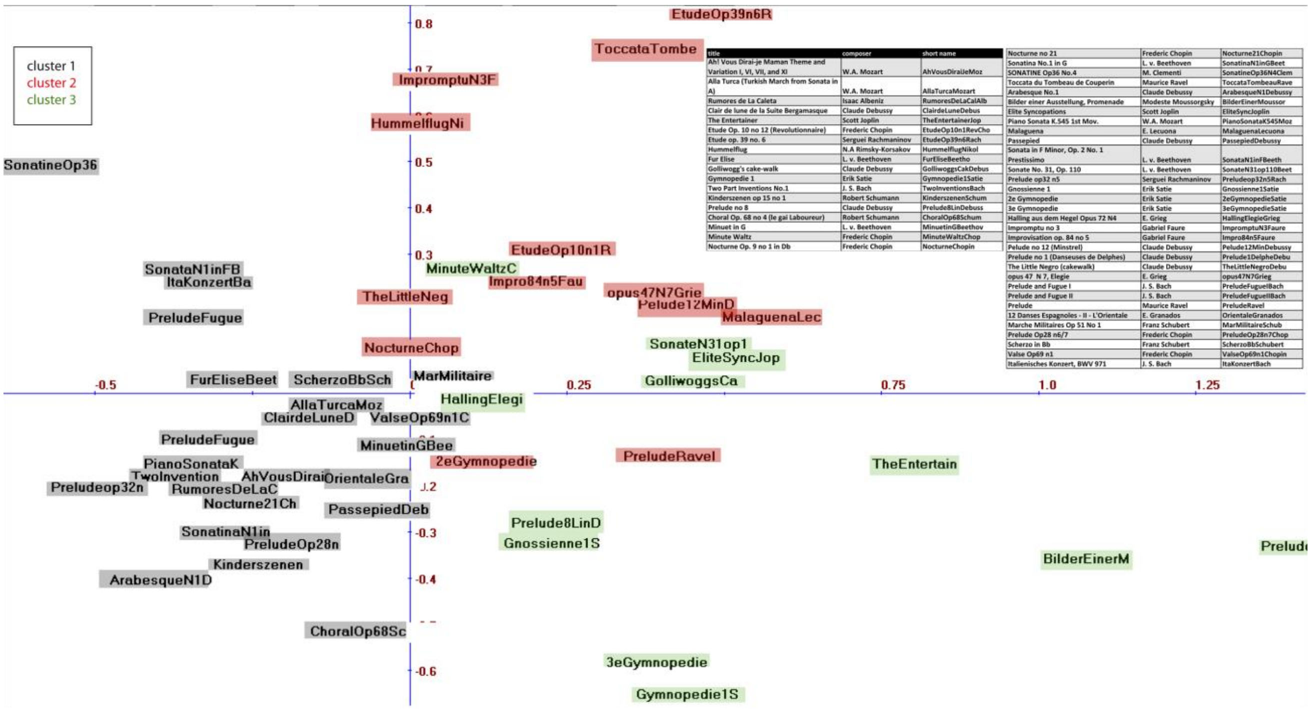


Figure 4. Student test (means comparison)



title	composer	short name
Ah! Vous Dirai-je Maman Theme and Variation I, VI, VII, and XI	W.A. Mozart	AhVousDiraiJeMoz
Alla Turca (Turkish March from Sonata in A)	W.A. Mozart	AllaTurcaMozart
Rumores de La Caleta	Isaac Albeniz	RumoresDeLaCalAlb
Clair de lune de la Suite Bergamasque	Claude Debussy	ClairdeLuneDebus
The Entertainer	Scott Joplin	TheEntertainerJop
Etude Op. 10 no 12 (Revolutionnaire)	Frederic Chopin	EtudeOp10n1RevCho
Etude op. 39 no. 6	Serguei Rachmaninov	EtudeOp39n6Rach
Hummelflug	N.A Rimsky-Korsakov	HummelflugNikol
Fur Elise	L. v. Beethoven	FurEliseBeetho
Golliwogg's cake-walk	Claude Debussy	GolliwoggsCakDebus
Gymnopedie 1	Erik Satie	Gymnopedie1Satie
Two Part Inventions No.1	J. S. Bach	TwoInventionsBach
Kinderszenen op 15 no 1	Robert Schumann	KinderszenenSchum
Prelude no 8	Claude Debussy	Prelude8LinDebuss
Choral Op. 68 no 4 (le gai Laboureur)	Robert Schumann	ChoralOp68Schum
Minuet in G	L. v. Beethoven	MinuetinGBeethov
Minute Waltz	Frederic Chopin	MinuteWaltzChop
Nocturne Op. 9 no 1 in Db	Frederic Chopin	NocturneChopin
Nocturne no 21	Frederic Chopin	Nocturne21Chopin
Sonatina No.1 in G	L. v. Beethoven	SonatinaN1inGBeet
SONATINE Op36 No.4	M. Clementi	SonatineOp36N4Clem
Toccata du Tombeau de Couperin	Maurice Ravel	ToccataTombeauRave
Arabesque No.1	Claude Debussy	ArabesqueN1Debussy
Bilder einer Ausstellung, Promenade	Modeste Moussorgsky	BilderEinerMoussor
Elite Syncopations	Scott Joplin	EliteSyncJoplin
Piano Sonata K.545 1st Mov.	W.A. Mozart	PianoSonataK545Moz
Malaguena	E. Lecuona	MalaguenaLecuona
Passepied	Claude Debussy	PassepiedDebussy
Sonata in F Minor, Op. 2 No. 1 Prestissimo	L. v. Beethoven	SonataN1inFBeeth
Sonate No. 31, Op. 110	L. v. Beethoven	SonateN31op110Beet
Prelude op32 n5	Serguei Rachmaninov	Preludeop32n5Rach
Gnossienne 1	Erik Satie	Gnossienne1Satie
2e Gymnopedie	Erik Satie	2eGymnopedieSatie
3e Gymnopedie	Erik Satie	3eGymnopedieSatie
Halling aus dem Hegel Opus 72 N4	E. Grieg	HallingElegieGrieg
Impromptu no 3	Gabriel Faure	ImpromptuN3Faure
Improvisation op. 84 no 5	Gabriel Faure	Impro84n5Faure
Pelude no 12 (Minstrel)	Claude Debussy	Pelude12MinDebussy
Prelude no 1 (Danseuses de Delphes)	Claude Debussy	Prelude1DelpheDebu
The Little Negro (cakewalk)	Claude Debussy	TheLittleNegroDebu
opus 47 N 7, Elegie	E. Grieg	opus47N7Grieg
Prelude and Fugue I	J. S. Bach	PreludeFugueIBach
Prelude and Fugue II	J. S. Bach	PreludeFugueIIBach
Prelude	Maurice Ravel	PreludeRavel
12 Danses Espagnoles - II - L'Orientale	E. Granados	OrientaleGranados
Marche Militaires Op 51 No 1	Franz Schubert	MarMilitaireSchub
Prelude Op28 n6/7	Frederic Chopin	PreludeOp28n7Chop
Scherzo in Bb	Franz Schubert	ScherzoBbSchubert
Valse Op69 n1	Frederic Chopin	ValseOp69n1Chopin
Italienisches Konzert, BWV 971	J. S. Bach	ItaKonzertBach

Figure 5. Individuals projection on the PCA first two axes and corpus details

mic patterns which constitute interesting educational material (e.g. *1st Arabesque* by Debussy). Class 2 contains advanced to virtuoso works (Chopin's *Etude*, Ravel's *Toccata*), featuring a vivid tempo, large and numerous displacements on the keyboard, a complex harmony and many chords. We also note some borderline individuals (*The Little Negro* by Debussy, or the 2nd *Gymnopédie* by Satie), which could be considered as beginner pieces but still present uncommon harmonic and rhythmic structures, thus being hard to classify objectively. Class 3 seems to regroup pieces featuring a left hand playing a "bass+chord" accompaniment (ragtime, waltz, cakewalk). The level of most pieces is intermediate. Indeed, the Student test indicates that despite the high ratio of displacements and chords, the low tempo and the simplicity of the harmonies compensate for it. As such, this particular class is also representative of specific musical genres. This clustering serves as a complement to the "bounds" approach used in Score Analyzer.

5. SCORE ANALYZER PROTOTYPE

The criteria presented in the previous sections have been implemented in a Web application called Score Analyzer [1] (SA). This module is integrated to the @-MUSE platform as a Web service in order to automatically evaluate a piece level and identify its difficult parts. The SA engine takes any well-formed MusicXML file as input and parses it to extract knowledge exploitable from a performer point of view. Following the scheme we detailed previously (Figure 1), the context of the piece is briefly analyzed (title, composer) and a few statistics are displayed. Then, main parts of the piece are identified, and lastly, difficulty estimations are given for each criterion, using a mark from 1 (beginner/easy) to 4 (virtuoso). A mean is also calculated to give a global appreciation of the piece difficulty. This allows a better readability of the outputs for musicians. For each criterion, bounds were defined with the help of teachers: for instance, a chord ratio under 10% corresponds to the mark 1, while a displacement ratio above 20% corresponds to a 4. These bounds determination was transparent for teachers, as they were simply asked to rate each criteria from 1 to 4 on a training corpus. The given marks were then correlated to the ratio extracted on each piece, in order to calibrate average bounds corresponding to the difficulty levels felt by musicians. Thus, we notice that most of the criteria do not have a linear distribution, which constitutes a pianistic reality. The synchronization between both hands is also taken into account. For instance, if each hand obtains a mark of 2 for the displacements criterion, then the global difficulty mark for this criterion will be 3, as synchronizing both hands will create an additional difficulty.

As such, we define this method as "semi-objective". Indeed, score level estimation can never be a totally objective task: players will judge a piece differently according to their taste, level or background. Therefore, we use two distinct methods to validate SA estimations. The first one consists in confronting it to the clustering obtained through the PCA described in the previous part. This is the "objective" validation. The second one simply consists in confronting SA results to pianists estimations ("subjective" validation). To facilitate the comparisons, we merged advanced and virtuosi pieces into the same class within SA. The contingencies table (Table 2) allows to better visualize the differences between the PCA and

[1] http://e-piano.univ-reunion.fr/tests/ScoreAnalyser/readScore.php, visited on the 05/06/2012, beta version.

SA marks \ PCA marks	1	2	3
1	TwoInventionsBach KinderszenenSchum ChoralOp68Schum SonatinaN1inGBeet PianoSonataK545Moz		
2	AhVousDiraiJeMoz AllaTurcaMozart RumoresDeLaCalAlb ClairdeLuneDebus FurEliseBeetho MinuetinGBeethov Nocturne21Chopin SonatineOp36N4Clem ArabesqueN1Debussy PassepiedDebussy Preludeop32n5Rach PreludeFugueIBach PreludeFugueIIBach OrientaleGranados PreludeOp28n7Chop ScherzoBbSchubert ValseOp69n1Chopin MarMilitaireSchub ItaKonzertBach	TheEntertainerJop GolliwoggsCakDebus Gymnopedie1Satie BilderEinerMoussor EliteSyncJoplin Gnossienne1Satie 3eGymnopedieSatie HallingElegieGrieg Prelude1DelpheDebu	HummelflugNikol Pelude12MinDebussy TheLittleNegroDebu opus47N7Grieg 2eGymnopedieSatie PreludeRavel
3	SonataN1inFBeeth	Prelude8LinDebuss MinuteWaltzChop SonateN31op110Beet	EtudeOp10n1RevCho EtudeOp39n6Rach NocturneChopin ToccataTombeauRave MalaguenaLecuona ImpromptuN3Faure Impro84n5Faure

Table 2. Contingencies table between SA and PCA marks

SA marks \ teachers marks	1	2	3
1	TwoInventionsBach KinderszenenSchum ChoralOp68Schum SonatinaN1inGBeet	PianoSonataK545Moz	
2	2eGymnopedieSatie MinuetinGBeethov 3eGymnopedieSatie PreludeFugueIBach PreludeOp28n7Chop ValseOp69n1Chopin TheLittleNegroDebu	TheEntertainerJop GolliwoggsCakDebus Gymnopedie1Satie BilderEinerMoussor EliteSyncJoplin Gnossienne1Satie HallingElegieGrieg Prelude1DelpheDebu AhVousDiraiJeMoz AllaTurcaMozart RumoresDeLaCalAlb FurEliseBeetho Nocturne21Chopin SonatineOp36N4Clem ArabesqueN1Debussy PreludeFugueIIBach OrientaleGranados ScherzoBbSchubert MarMilitaireSchub opus47N7Grieg PreludeRavel ItaKonzertBach	HummelflugNikol Pelude12MinDebussy ClairdeLuneDebus PassepiedDebussy Preludeop32n5Rach
3		Prelude8LinDebuss MinuteWaltzChop SonateN31op110Beet	EtudeOp10n1RevCho EtudeOp39n6Rach NocturneChopin ToccataTombeauRave MalaguenaLecuona ImpromptuN3Faure SonataN1inFBeeth

Table 3. Contingencies table between SA and teachers marks

Score Analyzer's results. While they seem numerous, only one is a major disagreement (3/1 marks on Beethoven Sonata in F). The other distinctions, especially the intermediate/beginners ones, may be due to the fact that humans balance criteria whereas the PCA considers each of them of equal importance. Therefore, we noticed that for pianists, an increase of the displacement ratio raises the piece level much faster than other criteria. Moreover, as stated in the previous part, the clustering given by the PCA is also affected by the musical genre of the piece. Humans do not tend to be affected by this metadata, even if some genres are naturally associated with higher levels (i.e. impressionist or contemporary music).

For the "subjective" evaluation, we asked three piano teachers to estimate the difficulty level of each piece by attributing it a mark between 1 and 3. No criteria were imposed. When opinions differ, the final mark is picked according to the majority. The results given in Table 3 show a better correspondence between SA estimations and human ones, which reinforces the "bounds" method defined previously. The main difference consists in underestimations from SA, especially on advanced pieces. Indeed, pianists also take expression and musicality difficulties into account, while our system only consider technical difficulties. Therefore, this study leads us to pursue our work by expanding the set of criteria to improve our estimations.

6. CONCLUSION

In this paper, we proposed an automatic Score Analyzer to determine the difficulty level of piano pieces. This prototype is based on seven criteria characterizing technical features of a piano piece: playing speed, fingerings, hands displacements, polyphony, harmony, rhythm and length. We thus proposed methods to extract these criteria from a MusicXML scores, and realized a PCA to validate them. This analysis permitted to establish three classes among a corpus of fifty selected piano pieces. These classes were then confronted to Score Analyzer estimations, which are tuned according to piano teachers expertise.

Improvements on this work include the integration of fingering related difficulties, but also the adaptation to students levels. Indeed, the sense of difficulty within a musical work is mostly dependent from the musician's background. We thus imagine a weighting system to personalize our analysis. We also intend to implement local analysis (by measures) in order to identify specific difficult parts. The criteria decomposition would then allow to extract the main cause of the difficulty and thus link it to an annotation created on the @-MUSE platform. Other perspectives include integration of "expressive" criteria (emotions, nuances, rubato, attacks), as well as adaptations and tests on scores for different instruments.

7. REFERENCES

[1] V. Sébastien, D. Sébastien, N. Conruyt, "Constituting a Musical Sign Base through Score Analysis and Annotation", *International Journal On Advances in Networks and Services*, No 3&4, pp. 386-398, 2011.

[2] M. A. Winget: "Annotations on musical scores by performing musicians: Collaborative models, interactive methods, and music digital library tool development", *Journal of the American Society for Information Science and Technology*, 2008.

[3] G. Castan, M. Good, and P. Roland: "Extensible Markup Language (XML) for Music Applications: An Introduction", *The Virtual Score: Representation, Retrieval, Restoration*, MIT Press, pp. 95-102, 2001.

[4] M. W. Camp: *Teaching piano: the synthesis of mind, ear and body*, Alfred Music Publishing, 1992.

[5] O. Lartillot: "Une analyse musicale automatique suivant une heuristique perceptive", *3ème Conférence Internationale Francophone sur l'Extraction et la Gestion des Connaissances*, EGC 03, Lyon, 2003.

[6] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson: "The Music Ontology", *Proceedings of the International Conference on Music Information Retrieval*, ISMIR, 2007.

[7] P. Rigaux: "Neuma Ontology Specification", *Project Neuma Report*, Lamsade-CNRS, ANR-08, 2008.

[8] C.-C. Lin: "*An Intelligent Virtual Piano Tutor*", National Chung Cheng University 2006.

[9] A. Al Kasimi, E. Nichols, and C. Raphael, "A simple algorithm for automatic generation of polyphonic piano fingerings": *8th International Conference on Music Information Retrieval*, Vienna, 2007.

[10] R. Parncutt, J. A. Sloboda, M. Raekallio, E. F. Clarke, and P. Desain: "An Ergonomic Model of Keyboard Fingering for Melodic Fragments", *Music Perception: An Interdisciplinary Journal*, Vol. 14, No. 4, 1997, pp. 341-382.

[11] J. H. Ward: "Hierarchical Grouping to Optimize an Objective Function", *Journal of the American Statistical Association*, No. 48, pp. 236–244, 1963.

DIGITAL DOCUMENT IMAGE RETRIEVAL USING OPTICAL MUSIC RECOGNITION

Andrew Hankinson **John Ashley Burgoyne** **Gabriel Vigliensoni**
Alastair Porter **Jessica Thompson** **Wendy Liu**
Remi Chiu **Ichiro Fujinaga**

Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
Schulich School of Music, McGill University, Montréal, QC, Canada
andrew.hankinson@mail.mcgill.ca

ABSTRACT

Optical music recognition (OMR) and optical character recognition (OCR) have traditionally been used for document transcription—that is, extracting text or symbolic music from page images for use in an editor while discarding all spatial relationships between the transcribed notation and the original image. In this paper we discuss how OCR has shifted fundamentally from a transcription tool to an indexing tool for document image collections resulting from large digitization efforts. OMR tools and procedures, in contrast, are still focused on small-scale modes of operation. We argue that a shift in OMR development towards document image indexing would present new opportunities for searching, browsing, and analyzing large musical document collections. We present a prototype system we built to evaluate the tools and to develop practices needed to process print and manuscript sources.

1. INTRODUCTION

Optical character recognition (OCR) is used to convert digital images of text into computer-manipulable representations, which in turn are used to store and index the content of books, newspapers, scholarly journals, and magazines. OCR has been integrated into many large-scale print digitization initiatives, and is currently being used to provide users with the unprecedented ability to search and retrieve millions of sources instantly—a task that previously would have taken many lifetimes.

While OCR is opening up new avenues for users to search, discover, and analyse large quantities of textual material, the content of printed music documents is still trapped almost entirely in the physical world. Even collections that have been digitized and placed online are still merely pictures of pages, with no means of extracting their contents. Unfortunately, current optical music recognition (OMR) software packages have not been designed to process large volumes of page images efficiently. Rather, they are still designed for small-scale, single user transcription. In order to provide OMR tools comparable to the OCR tools now available there is a need for developing tools, technologies, and best practices for recognizing, indexing, searching, and retrieving large amounts of digital page images.

We believe that large-scale OMR projects are critical to research in MIR and also computational musicology. The vast majority of human musical output from the past 1000 years does not exist in any kind of manipulable digital format but rather lies within the enormous collections of printed music and music manuscripts sitting on library shelves across the globe. Relying on humans to transcribe and share this music is expensive and unsatisfactory for many purposes. Many human transcriptions, especially of early music, involve a substantial amount of personal interpretation on the part of the transcriber, and end users may not entirely agree with a particular interpretation. Moreover, many musicologists are interested in studying how the extra-musical content on a page informs the musical content, and these scholars need to be able to access the orginal page images in order to draw concusion. By combining automatic transcription with retrieval of the original page image, we can build systems that permit users to retrieve documents by content but then rely on the original image for study, relaxing the need for exact or "objective" transcription. Analogous to the situation with OCR, even lower-accuracy OMR would be sufficient to direct a user toward documents of interest, an order of magnitude more quickly than the current situation whereby researchers must visit library shelves and manually study every page.

In this paper we argue that there are a number of OMR technologies that must be in place to enable mass music recognition and retrieval projects. We will discuss the development of similar OCR technologies built to support large-scale text document image indexing and retrieval systems and compare that to the existing OMR technologies. As part of this discussion, we present a prototype project developed as part of the Single Interface for Music Score Searching and Analysis (SIMSSA) initiative. This project includes the development of an

OMR workflow system, a notation encoding format for storing the results of our OMR system, and the development of search and retrieval tools. We discuss our findings from this prototype, and present some of opportunities for future work.

2. PREVIOUS WORK

Both OCR and OMR were initially conceived as transcription technologies; that is, a digital page image was supplied, the textual or musical content was extracted, and then the original image was discarded. The result was the transcribed content of the page in a format suitable for further editing in a word processor or notation editor. This method left no direct correspondence between the page content and its location on the original image in the output format.

When computing advanced enough to display images, OCR began to be used as a means of navigating and retrieving document images based on their textual content. Although the recognition process remained the same, OCR file formats began preserving correspondence between document content—words, paragraphs, columns, graphics—and the original page image. When combined with a search engine, this provided users with the ability to navigate to the exact occurence of a search word or phrase. This was the emergence of the shift in OCR from a technology that "merely" transcribed text, to one that permitted navigation through large numbers of digital document images. Many researchers at this conference have probably experienced the benefits of these systems for locating journal articles or conferences papers in systems like JSTOR.

In the next section we will look at a number of technologies in OCR that have supported its emergence as a document image navigation technology. We will then consider a few of the projects that have tried to do similar things with printed music documents.

2.1. Document Image Formats

One of the most crucial components of a document image indexing system is the ability to correlate recognized objects with their location on the page. In this section we discuss a few formats developed for textual document indexing.

Nagy [1] describes an early system for using OCR on document images from pages of technical journals, allowing users to search and retrieve image segments containing their query terms. He notes that a "major strength of the approach is the preservation of the original layout of the documents, which not only augments reading comprehension but also often conveys indispensable information on its own." This seems to be the first such mention of retrieving document page images using OCR analysis.

Stehno et al. [2] describe the METS/ALTO format (Metadata Encoding and Transmission Standard / Analyzed Layout and Text) for mapping layout structures and text passages from book pages. This is currently the most widely used standard for preserving text layout derived from OCR systems.

Several other formats deserve a brief mention as well. The hOCR format is supported by the OCRopus and Tesseract software, used by the Google Book digitization project [3]. Portions of the Internet Archive digitization project use the DjVuXML format which contains the words and coordinates of a "hidden" layer [4]. The HathiTrust Project uses their own XML format.

The PAGE format [5] is designed to encode data for evaluating OCR document analysis. It is different than most other formats presented here, in that it encodes ground-truth data for evaluating document layout. This includes encoding features like reading order (the order in which columns or segments of a page are read by a human).

In all of these cases, document layout and text content are maintained in an integrated document format, allowing the OCR system to store image coordinates for each document element (words, lines, paragraphs, etc.) recognized by the analysis software. In the next section we will briefly discuss how this data may be used in an indexing system to retrieve page images.

2.2. Document Image Retrieval

Indexing for page image retrieval is more complex than indexing for simple text retrieval. An index must be built containing all the words that have been recognized from the images, but these must be further correlated with their page and location on the page image. Retrieving page images requires indexing and storing key words and their positions in the document images where they occur. This gives users the ability to enter a search query (a word or phrase) and retrieve the pages where the result of their query can be found, highlighting their exact positions on the page image.

The HathiTrust has constructed a correlated text and image index for their collection. They incorporate the graphical locations for each recognized word into their index [6]. They note that the coordinate positions for every word accounts for 85% of the index size of a particular book. This means that for their required goal of 10 million books, their expected index size is two terabytes of which most of the information is OCR coordinate data.

2.3. Large-Scale OCR

To handle large numbers of documents, OCR applications have moved away from standalone desktop applications to server-based solutions. This allows distributed task separation, whereby multiple teams can simultaneously work on digitization, recognition, correction, and publishing without being bound to a single workstation. Many tasks can be partially or fully automated, requiring human intervention only as a quality-control measure.

In the commercial sector there are a number of large-scale solutions. Perhaps the most successful example of this is the Abbyy Recognition Server[1], a centralized OCR workflow management system. Documents are ingested by digitization, automatically recognized, then verified, corrected, and further indexed by humans sitting at multiple workstations. For open-source software, there are a number of command-line tools that can be chained together to form an automated OCR system. The OCRopus and Tesseract systems [7], developed by Google for their book search projects, contains a number of tools for creating highly customizable OCR systems. The recently-completed IMPACT (Improving Access to Text) project [8], a €16.5M research project, focused on building new OCR tools and best practices for libraries and archives. They have created a system that allows multiple image processing, OCR, and results evaluation tools to be chained together to form an *ad hoc* recognition system.

Since recognition systems will never be perfect, techniques that enable humans to correct OCR and ensure that recognition errors will not create problems for retrieval. Correction, however, can be very time and labour intensive. Some unique solutions have been developed to help offset the costs of this task. The Australian Newspaper Project [9] has created a "crowd-sourced" correction system, where more than 9,000 volunteers have now corrected more than 12.5 million lines of text, with more corrections added all the time. The reCAPTCHA project [10] has produced over 5 billion human-corrected OCR words by presenting the correction task as a spam-fighting challenge to prove that the corrector is a human and not an automated system. Tools for distributed proofreading and correction allow for a constantly-improving search and retrieval system, and also for the collection of pixel-aligned ground-truth data to further improve the accuracy of OCR systems.

2.4. Music Document Image Retrieval

The purpose of providing a review of tools and techniques employed by text search projects is to compare and contrast it to similar work done for OMR. Unlike the large text initiatives, such as the HathiTrust, Google Books, and Internet Archive projects, we are unaware of any publicly available databases that allow users to retrieve page images from printed books based on automatic transcription of the page contents using OMR. There are, however, a few projects that have developed some functionality worth mentioning here.

The PROBADO Music Project [11],[12],[13] [14] is perhaps the largest and longest-running project incorporating large-scale OMR for use in search systems. This project seeks to provide a unified interface for retrieving symbolic and audio representations of music pieces. As of October 2010, their dataset consisted of 50,000 pages from 292 books. The content of their dataset is music printed in common Western notation in a variety of genres and instrumentations, including opera, symphonic works, and Classical and Romantic piano music.

The primary goal of the PROBADO project is to allow symbolic, image, and audio synchronization, providing users with the ability to navigate a score and hear the audio, or navigate the audio and jump to its corresponding position in the score. Their technique generates MIDI files from OMR, rendered to an audio representation, and then aligned with different audio recordings of the work. The audio is then aligned at the measure level with a score image, allowing the system to highlight the current measure as the audio plays.

The PROBADO project uses the SharpEye ASCII file format for storing the notation-to-pixel coordinate information. This format is documented at [15], but is only supported by the SharpEye OMR system. Similarly, Hankinson et al. [16] propose the use of the Music Encoding Initiative (MEI) format for maintaining notation-to-pixel correspondence.

Bainbridge et al. [17] describe a Greenstone plug-in utilizing the CANTOR OMR system. Their system transcribes the notated music and makes it available for searching. In their system they make the original page image available for viewing. Unfortunately, development on this system seems to have stopped, and no working version of their retrieval system can be found.

3. LU PROTOTYPE

We have created a prototype system providing notation-based retrieval of document images in a web application. The *Liber Usualis* (LU) [18] is a liturgical service book produced by the Roman Catholic church and an important source for Gregorian chant. It uses square-note neume notation derived from the earlier Franconian style but modernized by the monks at Solesmes, France in the late 19th Century. There has been very little work on OMR for this type of notation, with the exception of [19]. We performed OMR and OCR on all 2,340 page images in this book, maintaining notation and image correspondence. We then developed a web application that allows basic query input based on *n*-gram indexing of the notation content, highlighting the locations of results *in situ* on the page image. In this section we will briefly review the components of this prototype. A full overview may be found in [20], and some details of the specific technologies we developed may be found in [21], [22]. We have made a public demo of our retrieval system available online.[2]

3.1. OMR Workflow

Our workflow is illustrated in Figure 1. We begin with a page image, captured by either scanning or photographing a book. In the case of the LU, we began with a complete PDF file of pre-scanned images. We then sent each page through the workflow.

[1] http://www.abbyy.com/recognition_server/

[2] http://ddmal.music.mcgill.ca/liber

The first step was automatic page layout analysis, which we used to separate the textual and musical areas of the page. We performed the layout analysis using a version of Aruspix, an application originally designed for OMR of Renaissance printed music, which we modified to operate on the neume notation in the LU. The page images were automatically scaled and straightened. Aruspix is capable of automatically locating and identifying various graphical page elements, providing the ability to distinguish between musical and textual content: musical staves, lyrics, ornate letters, lyrics, title elements, or other text. The different page elements are given different pixel colours after the automated analysis, creating separable "layers" that contain either exclusively music or exclusively text elements. The automated analysis saves a considerable amount of time and labour, although any mis-classified page elements do need to be corrected manually. Figure 2 shows the correction interface in Aruspix, with a pop-up context menu allowing the operator to select an area of the image and re-classify it as a different type of page. The LU does not have a particularly complex layout, and most pages took between 30 and 130 seconds to correct (median 77 s).

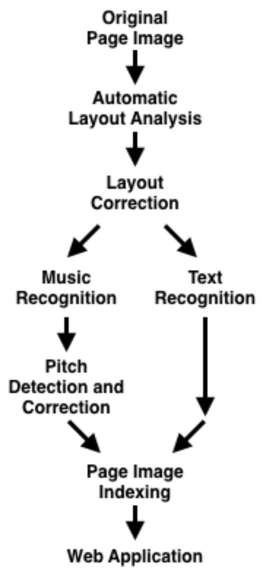


Figure 1: OMR Workflow for the Liber Usualis

Following the layout analysis, the text layers were sent through an OCR workflow stage, which allowed the text to be searched and linked to the specific regions of each page on which the text occurred. We used OCRopus, a third-party open-source OCR engine, to perform the text analysis. Minimal work was done to correct the OCR output, however. Post-OCR, we used a simple edit distance to auto-correct recognized text from a dictionary of liturgical Latin words. Lyrics that were broken into syllables were automatically re-joined at a hyphenated break to form complete words. No further human correction or processing was performed. The resulting text has a large number of errors, but it was sufficient for a "proof of concept." The output of this stage was fully OCRed text lines with the bounding-box co-ordinates for the full line.

The music layer was sent through an OMR workflow. Using the Gamera toolkit [23], we first removed the staff lines from each musical layer. Removing staff lines not only facilitated OMR but also allowed us to compute precise bounding boxes for each musical element. These bounding boxes are essential information for retrieval systems that wish to show the results of musical queries on the page. Gamera uses adaptive k-nearest-neighbour classifiers for musical symbols, improving its classification performance by using the information from the errors corrected on previous pages. It took between 7 and 16 minutes to correct the errors on most pages (median 11 min). Gamera keeps track of the location of staff lines when it removes them, but it classifies on the shapes, not the pitches of musical symbols. The last step of the OMR workflow was a customized algorithm for combining information about the location of the staff lines, the bounding box for each shape, and the shape of the musical symbol itself to add pitch information to every symbol.

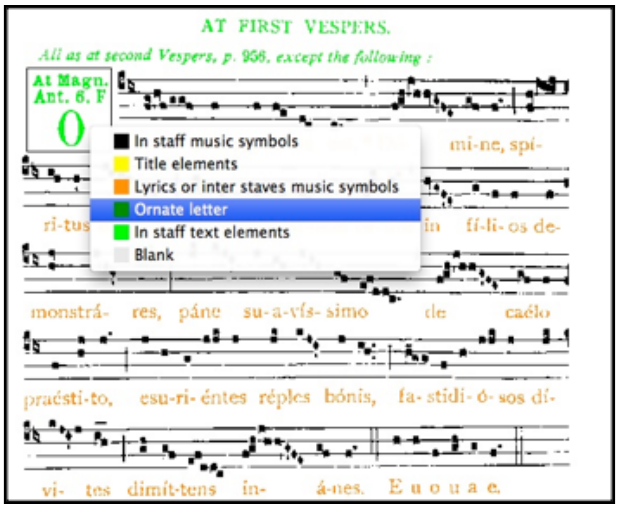

AT FIRST VESPERS.
All as at second Vespers, p. 956, except the following :
At Magn. Ant. 6. F
O
In staff music symbols
Title elements
Lyrics or inter staves music symbols
Ornate letter
In staff text elements
Blank
mi-ne, spí-
ri-tus
in fí-li-os de-
monstrá- res, pánc su-a-vís-simo de caélo
praésti-to, esu-ri-éntes réples bónis, fa-stidi-ó-sos dí-
vi- tes dimít-tens in- á-nes. E u o u a e.

Figure 2: Layout Analysis and Correction in Aruspix

After extracting the pitch information and bounding boxes for every musical symbol and text line on every page, the final step of the workflow was to store the recognized page content into a standard file format. We chose the MEI format for a number of reasons. MEI is an XML-based notation encoding scheme, but unlike most notation formats it can be extended to support many different types of notation [24]. This was particularly valuable in our case, since the neume notation used by the LU is a revival of much older plainchant notation with additional symbols added to indicate breathing marks or articulations. With MEI we were able to develop a custom encoding scheme to support neume notation markup, while maintaining the broader document framework and markup structure of MEI.

To support document image indexing and retrieval, MEI provides the ability to define image zones—pixel-based bounding boxes that store co-ordinates on a reference image—and correlate them with the recognized musical and textual elements. It is important to note here that this is done while still preserving the musical structure; that is, the notation maintains the melodic and symbolic structures that are expected from a notation encoding scheme. Each musical and textual element is then correlated with a defined zone using the MEI `@facs` attribute. The end result is an XML hierarchy containing both the musical, textual, and graphical information correlated and ready to be indexed by a search engine to facilitate image retrieval.

3.2. Indexing, Searching and Retrieval

One of the most important musicological uses for the LU is as a compendium of important chant melodies that appear in some form across a wide variety of ancient musical manuscripts and many later compositions. Despite its importance, there is no thematic catalogue of its musical content, and at more than 2,000 pages, it can be very time-consuming for researchers or musicians to find what they are looking for. To facilitate retrieval we created an efficient index for retrieving musical fragments from across the LU while maintaining information the location of these melodic fragments on their respective pages. Following Stephen Downie [25], we generated indexes on *n*-grams, for *n* from 2 to 10, on the following five features:

- pitches, a concatenated string of all of the pitch names;
- intervals, represented as the directed melodic intervals between successive pitches in musical steps;
- semitones, represented as the directed melodic intervals between successive pitches in semitones;
- contour, represented as sequences of "up," "down," or "repeated," i.e., the direction of the intervals; and
- neumes, represented by their component neume names, i.e., if an *n*-gram was represented by the neume sequence "punctum clivis clivis."

We also indexed the textual content of each page, including the co-ordinate information for each recognized line.

For each *n*-gram, the index also included the page on which an item appeared and its aggregate bounding box. Combined, the indexes include approximately three million unique *n*-grams, which we store in an Apache Solr instance[3].

Users do not interact directly with the Solr instance. We built a web application based on the open-source Diva.js viewer [26] to present the original document images and highlight the results of queries on these images. The web application uses the indexed *n*-grams to provide a number of search capabilities:

- strict or pitch-invariant sequences, where the user can type in a sequence of pitch names and it will either search for the literal string of pitches, or use the semitone index to search for possible matches that use the same intervallic content but contain different pitches;
- contour, containing the rough shape of the target melody, e.g., "dduurr";
- intervals, containing the specific shape of the target melody, e.g., "d2 d2 u2 u3 r r";
- neumes, where the user specifies a sequence of neume names, e.g., "punctum clivis clivis"; and
- text, for retrieving pages based on lyrical or textual content.

Using the co-ordinate data from the OMR and stored in the MEI, the results from a users' query for a pitch sequence will bring the user to the page where their result can be found, with the bounding box around the search result highlighting their query. Figure 3 shows a screenshot from our web application with the result of the pitch-sequence query "*edcdeee*" highlighted on the original page image of the LU.

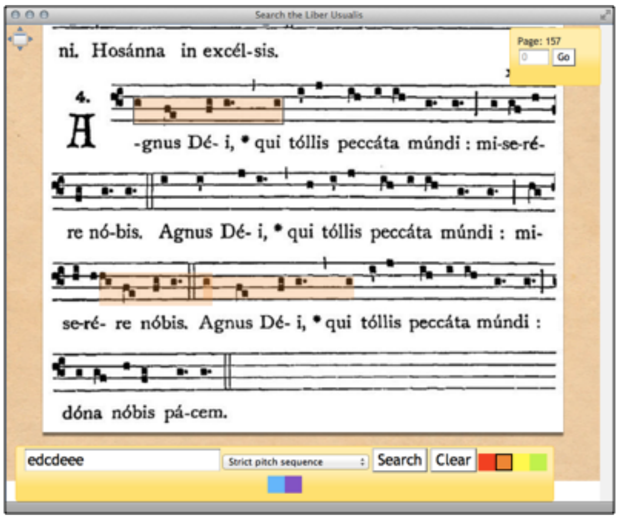

Search the Liber Usualis
Page: 157
Go
ni. Hosánna in excél-sis.
4.
A
-gnus Dé- i, * qui tóllis peccáta múndi : mi-se-ré-
re nó-bis. Agnus Dé- i, * qui tóllis peccáta múndi : mi-
se-ré- re nóbis. Agnus Dé- i, * qui tóllis peccáta múndi :
dóna nóbis pá-cem.
edcdeee
Strict pitch sequence
Search
Clear

Figure 3: The Liber Usualis Interface

4. DISCUSSION AND CONCLUSION

The SIMSSA initiative is a long-term research program for investigating and supporting large-scale OMR and document image retrieval for all types of music documents, from early manuscripts through to modern music notation. The work presented in this paper is an initial attempt at building systems that support processing and retrieval at a scale and quality level that, to date, has not been achieved for musical documents. In this paper we have identified technologies and techniques that have been developed to support OCR for transcribing and navigating large numbers of document images, and have demonstrated a prototype system we have developed as a platform for further research into how to shift OMR from small-scale transcription to large-scale document image navigation and retrieval.

There are many open research questions arising for this work that need further investigation. One of the most critical is the need for user studies and experimental interfaces for musical document retrieval. Most current symbolic search systems are built around query interfaces that provide limited access to the underlying

[3] http://lucene.apache.org/solr/

notated music, typically restricted simple pitch or rhythm queries. More robust systems must be built to support more complex analysis-retrieval tasks, such as investigating the occurrence of specific cadential patterns or movement between multiple simultaneous voices. More complex query and analysis systems will in turn require more sophisticated user interfaces, which will need a deeper understanding of what what types of questions musicologists, theorists, and performers would like to see supported in a retrieval system.

In our opinion, OMR must move beyond desktop applications and simple transcription. New modes of operation can and should be developed, including server- and browser-based recognition, distributed proofreading and correction, networked recognition systems, and expanded research on recognition evaluation by building curated ground-truth datasets covering different styles and types of music notation. Our prototype system represents a first step towards investigating many of these topics, and through the SIMSSA project we hope to spur further research to make the world's music collections available to all.

5. ACKNOWLEDGEMENTS

This work would not have been possible without the efforts of a number of people. The authors would like to thank Julie Cumming, Mahtab Ghamsari, Jamie Klassen, Saining Li, Mikaela Miller, Laura Osterlund, Laurent Pugin, and Caylin Smith for their contributions. The SIMSSA project is funded by the Social Sciences and Humanities Research Council of Canada. Further funding was provided by the Centre for Interdisciplinary Research in Music Media and Technology and the Canadian Foundation for Innovation.

6. REFERENCES

[1] Nagy, G. 1992. *Towards a structured-document-image utility*. In *Structured Document Image Analysis*, eds H. Baird, H. Bunke, and K. Yamamoto. Berlin: Springer.

[2] Stehno, B., A. Egger, and G. Retti. 2003. METAe—Automated encoding of digitized texts. *Literary and Linguistic Computing* 18 (1): 77–88.

[3] Breuel, T., and U. Kaiserslautern. 2007. The hOCR microformat for OCR workflow and results. In *Proc. Int'l. Conf. on Document Analysis and Recognition*, 1063–7.

[4] Kumar, R. 2008. Bulk access to OCR for 1 million books. http://blog.openlibrary.org/2008/11/24/bulk-access-to-ocr-for-1-million-books/.

[5] Pletschacher, S., and A. Antonacopoulos. 2010. The PAGE format framework. In *Proc. Int'l. Conf. on Pattern Recognition,* Istanbul, TR. 257–60.

[6] Farber, P. 2009. Large-scale full-text indexing with Solr. http://www.hathitrust.org/blogs/large-scale-search/large-scale-full-text-indexing-solr.

[7] Breuel, T.. 2009. Recent progress on the OCRopus OCR system. In *Int'l. Workshop on Multilingual OCR*, Barcelona, ES. 1–10.

[8] Balk, H., and L. Ploeger. 2009. Impact: Working together to address the challenges involving mass digitization of historical printed text. *OCLC Systems and Services* 25 (4): 233–48.

[9] Holley, R. 2009. Many hands make light work: Public collaborative OCR text correction in Australian historic newspapers. *National Library of Australia Staff Papers*.

[10] Von Ahn, L., B. Maurer, C. Mcmillen, D. Abraham, and M. Blum. 2008. reCaptcha: Human-based character recognition via web security measures. *Science* 321 (5895): 1465–8.

[11] Diet, J., and F Kurth. 2007. The PROBADO music repository at the bavarian state library. In *Proc. Int'l. Conf. Music Information Retrieval,* Vienna, AT.

[12] Kurth, F., M. Müller, C. Fremerey, Y. Chang, and M. Clausen. 2007. Automated synchronization of scanned sheet music with audio recordings. In *Proc. Int'l. Conf. on Music Information Retrieval* , Vienna, AT. 261–6.

[13] Fremerey, C. 2010. Automatic organization of digital music documents: Sheet music and audio. PhD diss., Mathematics and Natural Sciences, University of Bonn, Bonn, DE.

[14] Damm, D., F. Kurth, C. Fremerey, and M. Clausen. 2009. A concept for using combined multimodal queries in digital music libraries. *Research and Advanced Technology for Digital Libraries* 261-72.

[15] Jones, G. OMR engine output file format. http://www.visiv.co.uk/tech-mro.htm.

[16] Hankinson, A., L. Pugin, and I. Fujinaga. 2010. An interchange format for optical music recognition applications. In *Proc. Int'l. Society for Music Information Retrieval*, Utrecht, NL.

[17] Bainbridge, D., C.G. Nevill-Manning, I.H. Witten, L.A. Smith, and R.J. Mcnab. 1999. Towards a digital library of popular music. In *Proc. ACM Conf. on Digital Libraries*, Berkeley, CA. 161–9.

[18] Catholic Church. 1963. *The Liber Usualis, with introduction and rubrics in English*. Tournai, Belgium: Desclée.

[19] Ramirez, C., and J. Ohya. 2011. OMR of early plainchant manuscripts in square notation: A two-stage system. In *Proc. SPIE*, 1-10.

[20] Hankinson, A., J.A. Burgoyne, G. Vigliensoni, and I. Fujinaga. 2012. Creating a large-scale searchable digital collection from printed music materials. In *Proc. Advances in Music Information Research*, Lyon, FR.

[21] Hankinson, A., P. Roland, and I. Fujinaga. 2011. The music encoding initiative as a document encoding framework. In *Proc. Int'l. Society for Music Information Retrieval*, Miami, FL.

[22] Vigliensoni, G., J.A. Burgoyne, A. Hankinson, and I. Fujinaga. 2011. Automatic pitch detection in printed square notation. In *Proc. Int'l. Society for Music Information Retrieval*, Miami, FL.

[23] Macmillan, K, M Droettboom, and I Fujinaga. 2001. Gamera: A structured document recognition application development environment. In *Proc. Int'l. Symposium on Music Information Retrieval*, Bloomington, IA. 15–6.

[24] Hankinson, A., P. Roland, and I. Fujinaga. 2011. The Music Encoding Initiative as a document encoding framework. In *Proc. Int'l. Conf. Music Information Retrieval*, Miami, FL.

[25] Downie, S. Evaluating a simple approach to music information retrieval: Conceiving melodic *n*-grams as text. PhD diss., University of Western Ontario, London, Ontario.

[26] Hankinson, A., W. Liu, L. Pugin, and I. Fujinaga. 2011. Diva.Js: A continuous document image viewing interface. *Code4lib Journal* 14.

MUSIC/VOICE SEPARATION USING THE SIMILARITY MATRIX

Zafar RAFII
Northwestern University
EECS Department
Evanston, IL, USA
zafarrafii@u.northwestern.edu

Bryan PARDO
Northwestern University
EECS Department
Evanston, IL, USA
pardo@northwestern.edu

ABSTRACT

Repetition is a fundamental element in generating and perceiving structure in music. Recent work has applied this principle to separate the musical background from the vocal foreground in a mixture, by simply extracting the underlying repeating structure. While existing methods are effective, they depend on an assumption of periodically repeating patterns. In this work, we generalize the repetition-based source separation approach to handle cases where repetitions also happen intermittently or without a fixed period, thus allowing the processing of music pieces with fast-varying repeating structures and isolated repeating elements. Instead of looking for periodicities, the proposed method uses a similarity matrix to identify the repeating elements. It then calculates a repeating spectrogram model using the median and extracts the repeating patterns using a time-frequency masking. Evaluation on a data set of 14 full-track real-world pop songs showed that use of a similarity matrix can overall improve on the separation performance compared with a previous repetition-based source separation method, and a recent competitive music/voice separation method, while still being computationally efficient.

1. INTRODUCTION

A system that can efficiently separate a song into foreground (e.g. the soloist or voice) and background (the musical accompaniment) components would be of great interest for a wide range of applications. These applications include instrument/vocalist identification, music/voice transcription, melody extraction, audio remixing, and karaoke.

While there are many approaches that have been applied to this problem (see Section 2), one promising approach is to use analysis of the repeating structure in the audio. Many musical pieces are characterized by an underlying repeating structure (e.g. drum loop or 4-measure vamp) over which varying elements are superimposed. This is especially true for pop songs where a singer often overlays varying vocals on a repeating accompaniment.

Recent work (see Section 2) has exploited repetition to separate the repeating musical background from the non-repeating vocal foreground. This work has relied on the assumption that there is a global or a local period of repetition in the musical background. Should repeated elements be present (e.g. reuse of the same chord voicing in the piano) but performed in a way that is not obviously periodic (e.g. occasional chordal piano “fills” at the appropriate moments), existing repetition-based approaches fail.

In this work, we generalize the repetition-based source separation approach to handle cases where repetitions also happen intermittently or without a fixed period. Instead of looking for periodicities, the proposed method identifies repeating elements by looking for similarities, by means of a similarity matrix. Once identified, median filtering is then performed on the repeating elements to calculate a repeating spectrogram model for the background. A time-frequency mask can finally be derived to extract the repeating patterns (see Section 3). This allows the processing of music pieces with fast-varying repeating structures and isolated repeating elements, without the need to identify periods of the repeating structure beforehand.

The rest of this paper is organized as follows. Section 2 describes the related work. Section 3 introduces the proposed method. Section 4 presents an evaluation of the proposed method on a data set of 14 full-track real-world pop songs, against a previous repetition-based source separation method, and a recent competitive music/voice separation method. Section 5 concludes this article.

2. RELATED WORK

There have been a number of approaches applied to the problem of separating the foreground (typically the voice) from the background. In stereo recordings, panning information (e.g. the vocalist is typically panned to the middle) can be applied. Such approach fails when the vocalist is not center-panned (e.g. many Beatles recordings). Cross-channel timing and amplitude differences can be applied in more complex frameworks, such as in the Degenerate Unmixing Estimation Technique (DUET) [8]. This approach is difficult to apply to pop music, due to the reverberant effects added, as well as the violation of the sparsity assumption for music mixtures.

Other music/voice separation methods focus on modeling either the music signal, by generally training an accompaniment model from the non-vocal segments [6, 11], or the vocal signal, by generally extracting the predominant pitch contour [7, 9], or both signals via hybrid models [1, 13]. Most of these methods require a training phase on audio with labeled vocal/non-vocal segments.

Recently, a relatively simple approach has also been proposed for music/voice separation. The method is based on a median filtering of the mixture spectrogram at different frequency resolutions, in such a way that the harmonic and percussive elements of the accompaniment can be smoothed out, leaving out the vocals [3].

Another recent and promising approach is to apply analysis of the repeating structure in the audio to extract the repeating musical background from the non-repeating vocal foreground. In this work, we focus on this approach.

The first method to explicitly use repetition to separate the musical background from the vocal foreground is the REpeating Pattern Extraction Technique (REPET) [12]. The method seeks to identify a global period for the repeating structure, so that it can build a model of the repeating background. This model is then used to construct a time-frequency mask to separate the repeating musical background from the non-repeating vocal foreground.

The original REPET method can be successfully applied for music/voice separation on short excerpts (e.g. 10 second verse) [12]. For complete music pieces, the repeating background is likely to vary over time (e.g. verse followed by chorus). An extended version of REPET was therefore later introduced to handle variations in the repeating structure [10]. Rather than finding a global period, the method tracks local periods of the repeating structure. In both cases, the algorithm needs to identify periods of the repeating structure, as both methods assume periodically repeating patterns.

In this work, we propose to generalize the repetition-based source separation approach to handle cases where repetitions also happen intermittently or without a fixed period, by using a similarity matrix.

3. PROPOSED METHOD

3.1 Similarity Matrix

The similarity matrix is a two-dimensional representation where each point (a, b) measures the (dis)similarity between any two elements a and b of a given sequence. Since repetition/similarity is what makes the structure in music, a similarity matrix calculated from an audio signal can help to reveal the musical structure that underlies it [4].

Given a single-channel mixture signal x, we first calculate its Short-Time Fourier Transform (STFT) X, using half-overlapping Hamming windows of N samples length. We then derive the magnitude spectrogram V by taking the absolute value of the elements of X, after discarding the symmetric part, while keeping the DC component.

We then define the similarity matrix S as the matrix multiplication between transposed V and V, after normalization of the columns of V by their Euclidean norm. In other words, each point (j_a, j_b) in S measures the cosine similarity between the time frames j_a and j_b of the mixture spectrogram V. The calculation of the similarity matrix S is shown in Equation 1.

$$
\begin{aligned}
&S(j_a, j_b) = \frac{\sum_{i=1}^{n} V(i, j_a) V(i, j_b)}{\sqrt{\sum_{i=1}^{n} V(i, j_a)^2} \sqrt{\sum_{i=1}^{n} V(i, j_b)^2}} \\
&\text{where } n = N/2 + 1 = \text{\# frequency channels} \\
&\forall j_a, j_b \in [1, m] \;\; \text{where } m = \text{\# time frames}
\end{aligned} \tag{1}
$$

3.2 Repeating Elements

Once the similarity matrix S is calculated, we use it to identify the repeating elements in the mixture spectrogram V. For all the frames j in V, we look for the frames that are the most similar to the given frame j and save them in a vector of indices J_j. Assuming that the non-repeating foreground ($\approx$ voice) is sparse and varied compared to the repeating background ($\approx$ music) - a reasonable assumption for voice in music, the repeating elements revealed by the similarity matrix should be those that form the underlying repeating structure ($\approx$ music). The use of a similarity matrix actually allows us to identify repeating elements that do not necessarily happen in a periodic fashion.

We add the following constraint parameters to the algorithm. To limit the number of repeating frames considered similar to the given frame j, we define k, the maximum allowed number of repeating frames. We define t, the minimum allowed threshold for the similarity between a repeating frame and the given frame ($t \in [0, 1]$). Consecutive frames can exhibit high similarity without representing new instances of the same structural element, since frame duration is unrelated to the duration of musical elements. We therefore define d, the minimum allowed (time) distance between two consecutive repeating frames deemed to be similar enough to indicate a repeating element.

3.3 Repeating Model

Once the repeating elements have been identified for all the frames j in the mixture spectrogram V through their corresponding vectors of indices J_j, we use them to derive a repeating spectrogram model W for the background. For all the frames j in V, we derive the corresponding frame j in W by taking the median of the corresponding repeating frames whose indices are given by vector J_j, for every frequency channel. The calculation of the repeating spectrogram model W is shown in Equation 2.

$$
\begin{aligned}
&W(i, j) = \underset{l \in [1, k]}{median} \{ V(i, J_j(l) \} \\
&\text{where } J_j = [j_1 \ldots j_k] = \text{indices of repeating frames} \\
&\text{where } k = \text{maximum number of repeating frames} \\
&\forall i \in [1, n] = \text{frequency channel index} \\
&\forall j \in [1, m] = \text{time frame index}
\end{aligned} \tag{2}
$$

The rationale is that, assuming that the non-repeating foreground ($\approx$ voice) has a sparse time-frequency representation compare to the time-frequency representation of the repeating background ($\approx$ music), time-frequency bins

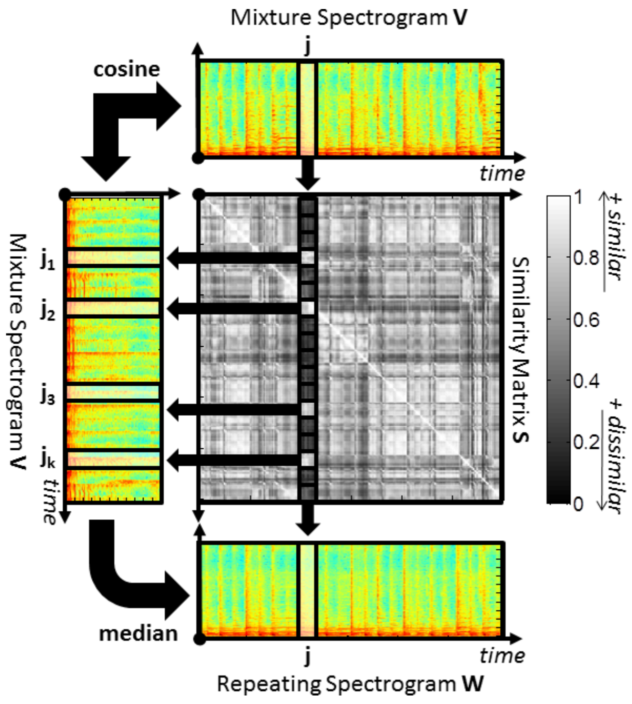


Figure 1. Derivation of the repeating spectrogram model W: (1) compute the similarity matrix S from the mixture spectrogram V using the cosine similarity measure; (2) for all frames j in V, identify the k frames $j_1 \dots j_k$ that are the most similar to frame j using S; (3) derive frame j of the repeating spectrogram model W by taking the median of the k frames $j_1 \dots j_k$ of V, for every frequency channel.

with little deviations between repeating frames would constitute a repeating pattern and would be captured by the median. Accordingly, time-frequency bins with large deviations between repeating frames would constitute a non-repeating pattern and would be removed by the median. The derivation of the repeating spectrogram model W from the mixture spectrogram V using the similarity matrix S is illustrated in Figure 1.

3.4 Time-frequency Mask

Once the repeating spectrogram model W is calculated, we use it to derive a time-frequency mask M. But first, we need to create a refined repeating spectrogram model W' for the background, by taking the minimum between W and V, for every time-frequency bin. Indeed, as noted in [10], if we assume that the non-negative mixture spectrogram V is the sum of a non-negative repeating spectrogram W and a non-negative non-repeating spectrogram $V - W$, then time-frequency bins in W can at most have the same value as the corresponding time-frequency bins in V. In other words, we want $W \leq V$, for every time-frequency bin; hence the use of the minimum function.

We then derive a time-frequency mask M by normalizing W' by V, for every time-frequency bin. The rationale is that time-frequency bins that are likely to constitute a repeating pattern in V will have values near 1 in M and will be weighted toward the repeating background ($\approx$ music). Accordingly, time-frequency bins that are unlikely to constitute a repeating pattern in V will have values near 0 in M and will be weighted toward the non-repeating foreground ($\approx$ voice). The calculation of the time-frequency mask M is shown in Equation 3.

$$\begin{aligned} W'(i,j) &= \min\big(W(i,j), V(i,j)\big) \\ M(i,j) &= \frac{W'(i,j)}{V(i,j)} \quad \text{with } M(i,j) \in [0,1] \\ \forall i &\in [1,n] = \text{frequency channel index} \\ \forall j &\in [1,m] = \text{time frame index} \end{aligned} \tag{3}$$

The time-frequency mask M is then symmetrized and applied to the STFT X of the mixture signal x. The estimated music signal is finally obtained by inverting the resulting STFT into the time domain. The estimated voice signal is obtained by simply subtracting the music signal from the mixture signal.

4. EVALUATION

4.1 Competitive Methods & Data Set

We label the proposed method, based on the use of a similarity matrix, *Proposed*. We compare separation performance of *Proposed* with two competitive music/voice separation methods on a data set of 14 full-track pop songs.

The first competitive method is an extension of the original REPET algorithm to handle variations in the underlying repeating structure [10]. We refer to this method as *REPET+*. The method first tracks local periods of the underlying repeating structure using a beat spectrogram, then models local estimates of the repeating background using the median, and finally extracts the repeating patterns from the mixture using a time-frequency mask. For the comparison, we used the separation results of *REPET+* with soft time-frequency masking and high-pass filtering with a cutoff frequency of 100 Hz on the voice estimates, as published in [10].

The second competitive method is the Multipass Median Filtering-based Separation (MMFS), another recently proposed simple and novel approach for music/voice separation [3]. The method is based on a median filtering of the mixture spectrogram at different frequency resolutions, in such a way that the harmonic and percussive elements of the accompaniment can be smoothed out, leaving out the vocals. For the comparison, we used the separation results of the best version of MMFS out of the four proposed versions, with high-pass filtering with a cutoff frequency of 100 Hz on the voice estimates, as published in [3].

The data set consists of 14 full-track real-world pop songs, in the form of split stereo WAVE files sampled at 44.1 kHz, with the accompaniment and vocals on the left and right channels, respectively. These 14 stereo sources were created from recordings released by the band *The Beach Boys*, where some of the accompaniments and vocals were made available as split stereo tracks [1] and separated tracks [2]. This data set was used in [10] for the evalu-

[1] Good Vibrations: Thirty Years of The Beach Boys, 1993
[2] The Pet Sounds Sessions, 1997

ation of *REPET+* against the best version of MMFS.

Following the framework adopted in [3] and [10], we then used those 14 stereo sources to create three data sets of 14 mono mixtures, by mixing the channels at three different voice-to-music ratios: -6 dB (music is louder), 0 dB (same original level), and 6 dB (voice is louder). Note that we are using the exact same data set as in [10], however it is not the exact same data set that was used in [3]. The authors of [3] did not mention which tracks they used for their experiments and also unlike them, but as in [10], we process the full tracks without segmenting them beforehand, since *Proposed* can handle long recordings, and this without memory or computational constraints.

4.2 Algorithm Parameters & Separation Measures

We calculated the STFT of each mixture for each of the three mixture sets (-6, 0, and 6 dB) using half-overlapping Hamming windows of $N = 2048$ samples length, corresponding to a duration of 46.4 milliseconds at a sampling frequency of 44.1 kHz. We then processed each mixture using *Proposed*. The parameters were fixed as follows: maximum number of repeating frames $k = 100$, minimum threshold for the similarity between a repeating frame and the given frame $t = 0$, and minimum distance between two consecutive repeating frames $d = 1$ second. Pilot experiments showed that those parameters lead to overall good separation results. For the comparison, we also applied a high-pass filtering with a cutoff frequency of 100 Hz on the voice estimates. This means that all the energy under 100 Hz in the voice estimates is transferred to the corresponding music estimates. The rational is that singing voice rarely happen below 100 Hz.

We measured separation performance by employing the *BSS_EVAL toolbox* [2]. The toolbox proposes a set of now widely adopted measures that intend to quantify the quality of the separation between a source and its corresponding estimate: Source-to-Distortion Ratio (SDR), Sources-to-Interferences Ratio (SIR), and Sources-to-Artifacts Ratio (SAR). Following the framework adopted in [3] and [10], we measured SDR, SIR, and SAR on segments of 45 second length from the music and voice estimates. Higher values of SDR, SIR, and SAR suggest better separation performance. We chose to use those measures because they are widely known and used, and also because they have been shown to be well correlated with human assessments of signal quality [5].

4.3 Comparative Results & Statistical Analysis

Figures 2, 3, and 4 show the separation performance using the SDR, SIR, and SAR, respectively, in dB, for the music component (top row) and the voice component (bottom row), at voice-to-music mixing ratio of -6 dB (left column), 0 dB (middle column), and 6 dB (right column). In each column, from left to right, the first results correspond to the best version of MMFS (*MMFS*), where the means of the distributions are represented as crosses (the standard deviations were not reported in [3]). The second results correspond to the extension of REPET for varying repeating structures (*REPET+*), where the means and standard deviations of the distributions are represented as error bars. The third results correspond to the proposed method with similarity matrix (*Proposed*), where the means and standard deviations of the distributions are represented as error bars. The mean values are displayed. Higher values are better.

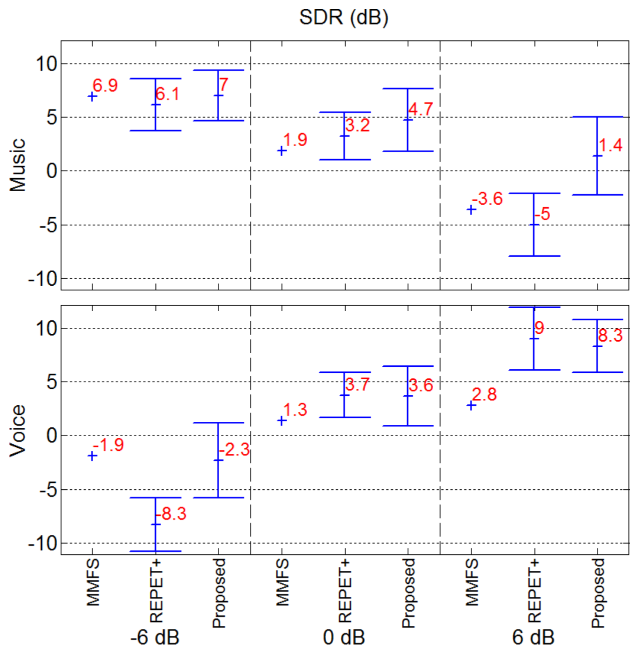


Figure 2. Separation performance using the SDR in dB, for the music component (top row) and the voice component (bottom row), at voice-to-music mixing ratio of -6 dB (left column), 0 dB (middle column), and 6 dB (right column), using the best version of MMFS (*MMFS*) (means represented as crosses), the extension of REPET for varying repeating structures (*REPET+*), and the proposed method with similarity matrix (*Proposed*) (means and standard deviations represented as error bars). Mean values are displayed. Higher values are better.

We compared the three different methods including a high-pass filtering with a cutoff frequency of 100 Hz on the voice estimates, because such post-processing of the estimates typically helps to produce better separation results. For our proposed method, the high-pass filtering increased SDR and SIR, for both the music and voice estimates. SAR however increased only for the music estimates. This is probably due to the fact that, although improving the separation performance overall, using a high-pass filtering on the voice estimates creates “holes” in their time-frequency representation, which tend to increase the separation artifacts, hence the decrease of SAR for the voice estimates.

As we can see in Figures 2, 3, and 4, as the voice-to-music ratio gets larger, SDR, SAR, and SIR get lower for the music estimates and larger for the voice estimates, and vice versa. This is an intuitive result also observed for *MMFS* and *REPET+*. Indeed, as the voice component gets louder compared to the music component, it then

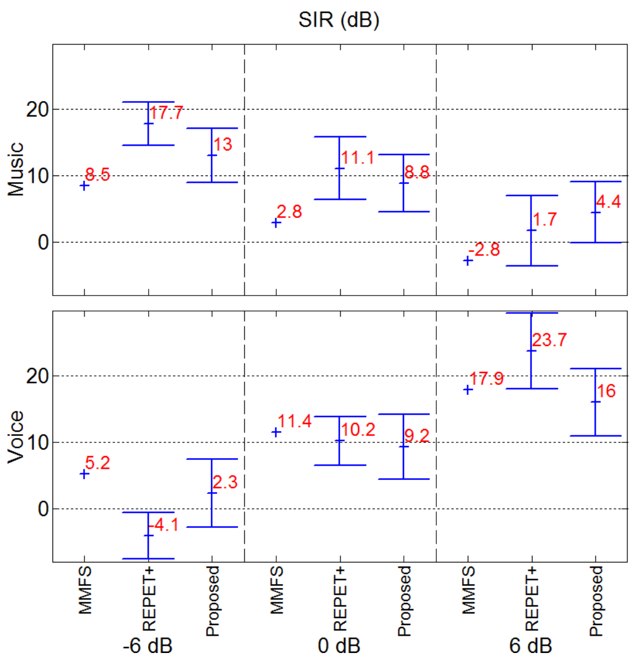


Figure 3. Separation performance using the SIR in dB.

becomes easier to extract the voice component, and accordingly harder to extract the music component, and vice versa. A multiple comparison test showed that those results were statistically significant in each case. We used an Analysis of Variance (ANOVA) when the compared distributions were all normal, and a Kruskal-Wallis test when at least one of the compared distributions was not normal. We used a Jarque-Bera normality test to determine if a distribution was normal or not.

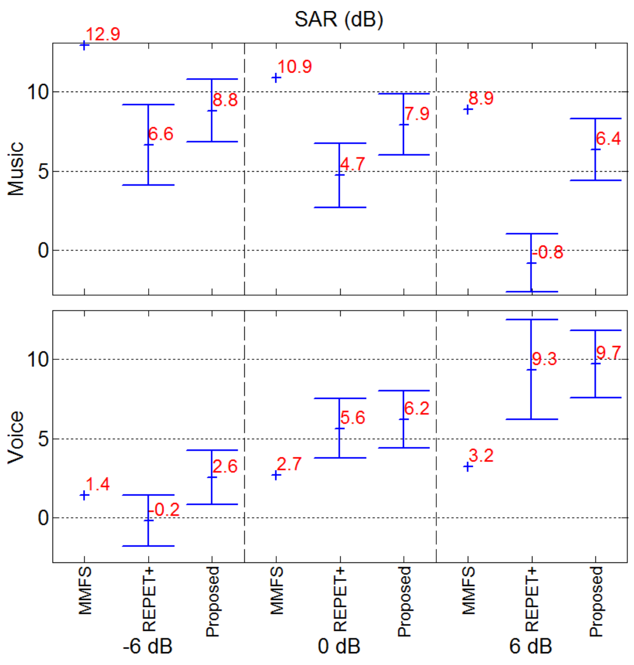


Figure 4. Separation performance using the SAR in dB.

As we can see in Figures 2, 3, and 4, compared with *MMFS*, *Proposed* gave overall better SDR for both the music and voice estimates, better SIR for the music estimates, and better SAR for the voice estimates, and this for all the three voice-to-music ratios. A one-sample t-test comparing the means of the distributions of *Proposed* with the means of *MMFS* (since the only values provided in [3] were the means) showed that those results were statistically significant in each case, except for the SDR at voice-to-music ratio of -6 dB, where the improvement of *Proposed* compared with *MMFS* was not significant for the music estimates, and a decrease, although not significant, was observed for the voice estimates. This suggest that, compared with *MMFS*, *Proposed* has globally better separation performance, particularly it is better at removing the “vocal” interferences from the accompaniment, and at limiting the separation artifacts in the voice estimates.

As we can also see in Figures 2, 3, and 4, for the music estimates, compared with *REPET+*, *Proposed* gave overall better SDR and SAR for all the three voice-to-music ratios, and better SIR at voice-to-music ratio of 6 dB. A multiple comparison test showed that those results were statistically significant in each case, except for the SDR at voice-to-music ratio of -6 dB where the improvement of *Proposed* compared with *REPET+* was not significant. For the voice estimates, compared with *REPET+*, *Proposed* gave overall better SAR for all the three voice-to-music ratios, and better SDR and SIR at voice-to-music ratio of -6 dB. A multiple comparison test showed that those results were statistically significant in each case, except for the SAR where the improvement of *Proposed* compared with *REPET+* was only significant at voice-to-music ratio of -6 dB. We used ANOVA when the compared distributions were all normal, and a Kruskal-Wallis test when at least one of the compared distributions was not normal. This suggest that, compared with *REPET+*, *Proposed* has globally better separation performance for a component (music or voice), as the given component becomes softer compared with the other one.

The average computation time of *Proposed* over all the mixtures and all of the three mixture sets (-6, 0, and 6 dB) was 0.563 second for 1 second of mixture, when implemented in Matlab on a PC with Intel(R) Core(TM)2 Quad CPU of 2.66 GHz and 6 GB of RAM. In other words, *Proposed* can perform music/voice separation of a mixture audio in half the time of the playback of the audio, for recordings of the length of a typical pop song. This is encouraging, since *Proposed* builds a similarity matrix that is $O(n^2)$, where n is the length of the audio file. As a point of comparison, the average computation time for *REPET+* for the exact same data set was 1.1830 second for 1 second of mixture [10].

5. CONCLUSION

In this work, we have proposed a generalization of the REpeating Pattern Extraction Technique (REPET) method for the task of music/voice separation, based on the calculation of a similarity matrix. The REPET approach is based on the separation of a musical background from a vocal foreground, by extraction of the underlying repeating structure. The basic idea is to identify elements that exhibit similarity, and compare them to repeating models derived from them to extract the repeating patterns.

Unlike the previous REPET methods that assume periodically repeating patterns, the proposed method with similarity matrix generalizes to repeating structures where repetitions can also happen intermittently or without a fixed period, therefore allowing the processing of music pieces with fast-varying repeating structures and isolated repeating elements, without the need to identify periods of the underlying repeating structure beforehand.

Evaluation on a data set of 14 full-track real-world pop songs showed that the proposed generalization of REPET with similarity matrix can overall improve on the separation performance compared with the extension of REPET for varying repeating structures, and another recent competitive music/voice separation method based on median filtering, while still being computationally efficient. Given the SDR, which can be understood as a measure of the overall quality of the separation, our evaluation showed that when the results between the proposed method and the competitive methods were statistically significant, the proposed method gave higher results, and this compared with both the competitive methods.

The proposed generalization of REPET is only based on a similarity matrix. In other words, it does not depend on particular features, does not rely on complex frameworks, and does not need prior training. Because it is only based on self-similarity, it has the advantage of being simple, fast, blind, and therefore completely and easily automatable.

6. ACKNOWLEDGMENTS

The authors would like to thank Antoine Liutkus, Roland Badeau, Gaël Richard, and their colleagues from Telecom ParisTech for the fruitful discussions. This work was supported by NSF grant number IIS-0812314.

7. REFERENCES

[1] Jean-Louis Durrieu, Bertrand David, and Gaël Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *IEEE Journal on Selected Topics on Signal Processing*, 5(6):1180–1191, October 2011.

[2] Cedric Févotte, Rémi Gribonval, and Emmanuel Vincent. BSS_EVAL toolbox user guide. Technical Report 1706, IRISA, Rennes, France, April 2005. http://www.irisa.fr/metiss/bss eval/.

[3] Derry FitzGerald and Mikel Gainza. Single channel vocal separation using median filtering and factorisation techniques. *ISAST Transactions on Electronic and Signal Processing*, 4(1):62–73, 2010.

[4] Jonathan Foote. Visualizing music and audio using self-similarity. In *7th ACM International Conference on Multimedia (Part 1)*, pages 77–80, Orlando, FL, USA, October 30-November 0 1999.

[5] Brendan Fox, Andrew Sabin, Bryan Pardo, and Alec Zopf. Modeling perceptual similarity of audio signals for blind source separation evaluation. In *7th International Conference on Independent Component Analysis*, pages 454–461, London, UK, September 09-12 2007.

[6] Jinyu Han and Ching-Wei Chen. Improving melody extraction using probabilistic latent component analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, May 22-27 2011.

[7] Chao-Ling Hsu and Jyh-Shing Roger Jang. On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2):310–319, February 2010.

[8] Alexander Jourjine, Scott Rickard, and Özgür Yilmaz. Blind separation of disjoint orthogonal signals: Demixing n sources from 2 mixtures. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 2985–2988, Istanbul, Turkey, June 5-9 2000.

[9] Yipeng Li and DeLiang Wang. Separation of singing voice from music accompaniment for monaural recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1475–1487, May 2007.

[10] Antoine Liutkus, Zafar Rafii, Roland Badeau, Bryan Pardo, and Gaël Richard. Adaptive filtering for music/voice separation exploiting the repeating musical structure. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Kyoto, Japan, March 25-30 2012.

[11] Alexey Ozerov, Pierrick Philippe, Frédéric Bimbot, and Rémi Gribonval. Adaptation of Bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(5):1564–1578, July 2007.

[12] Zafar Rafii and Bryan Pardo. A simple music/voice separation system based on the extraction of the repeating musical structure. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, May 22-27 2011.

[13] Tuomas Virtanen, Annamaria Mesaros, and Matti Ryynänen. Combining pitch-based inference and non-negative spectrogram factorization in separating vocals from polyphonic music. In *ISCA Tutorial and Research Workshop on Statistical and Perceptual Audition*, pages 17–20, Brisbane, Australia, 21 September 2008.

BREATHY OR RESONANT – A CONTROLLED AND CURATED DATASET FOR PHONATION MODE DETECTION IN SINGING

Polina Proutskova
Goldsmiths, University of London
proutskova@googlemail.com

Christophe Rhodes
Goldsmiths, University of London
c.rhodes@gold.ac.uk

Geraint Wiggins
Queen Mary, University of London
geraint.wiggins@eecs.qmul.ac.uk

Tim Crawford
Goldsmiths, University of London
t.crawford@gold.ac.uk

ABSTRACT

This paper presents a new reference dataset of sustained, sung vowels with attached labels indicating the phonation mode. The dataset is intended for training computational models for automated phonation mode detection.

Four phonation modes are distinguished by Johan Sundberg [1]: breathy, neutral, flow (or resonant) and pressed. The presented dataset consists of ca. 700 recordings of nine vowels from several languages, sung at various pitches in various phonation modes. The recorded sounds were produced by one female singer under controlled conditions, following recommendations by voice acoustics researchers.

While datasets on phonation modes in speech exist, such resources for singing are not available. Our dataset closes this gap and offers researchers in various disciplines a reference and a training set. It will be made available online under Creative Commons license. Also, the format of the dataset is extensible. Further content additions and future support for the dataset are planned.

1. MOTIVATION: NARROW, WIDE, BREATHY, RESONANT SINGING IN VARIOUS DISCIPLINES

Phonation modes play an important role in singing: they are an essential characteristic of a singing style – all musical traditions have cultural preferences for the use of one or more phonation modes; they are used as a means for expressive performance; they can be indicative of voice disorders; subtle changes in phonation mode production are used routinely by singing teachers to determine the progress of a student.

Johan Sundberg in his seminal work “The Science Of The Singing Voice” identifies four different phonation modes in singing: breathy, neutral, flow (called resonant by other authors) and pressed [1]. To illustrate the differences between phonation modes let us bring some well-known examples.

Breathy vocalisation is used skillfully by jazz and popular music singers to express qualities like sweetness or sexuality: think of Marilyn Monroe's most famous performances like “I wanna be loved by you”[1] or “Happy birthday Mr President”[2]; or listen to Ella Fitzgerald's and Louis Armstrong's “Dream a little dream of me”[3]. This mode of vocal production can easily be distinguished by human listeners from the flow phonation mode, such as a resonant, vibrating vocalising by Liza Minelli on her “New York, New York”[4]; and from the pressed phonation, e.g. the tense, forceful voice of James Brown in “I feel good”[5].

At the same time, Liza Minelli can be quite forceful; Ella Fitzgerald's vocals are very dominant but light and economical most of the time, because flow phonation is natural for her voice. All the singers mentioned above use flow phonation, and all of them have their personal preferences for varying phonation and using these variations as stylistic markers or as their individual expressive devices.

In Muslim countries of the Old High Culture a call for prayer can be heard five times a day from the minarets of the mosques. Call for prayer is an art form and Muezzins are experts in using squeezed, narrow sound (pressed phonation) in their singing which makes their performances very expressive. Their vocal technique is very different from a rounded, flying performance of a Western European Gregorian chant (neutral phonation) and is at the same time distant from the brassy, resonant Greek or Russian liturgic singing (flow phonation).[6]

While the term phonation mode is borrowed from voice acoustics, the differentiation between breathy and pressed voices, between tense and open singing is operational in many voice-related research areas: ethnomusicology, singing education, medical research (phoniatrics, vocology) as well as in linguistics (phonetics). This is

[1]http://www.youtube.com/watch?v=MLU0jndUGg4

[2]http://www.youtube.com/watch?v=k4SLSlSmW74

[3]http://www.youtube.com/watch?v=j6TmogXhOZ8

[4]http://www.youtube.com/watch?v=rgusCINe260

[5]http://www.youtube.com/watch?v=XgDrJ5Z2rKw

[6]Here, again, situation with employed phonation modes can be quite ambiguous. For example, Greek Byzantine singers use somewhat pressed, nasalized phonation on higher pitches quite a lot. In fact, a Byzantine singer with a higher range often cannot be distinguished from an Arabic singer of a similar range in terms of their phonation mode usage. Also, some Gregorian chant interpreters such as Ensemble Organum deliberately use flow phonation in their performance.

how an ethnomusicologist Alan Lomax describes the difference between narrow and wide vocalisation:

"The measure concerns the contrast between the voices which sound mellow, relaxed and richly resonant (we call this *wide*) and the voices which sound tense, pinched and restricted in resonance (which we call *narrow*). Many singing styles can be characterized as having one or the other; in some rare cases both may occur; and many ways of vocalizing (like everyday American speech) are neutral in width – these we call *mid*, singers with a "speech" tone." [2, p. 125]. Lomax then gives a number of examples: narrow singing from Indonesia and Thailand; wide, open singing from Eastern Europe; the mid mode form the US and from Ireland.

These examples demonstrate that breathy, pressed or resonant singing production can be representative of a singing style or even a music culture. While each voice is different and two singers never sing the same way, every musical tradition displays cultural preferences for the use of particular phonation mode(s), which are imposed on the singers performing in this tradition. In many cases a single phonation mode is encouraged: for example baritone singers in Western classical music are trained to sing in flow phonation and move through their singing career using just this phonation mode. In contrast, in classical Ottoman tradition a singer was expected to operate in all four phonation modes.

Apart from being a cultural characteristic, breathy or tense vocalisation can be indicative of vocal disorders: hypofunction and hyperfunction of the glottis [3]. Their diagnostics and treatment are a prime concern in the disciplines of vocology (voice habilitation) and phoniatrics (in case of functional or anatomic pathologies) [4].

While in some examples even a less experienced listener can easily distinguish between various phonation modes, in other cases this distinction can be very subtle and requires training and expertise to be identified correctly. Lomax refers to his narrow vs wide singing descriptor (he calls this descriptor vocal width or vocal tension) as an "emotionally loaded quality" and thus explains why some people have difficulties in rating it [2]. Vocal width is one of 36 descriptors of the Cantometrics system – a global parametrisation of world's singing styles. Lomax and his Cantometrics team manually rated more than 5000 recordings of singing from around the world. Of all 36 descriptors vocal width appeared to be the hardest to rate consistently: the inter-rater consensus scores for this descriptor are the lowest (see [2], p. 168). Victor Grauer, the co-inventor of Cantometrics, admitted in personal communication (February 2011) that this descriptor is the most difficult to rate.

Naturally, voice therapists are experts in vocal production and could serve as experts for manual rating of phonation modes. Although, in practice their work is often more tailored to the needs of speech professionals. In singing it's singing teachers/educators who have the deepest operational knowledge of all the issues related to vocal production and in particular to phonation modes. Most singing students display various kinds of voice hypo- and/or hyperfunction during the stages of their progress. The students' perception mechanisms are usually not sufficient for self-control (in absence of any visual or any reliable auditory indicators). It is thus the task of the teacher to identify and to correct the subtlest dysfunction on the spot, over and over again, until the student has gained the bodily controls needed to regulate the voice source function on an automatic level.

2. PHONATION MODES IN VOICE ACOUSTICS – PREVIOUS WORK

Due to complications in terminology of narrow vs. wide singing within and across disciplines as well as to the subjectivity of the distinction between phonation modes, we turn to voice acoustics for objective definitions and physically measurable effects. The four phonation modes introduced by Sundberg [1]: breathy, neutral, flow (called resonant by other authors) and pressed are vocal production qualities resulting from the voice source (the vibrating vocal folds). In particular they are closely related to glottal resistance which is defined as the quotient of subglottal pressure to glottal airflow. A low subglottal pressure combined with a high glottal flow results in a breathy phonation. Pressed phonation arises when a high subglottal pressure is accompanied by a low glottal flow. The neutral mode lies between these two extremes. The flow phonation is characterised by a lower subglottal pressure than in pressed mode and also by a lower adduction force on the vocal folds. It is an economical voice production mode, because it uses much less effort than in pressed mode gaining a similar sound level, which can be significantly higher than in a neutral mode. At the same time the flow phonation allows various resonances of the vocal tract to be used most effectively, while the pressed phonation tends to restrict some of them.

Given the above, the four phonation modes are not discrete states of vocal production but are rather areas in a continuum space which can be distinguished on the psychoacoustic level. This continuum is not fully ordered: while breathy and pressed modes represent its two extremes, there are endless states between them, and flow phonation is a sweet spot in that continuum which optimises psychoacoustic values such as loudness and overtone richness. As we know from singing and teaching practice, a singer is usually capable of using one or more localities of this space.

While the phonation mode of a singing fragment can only be identified subjectively in a psychoacoustic experiment, the glottal wave - the signal produced by the voice source - can be measured during singing by means of a laryngograph (electroglottograph), a non-invasive tool which sends a small current through the larynx and records the changes in resistance [5-6]. Figure 1 shows typical graphs of the glottal wave in all phonation modes.

Electroglottography can be used to measure the glottal wave during the singing process. If, in contrast, audio recordings of previous events are studied, this technique is not applicable.

To analyse the sound production of the voice source in a recording the technique called inverse filtering is often used: the resonances of the vocal tract are estimated from the original signal and a filter is constructed to eliminate them [7-10]. Applying this filter to the original signal results in an estimation of the glottal wave.

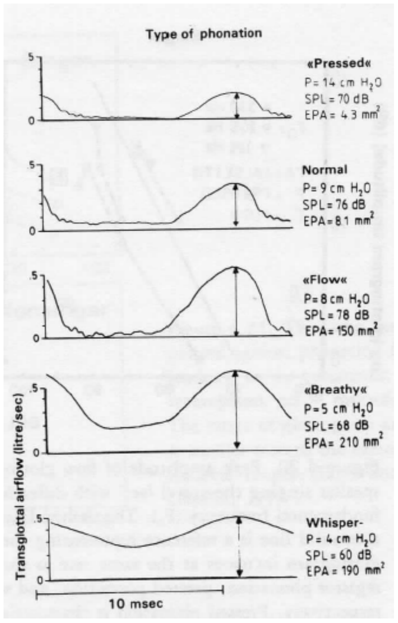


Figure 1. Typical graphs of the glottal wave pulse functions in various phonation modes (from [1], p. 85, used with permission of Northern Illinois University Press)

A number of publications dedicated to detection of pressed and breathy phonation modes employed descriptors derived from the glottal wave such as amplitude quotient (AQ), normalised amplitude quotient (NAQ) and the difference between the first two harmonics (H1-H2) [8, 11-14].

Unfortunately, all of them rely on internal datasets for their experiments which are neither well documented and controlled nor are they available to other researchers for benchmarking or new studies.

There have been single attempts to determine dominant phonation modes or typical values of glottal wave descriptors for various singing styles. For example Thalén and Sundberg [15] studied Western classical music, pop, jazz and blues, and in a later publication Zangger Borch and Sundberg [16] looked at rock, pop, soul and Swedish dance. Both these studies worked with recordings by just one singer. As a starting point both studies were certainly instructive. Unfortunately it is virtually impossible to make generalisations about a musical style based on samples from just one singer. At the same time the methodology suggested in these papers doesn't scale to batch processing applications. The datasets were not made available to other researchers.

In order to address high-level semantic questions about music like the relationship between a singing style and phonation modes using MIR, new approaches have to be developed that would allow processing of large, real-life data collections. One of the main obstacles for such a development is a lack of reference and training datasets that are well documented and supported and are available to all researchers.

In this paper we present a new dataset that will close this gap and will be a first step in the development of new scalable methods for study of phonation modes in singing. While we also start with recordings by only one singer, the format of the dataset is extensible (see Section 3.4). We plan to add further recordings in future as described in Section 4. Contributions by other researchers will also be welcome.

3. THE DATASET

3.1 The recordings

The dataset consists of ca. 700 WAV files. Each file contains a single recording of a sustained sung vowel. Recordings are of 750 sec length on average. We recommend to use 300 ms around the middle of the samples for analysis - here we can guarantee a relative stability in pitch, intensity, phonation and articulation (beginnings and ends of the samples can be less stable).

Sound	examples	Symbols used in the labels
[a:]	/a/ - low front unrounded sound, like in English *father*, German *Rat* or in Russian *мам*	A
[e:]	/e/ - high-mid front unrounded vowel, like in English *get*, German *Esel*, Russian *место*	E
[i:]	/i/ - high front unrounded, like in English *free*, German *Genie*, Russian *вид*	I
[o:]	/o/ - high-mid back rounded, like in German *rot*, Russian *кот,* somewhat similar to English *caught*	O
[ø:]	High-mid front rounded vowel, like German /ö/ in *schön*	OE
[u:]	/u/ - high back rounded, like in English *boot*, German *Fuß*, Russian *плуг*	U
[y:]	High front rounded sound, like German or Turkish /ü/, e.g. in German *müde*	UE
[ɨ:]	High central unrounded vowel, Russian /ы/ like in *мы*, similar to English *roses*	Y
[ɛ:]	Low-mid front unrounded, German /ä/ like in *Ähre*, Russian /э/ like in этот, similar to [æ] in Eng-	AE

	lish *cat*	

Table 1. The vowels represented in the dataset.

Pitches	Modes
A3 - G4	Breathy, neutral, flow, pressed
G4# - C5	Breathy, neutral, pressed
C5# - G5	Breathy, neutral

Table 2. This table indicates which phonation modes are represented for particular pitches in the dataset.

The vowel sounds represented on the recordings are listed in Table 1. These sounds were sung on all pitches on a semitone scale from A3 to G5, in every phonation mode given in Table 2.

3.2 The singer

All the recordings were produced by one female singer. This excludes any variation that would necessarily arise between singers, which is useful particularly at the initial stages of classification model training and testing.

The singer was professionally trained, with expertise in Western popular and in Russian traditional singing and a profound experience in a number of other music traditions.

The singer's vocal range is approximately D3 – C6, with the working range being usually limited to G3 – F5. At both extreme ends of the range, phonation became unreliable and they were not included into the dataset. The singer's break between the modal and the head (falsetto) register is around E5, thus the surrounding pitches(D5# to F5#) can also be less reliable. Still we decided to include vocalisation in the head register into the dataset to make it more representative, thus all pitches up to G5 were included.

In the head register the singer was unable to produce pressed sounds, thus the pressed phonation mode is only represented up to the upper end of the modal register (see Table 2). Why this is the case seems to be an unsolved problem. While this seems to be common among singers of various traditions in Europe and the Near East, it is unclear whether in other cultures (e.g. in some East Asian traditions) the singers are in fact capable of producing pressed vocalisation in their head register. This observation leads to the question whether the ability to use particular phonation modes on particular pitches is innate or ontogenetic (culturally constructed).

Also, flow phonation could only be produced in the chest voice – up to A4 (recordings up to G4 retained for the dataset). Above A4 it becomes impossible to sing most vowels in the flow mode; at the same time, the neutral mode in the middle and head voice partly gains the qualities of the flow mode, such as intensity and richness in overtones, though it is very different from the chesty flow phonation. The singer reported from her experience of teaching Russian traditional singing, which heavily uses the flow mode, that this limit is typical for female singers, though some exceptional performers are capable of producing the flow phonation at as high as C5.

In the lower range, at G3 and below, the opposite is the case: the neutral phonation becomes more and more similar to the flow mode – for this reason recordings below A3 were excluded from the dataset.

The singer apparently had more difficulties with some vowels than with others in particular modes. For example, high front sounds like [i:] and [y:] proved to be harder to achieve in flow phonation.

3.3 Recording conditions

The recordings were made with Olympus LS10 linear PCM digital recorder. We chose 96 kHz sampling rate and 24 bits bit resolution in compliance with the recommendations for acoustic analysis and archiving by the International Association of Sound- and Audiovisual Archives (IASA TC-04) [17].

The built-in high-sensitivity, low-noise stereo microphone of Olympus LS10 is a combination of two microphone heads positioned at an 90° angle. It has an overall frequency response 20 – 44000 Hz. In the frequency range of 150 -3000 Hz it displays a flat frequency response of ±2dB and in the range up to 20 kHz the response is ±5dB.

The lowest fundamental frequency recorded was 220 Hz (A3) which is about ten times higher than the microphone's low frequency response limit. This guarantees the flat phase response and preserves the exact shape of the waveform – a necessary condition for applications such as inverse filtering [18].

The highest frequencies perceived by the human ear are about 20 kHz which is within the microphone's flat response range and is way below the half of the upper limit of the microphone's frequency response. See [18] for detailed instructions on the choice and positioning of the microphone.

The recorder and the microphone were positioned horizontally at the level of the singer's mouth, at the distance of 50 cm as recommended by the manufacturer for best voice capturing.

The recording session took place in a quiet room environment. The requirement of a signal-to-noise ratio of at least 15 dB has been adhered to [18].

3.4 The labels

The metadata is stored in a table of a relational database, see Table 3. This way of organising metadata is advantageous, because it can be easily extended by further fields and is scalable for unlimited number of entries and relationships. For example if we add recordings by other singers, a new field indicating the singer will be introduced to the table.

Labels were provided by the singer. They mark the pitch, the vowel and the phonation mode the singer intended to reproduce. We also performed a listening test, excluding all recordings where phonation was ambiguous or of a poor quality.

Metadata fields	ID	File	Pitch	Vowel	Phonation mode	Version
example	212	212.wav	C5#	AE	breathy	2

Table 3. Metadata fields of the dataset. For vowel symbols please consult Table 1. Version is optional, it is only used when several recordings of the same vowel at the same pitch in the same phonation mode were recorded. Currently simple numeric IDs are used. In future a use of content derived IDs is planned.

3.5 Dataset availability and license

The dataset will be made available for download under Creative Commons CC BY-NC-SA license. This license allows free sharing of the dataset as well as altering it or building new work based upon it. There are following conditions for the use of the dataset according to this license:

- attribution – reference the creators
- no commercial use
- share alike – if you alter, transform or build upon it, you may distribute the result only under the same license.

4. FUTURE WORK

Our dataset is a first step in creating reference and training collections for the study of phonation mode use in singing. There are several directions in which this dataset can be improved and extended:

1. To further improve recording quality for the particular task of glottal wave estimation a professional microphone specifically designated for voice measurements should be used for the production of the recordings (LS2-type microphones as specified by IEC 61094-1 and ANSI S1.15-1997 standards). For other experiment designs, a dataset with varying recording quality and recording conditions could be useful.

2. To enhance the reliability of the labels, they can be verified by independent experts, ideally by singing teachers representing various music cultures.

3. Electroglottograph can be used to measure the glottal wave at the singer's glottis during recording. This would allow more objective judgements about the phonation mode. These measurements would also provide an excellent reference for glottal wave estimations on new, unseen data.

4. Alternatively, if an exact measurement of the glottal airflow is required, an airflow mask developed by Rothenberg can be used, which also has an advantage of the low frequency limit of 0 Hz [19].

5. The scope of the dataset can be generalised by including recordings of other singers, male, female as well as children. This would introduce inter-performer variation, which needs to be studied and is necessary to construct real-life classifiers. It is important that singers from different musical traditions are represented, because the ability to utilise various phonation modes can vary greatly across cultures. Ideally, a representative dataset with recordings from all around the globe could be compiled, which would allow to study the distribution of phonation mode use in singing among humans.

6. Another way of generalisation, in particular in view of practical tasks of automatic phonation mode detection, would be to introduce recordings by groups of singers, from small groups to large choirs. Also, recordings where singers are accompanied by musical instruments could be included.

5. CONCLUSIONS

Phonation mode is an important characteristic of singing, playing a vital role in many singing-related disciplines. It remains under-researched, one of the reasons being the lack of reference and training data. The dataset presented here closes this gap. It is aimed at MIR researchers who wish to develop automated methods for phonation mode detection in singing.

6. REFERENCES

[1] Sundberg, J. (1987). *The science of the singing voice*. Illinois University Press.

[2] Lomax, A. (1977). *Cantometrics: A Method of Musical Anthropology (audio-cassettes and handbook)*. Berkeley: University of California Media Extension Center.

[3] Froeschels, E. (1943). Hygiene of the voice. *Arch Otolaryngol.*, 38(2):122–130.

[4] Ramig, L. O. and Verdolini, K. (1998). Journal of speech, language, and hearing research. *Journal of Speech, Language, and Hearing Research*, 41:101–116.

[5] Howard, D. M. (2010). Electrolaryngographically revealed aspects of the voice source in singing. *Logopedics Phoniatrics Vocology*, 35(2):81–89.

[6] Pulakka, H. (2005). Analysis of human voice production using inverse filtering, high-speed imaging, and electroglottography. Master's thesis, HELSINKI UNIVERSITY OF TECHNOLOGY, Department of Computer Science and Engineering.

[7] Fritzell, B. (1992). Inverse filtering. *Journal of Voice*, 6(2):111–114.

[8] Walker, J. and Murphy, P. (2007). A review of glottal waveform analysis. In *PROGRESS IN NONLINEAR SPEECH PROCESSING*, volume 4391 of *Lecture Notes in Computer Science*, pages 1–21. Springer.

[9] Drugman, T., Bozkurt, B., and Dutoit, T. (2012). A comparative study of glottal source estimation techniques. *Computer Speech and Language*, 26:20–34.

[10] Gudnason, J., Mark R.P. Thomas, D. P. E., and Naylor, P. A. (2012). Data-driven voice source waveform analysis and synthesis. *Speech Communication*, 54:199–211.

[11] Orr, R., Cranen, B., de Jong, F., d'Alessandro, C., and Scherer, K. (2003). An investigation of the parameters derived from the inverse filtering of flow and microphone signals. In *Voice Quality: Functions, Analysis and Synthesis (VOQUAL '03)*. Taalwetenschap Otorhinolaryngology.

[12] Drugman, T., Dubuisson, T., Moinet, A., D'Alessandro, N., and Dutoit, T. (2008). Glottal source estimation robustness. In *Proc. of the IEEE International Conference on Signal Processing and Multimedia Applications (SIGMAP08)*.

[13] Lehto, L., Airas, M., Björkner, E., Sundberg, J., and Alku, P. (2007). Comparison of two inverse filtering methods in parameterization of the glottal closing phase characteristics in different phonation types. *J Voice*, 21(2):138–50.

[14] Sundberg, J., Thalén, M., Alku, P., and Vilkman, E. (2004). Estimating perceived phonatory pressedness in singing from flow glottograms. *J Voice*, 18(1):56–62.

[15] Thalén, M. and Sundberg, J. (2001). Describing different styles of singing: a comparison of a female singer's voice source in "classical", "pop", "jazz" and "blues". *Logoped Phoniatr Vocol*, 26(2):82–93.

[16] Borch, D. Z. and Sundberg, J. (2011). Some phonatory and resonatory characteristics of the rock, pop, soul, and swedish dance band styles of singing. *J Voice*, 25(5):532–7.

[17] K. Bradley (eds) (2009). *Guidelines on the Production and Preservation of Digital Audio Objects: Standards, Recommended Practices and Strategies (IASA-TC 04)*. IASA (International Association for Sound- and Audiovisual Archives) Technical Committee, 2 edition.

[18] Svec, J. G. and Granqvist, S. (2010). Guidelines for selecting microphones for human voice production research. *American Journal of Speech-Language Pathology*, 19:356–368.

[19] Rothenberg, M. (1973). A new inverse-filtering technique for deriving the glottal air flow waveform during voicing. *The Journal of the Acoustical Society of America*, 53:1632–1645.

ROBUST SINGER IDENTIFICATION IN POLYPHONIC MUSIC USING MELODY ENHANCEMENT AND UNCERTAINTY-BASED LEARNING

Mathieu Lagrange
STMS - IRCAM -
CNRS - UPMC
mathieu.lagrange@ircam.fr

Alexey Ozerov
Technicolor
Research & Innovation, France
alexey.ozerov@technicolor.com

Emmanuel Vincent
INRIA, Centre de Rennes -
Bretagne Atlantique
emmanuel.vincent@inria.fr

ABSTRACT

Enhancing specific parts of a polyphonic music signal is believed to be a promising way of breaking the glass ceiling that most Music Information Retrieval (MIR) systems are now facing. The use of signal enhancement as a pre-processing step has led to limited improvement though, because distortions inevitably remain in the enhanced signals that may propagate to the subsequent feature extraction and classification stages. Previous studies attempting to reduce the impact of these distortions have relied on the use of feature weighting or missing feature theory. Based on advances in the field of noise-robust speech recognition, we represent the uncertainty about the enhanced signals via a Gaussian distribution instead that is subsequently propagated to the features and to the classifier. We introduce new methods to estimate the uncertainty from the signal in a fully automatic manner and to learn the classifier directly from polyphonic data. We illustrate the results by considering the task of identifying, from a given set of singers, which one is singing at a given time in a given song. Experimental results demonstrate the relevance of our approach.

1. INTRODUCTION

Being able to focus on specific parts of a polyphonic musical signal is believed to be a promising way of breaking the glass ceiling that most Music Information Retrieval (MIR) tasks are now facing [3]. Many approaches were recently proposed to enhance specific signals (e.g., vocals, drums, bass) by means of source separation methods [7, 19].

The benefit of signal enhancement has already been proven for several MIR classification tasks, such as singer identification [10, 16], instrument recognition [12], tempo estimation [4], and chord recognition [20]. In most of those works, signal enhancement was used as a pre-processing step. Since the enhancement process must operate with limited prior knowledge about the properties of the specific parts to be enhanced, distortions inevitably remain in the enhanced signals that propagate to the subsequent feature extraction and classification stages resulting in limited improvement or even degradation of the classification accuracy.

A few studies have attempted to reduce the impact of these distortions on the classification accuracy. In [10, 15], feature weighting and frame selection techniques were proposed that associate a constant reliability weight to each feature over all time frames or to all features in each time frame. In practice, however, distortions affect different features in different time frames so that the assumption of constant reliability does not hold. A more powerful approach consists of estimating and exploiting the reliability of each feature within each time frame. A first step in this direction was taken in [8], where recognition of musical instruments in polyphonic audio was achieved using the missing feature theory. This theory adopted from noise-robust speech recognition assumes that only certain features are observed in each time frame while other features are missing and thus discarded from the classification process [5].

Nevertheless, the approach in [8] has the following three limitations. First, such *binary uncertainty* (either observed or missing) does not account for partially distorted features nor for correlations between the distortions affecting different features. To avoid this limitation, it was proposed in the speech recognition field to use the so-called *Gaussian uncertainty* [6], where the distortions over a feature vector are modeled as a zero-mean multivariate Gaussian with possibly non-diagonal covariance matrix. Second, this approach necessitates clean data to train the classifiers, while for some tasks, e.g., singer identification, collecting such clean data may be impossible. Third, the approach in [8] relies on manual f0 annotation and its use in a fully automatic system has not been demonstrated.

The contribution of this paper is threefold: (1) promoting the use of Gaussian uncertainty instead of binary uncertainty for robust classification in the field of MIR, (2) using a fully automatic procedure for Gaussian uncertainty estimation, (3) learning classifiers directly from noisy data with Gaussian uncertainty.

To illustrate the potential of the proposed approach we consider in this paper the task of singer identification in popular music and address it, in line with [10, 16], using Gaussian Mixture Model (GMM)-based classifiers and Mel-frequency cepstral coefficients (MFCCs) as features. We consider this task since it is one of the MIR classification tasks for which the benefit of signal enhancement is

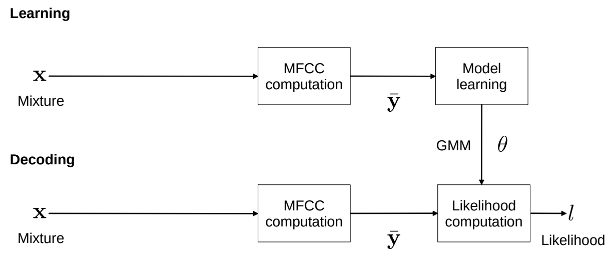


Figure 1. The standard classification scheme.

most obvious. Indeed, the information about singer identity is mostly concentrated in the singing voice signal.

The remainder of this paper is organized as follows. Some background about singer identification and baseline approaches is provided in Section 2. The proposed approach based on Gaussian uncertainty is detailed in Section 3. Experiments are presented in Section 4 and a discussion is provided in Section 5.

2. SINGER IDENTIFICATION

2.1 Background

When it comes to characterizing a song from its content, identifying the singer that is performing at a given time in a given song is arguably an interesting and useful piece of information. Indeed, most listeners have a strong commitment to the singer while listening to a given song. However, the literature about automatic singer identification is relatively scarce, compared for example with musical genre detection. This may be explained by several difficulties that pose interesting challenges for research in machine listening.

First, the human voice is a very flexible and versatile instrument and very small changes in its properties have noticable effects on human perception. Second, the musical accompaniment that forms the background is very diverse and operates at about the same loudness as the singing voice. Hence, very little can be assumed on both sides and the influence of the background cannot be neglected.

For humans, though, it is relatively easy to focus on the melody sung by the singer as our hearing system is highly skilled at segregating human vocalizations within cluttered acoustical environments. This segregation is also made possible by compositional choices. For example, most of the time in pop music, only one singer is singing at a time, and if not, the others are background vocals that are usually more easily predictable and sung at a relatively low volume.

From an application perspective, singing voice enhancement is expected to be useful for the identification of singers which have sung with different bands or with different instrumentations, such as unplugged versions. More on the so-called album effect can be found in [14]. In this case, classifying the mixture signal will induce variability in the singer models due to occlusion, while classifying the singing voice signal alone should provide better identification. The same remark applies to the case where a song features multiple singers and one needs to identify which singer is singing at a given time. For some other repertoires where the notions of singer and artist/band are very tightly linked, it is questionable whether the singing voice signal suffices for classification, because the musical background can also provide discriminative cues. Nevertheless, singing voice enhancement is likely to remain beneficial by enabling the computation of separate features over the singing voice and over the background and their fusion in the classification process. In this paper, for simplicity, we illustrate the potential of our approach by considering the singing voice signal only unless otherwise stated.

2.2 Baseline Approaches

More formally, let us assume that each recording x_{fn} (also called mixture), represented here directly in the Short Term Fourier Transform (STFT) domain, $f = 1, \dots, F$ and $n = 1, \dots, N$ being respectively frequency and time indices, is the sum of two contributions: the main melody (here the singing voice) v_{fn} and the accompaniment a_{fn}. This can be written in the following vector form:

$$\mathbf{x}_n = \mathbf{v}_n + \mathbf{a}_n, \tag{1}$$

where $\mathbf{x}_n = [x_{1n}, \dots, x_{Fn}]^T$, $\mathbf{v}_n = [v_{1n}, \dots, v_{Fn}]^T$ and $\mathbf{a}_n = [a_{1n}, \dots, a_{Fn}]^T$.

We assume that there are K singers to be recognized, and for each singer there is a sufficient amount of training and testing mixtures. In line with [10, 16], we adopt a singer identification approach based on MFCC features and GMMs.

Without any melody enhancement such an approach consists in the following two steps [13] (Fig. 1):

1. *Learning:* For each singer $k = 1, \dots, K$, the corresponding GMM model is estimated in the maximum likelihood (ML) sense from the features (here MFCCs) $\bar{\mathbf{y}}$ computed directly from the training mixtures of that singer.

2. *Decoding:* A testing mixture $\mathbf{x}$ is assigned to the singer k for which the likelihood of model θ_k evaluated on the features extracted in the same way is maximum [1].

In order to gain invariance with respect to the accompaniment, one needs to separate the contribution of the accompaniment and the singer within the mixture. This separation may be embedded within the classifier, as in [22]. In this case, the separation has to be performed in the feature domain, usually the log Mel spectrum.

Alternatively, melody enhancement can be applied as a pre-processing step [10, 16] over the spectrogram of the mixture. since the spectrogram have better spectral resolution than the log Mel spectrum, this approach can potentially achieve better discrimination, as in that case, the features (MFCCs) are no longer computed from the audio mixture, but from the corresponding melody estimate $\bar{\mathbf{v}}$ (Fig. 2).

[1] In order not to overload the notations, the singer index k is omitted hereafter, where applicable.

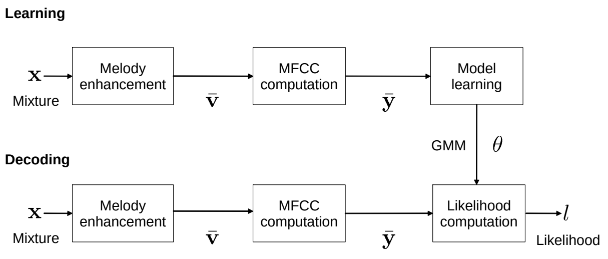


Figure 2. Considering melody enhancement as a pre-processing step.

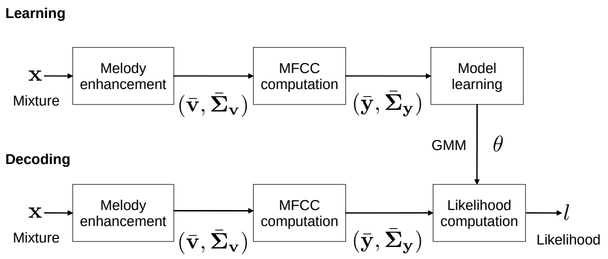

Learning
$\mathbf{x}$
Mixture
Melody enhancement
$(\bar{\mathbf{v}}, \bar{\mathbf{\Sigma}}_{\mathbf{v}})$
MFCC computation
$(\bar{\mathbf{y}}, \bar{\mathbf{\Sigma}}_{\mathbf{y}})$
Model learning
GMM
θ
Decoding
$\mathbf{x}$
Mixture
Melody enhancement
$(\bar{\mathbf{v}}, \bar{\mathbf{\Sigma}}_{\mathbf{v}})$
MFCC computation
$(\bar{\mathbf{y}}, \bar{\mathbf{\Sigma}}_{\mathbf{y}})$
Likelihood computation
l
Likelihood

Figure 3. Proposed approach with melody enhancement and Gaussian uncertainty.

3. PROPOSED APPROACH

Inspired by some approaches in speech processing [6], we propose to consider Gaussian uncertainty by augmenting the melody estimates $\bar{\mathbf{v}}$ by a set of covariance matrices $\mathbf{\Sigma_v}$ representing the errors about these estimates. This Gaussian uncertainty is first estimated in the STFT domain, then propagated through MFCC computation, and finally exploited for GMM learning and decoding steps (Fig. 3).

3.1 Melody Enhancement

Given the mixture, we assume that each STFT frame $\mathbf{v}_n$ of the melody is distributed as

$$\mathbf{v}_n|\mathbf{x}_n \sim \mathcal{N}(\bar{\mathbf{v}}_n, \bar{\mathbf{\Sigma}}_{\mathbf{v},n}), \tag{2}$$

and we are looking for an estimate of $\bar{\mathbf{v}}_n$ and $\bar{\mathbf{\Sigma}}_{\mathbf{v},n}$.

In this study, we have chosen the melody enhancement method [2] proposed by Durrieu *et al.* [7]. This method has shown very promising results for vocals enhancement task within the 2011 Signal Separation Evaluation Campaign (SiSEC 2011) [2] and its underlying probabilistic model facilitates STFT domain uncertainty computation.

The main melody $\mathbf{v}$, usually a singer, is modeled thanks to a source/filter model, and the accompaniment $\mathbf{a}$ is modeled using Non-negative Matrix Factorization (NMF) model. The leading voice is assumed to be harmonic and monophonic. The separation system mainly tracks the leading voice following two cues: first its energy, and second the smoothness of the melody line. Therefore, the resulting separated leading voice is usually the instrument or voice that is the most salient in the mixture, over certain durations of the signal. Overall this modeling falls into the framework of constrained hierarchical NMF with Itakura-Saito divergence [19], which allows a probabilistic Gaussian interpretation [9].

More precisely the method is designed for stereo mixtures. Let mixing equation

$$\underline{x}_{j,fn} = \underline{v}_{j,fn} + \underline{a}_{j,fn} \tag{3}$$

be a stereophonic version of the monophonic mixing equation (1), where $j = 1, 2$ is the channel index and equations (1) and (3) are related for any signal $\underline{s}_{j,fn}$ as

$$s_{fn} = (\underline{s}_{1,fn} + \underline{s}_{2,fn})/2. \tag{4}$$

[2] The Python source code is available at `http://www.durrieu.ch/research/jstsp2010.html`

A probabilistic Gaussian interpretation of modeling in [7] assumes $\underline{v}_{j,fn}$ and $\underline{a}_{j,fn}$ are zero-mean Gaussians that are mutually independent and independent over channel j, frequency f and time n. The corresponding constrained hierarchical NMF structured modeling allows the estimation of their respective variances $\sigma^2_{\underline{v},j,fn}$ and $\sigma^2_{\underline{a},j,fn}$ from the multichannel mixture. With these assumptions the posterior distribution of $\underline{v}_{j,fn}$ given $\underline{x}_{j,fn}$ can be shown to be Gaussian with mean

$$\bar{\underline{v}}_{j,fn} = \frac{\sigma^2_{\underline{v},j,fn}}{\sigma^2_{\underline{v},j,fn} + \sigma^2_{\underline{a},j,fn}} \underline{x}_{j,fn} \tag{5}$$

obtained by Wiener filtering, as in [7], and the variance [21]

$$\bar{\sigma}^2_{\underline{v},j,fn} = \frac{\sigma^2_{\underline{v},j,fn}\sigma^2_{\underline{a},j,fn}}{\sigma^2_{\underline{v},j,fn} + \sigma^2_{\underline{a},j,fn}}. \tag{6}$$

Finally, thanks to the posterior between-channel independence of $\underline{v}_{j,fn}$ and the down-mixing (4), $\bar{\mathbf{v}}_n$ and $\bar{\mathbf{\Sigma}}_{\mathbf{v},n}$ in (2) are computed as

$$\bar{\mathbf{v}}_n = \left\{(\bar{\underline{v}}_{1,fn} + \bar{\underline{v}}_{2,fn})/2\right\}_f, \tag{7}$$

$$\bar{\mathbf{\Sigma}}_{\mathbf{v},n} = \mathrm{diag}\left[\left\{(\bar{\sigma}^2_{\underline{v},1,fn} + \bar{\sigma}^2_{\underline{v},2,fn})/2\right\}_f\right]. \tag{8}$$

Note that any Gaussian model-based signal enhancement method, e.g., one of the methods implementable via the general source separation framework in [19], is suitable to compute this kind of uncertainty in the time-frequency domain.

3.2 Uncertainty Propagation during MFCC Computation

Let $\mathcal{M}(\cdot)$ be the nonlinear transform used to compute an M-dimensional MFCC feature vector $\mathbf{y}_n \in \mathbb{R}^M$. It can be expressed as [1]

$$\mathbf{y}_n = \mathcal{M}(\mathbf{v}_n) = \mathbf{D}\log(\mathbf{M}|\mathbf{v}_n|), \tag{9}$$

where $\mathbf{D}$ is the $M \times M$ DCT matrix, $\mathbf{M}$ is the $M \times F$ matrix containing the Mel filter coefficients, and $|\cdot|$ and $\log(\cdot)$ are both element-wise operations.

In line with (2), we assume that the clean (missing) feature $\mathbf{y}_n = \mathcal{M}(\mathbf{v}_n)$ is distributed as

$$\mathbf{y}_n|\mathbf{x}_n \sim \mathcal{N}(\bar{\mathbf{y}}_n, \bar{\mathbf{\Sigma}}_{\mathbf{y},n}), \tag{10}$$

which is an approximation because of the Gaussian assumption (2) and the nonlinear nature of $\mathcal{M}(\cdot)$.

To compute the feature estimate $\bar{\mathbf{y}}_n$ and its Gaussian uncertainty covariance $\bar{\mathbf{\Sigma}}_{\mathbf{y},n}$ we propose to use the Vector Taylor Series (VTS) method [17] that consists in linearizing the transform $\mathcal{M}(\cdot)$ by its first-order vector Taylor expansion in the neighborhood of the voice estimate $\bar{\mathbf{v}}_n$:

$$\mathbf{y}_n = \mathcal{M}(\mathbf{v}_n) \approx \mathcal{M}(\bar{\mathbf{v}}_n) + J_{\mathcal{M}}(\bar{\mathbf{v}}_n)\,(\mathbf{v}_n - \bar{\mathbf{v}}_n), \quad (11)$$

where $J_{\mathcal{M}}(\bar{\mathbf{v}}_n)$ is the Jacobian matrix of $\mathcal{M}(\mathbf{v}_n)$ computed in $\mathbf{v}_n = \bar{\mathbf{v}}_n$. This leads to the following estimates of the noisy feature value $\bar{\mathbf{y}}_n$ and its uncertainty covariance $\bar{\mathbf{\Sigma}}_{\mathbf{y},n}$ (10), as propagated through this (now linear) transform:

$$\begin{aligned} \bar{\mathbf{y}}_n &= \mathcal{M}(\bar{\mathbf{v}}_n), & (12)\\ \bar{\mathbf{\Sigma}}_{\mathbf{y},n} &= \mathbf{D}\frac{\mathbf{M}}{\mathbf{M}|\bar{\mathbf{v}}_n|\mathbf{1}_{1\times F}}\bar{\mathbf{\Sigma}}_{\mathbf{v},n}\left[\mathbf{D}\frac{\mathbf{M}}{\mathbf{M}|\bar{\mathbf{v}}_n|\mathbf{1}_{1\times F}}\right]^T & (13)\end{aligned}$$

where $\mathbf{1}_{1\times F}$ is an $1 \times F$ vector of ones and the magnitude $|\cdot|$ and the division are both element-wise operations.

3.3 GMM Decoding and Learning with Uncertainty

Each singer is modeled by a GMM $\theta = \{\boldsymbol{\mu}_i, \mathbf{\Sigma}_i, \omega_i\}_{i=1}^I$, where $i = 1, \dots, I$ are mixture component indices, and $\boldsymbol{\mu}_i$, $\mathbf{\Sigma}_i$ and ω_i ($\sum_i \omega_i = 1$) are respectively the mean, the covariance matrix and the weight of the i-th component. In other words, each clean feature vector $\mathbf{y}_n$ is modeled as follows:

$$p(\mathbf{y}_n|\theta) = \sum_{i=1}^I \omega_i N(\mathbf{y}_n|\boldsymbol{\mu}_i, \mathbf{\Sigma}_i), \quad (14)$$

where

$$N(\mathbf{y}_n|\boldsymbol{\mu}_i, \mathbf{\Sigma}_i) \triangleq \frac{1}{\sqrt{(2\pi)^M|\mathbf{\Sigma}_i|}}\left[-\frac{(\mathbf{y}_n-\boldsymbol{\mu}_i)^T\mathbf{\Sigma}_i^{-1}(\mathbf{y}_n-\boldsymbol{\mu}_i)}{2}\right]. \quad (15)$$

Since the clean feature sequence $\mathbf{y} = \{\mathbf{y}_n\}_n$ is not observed, its likelihood, given model θ, cannot be computed using (14). Thus in the “likelihood computation” step (Fig. 3), we rather compute the likelihood of the noisy features $\bar{\mathbf{y}}$ given the uncertainty and the model, that can be shown to be equal to [6]:

$$p(\bar{\mathbf{y}}|\bar{\mathbf{\Sigma}}_{\mathbf{y}}, \theta) = \prod_{n=1}^N \sum_{i=1}^I \omega_i N(\bar{\mathbf{y}}_n|\boldsymbol{\mu}_i, \mathbf{\Sigma}_i + \bar{\mathbf{\Sigma}}_{\mathbf{y},n}). \quad (16)$$

We see that in this likelihood Gaussian uncertainty covariance $\bar{\mathbf{\Sigma}}_{\mathbf{y},n}$ adds to the prior GMM covariance $\mathbf{\Sigma}_i$, thus adaptively decreasing the effect of signal distortion.

In the “model learning” step (Fig. 3), we propose to estimate the GMM parameters θ by maximizing the likelihood (16). This can be achieved via the iterative Expectation-Maximization (EM) algorithm introduced in [18] and summarized in Algorithm 1. The derivation of this algorithm is omitted here due to lack of space and the Matlab source code for GMM decoding and learning is available at `http://bass-db.gforge.inria.fr/amulet`.

Algorithm 1 One iteration of the EM algorithm for the likelihood integration-based GMM learning from noisy data.

E step. Conditional expectations of natural statistics:

$$\gamma_{i,n} \propto \omega_i N(\bar{\mathbf{y}}_n|\boldsymbol{\mu}_i, \mathbf{\Sigma}_i + \bar{\mathbf{\Sigma}}_{\mathbf{y},n}), \quad \text{and} \quad \sum_i \gamma_{i,n} = 1, \quad (17)$$

$$\hat{\mathbf{y}}_{i,n} = \mathbf{W}_{i,n}\,(\bar{\mathbf{y}}_n - \boldsymbol{\mu}_i) + \boldsymbol{\mu}_i, \quad (18)$$

$$\widehat{\mathbf{R}}_{\mathbf{yy},i,n} = \hat{\mathbf{y}}_{i,n}\hat{\mathbf{y}}_{i,n}^T + (\mathbf{I} - \mathbf{W}_{i,n})\,\mathbf{\Sigma}_i, \quad (19)$$

where

$$\mathbf{W}_{i,n} = \mathbf{\Sigma}_i\left[\mathbf{\Sigma}_i + \bar{\mathbf{\Sigma}}_{\mathbf{y},n}\right]^{-1}. \quad (20)$$

M step. Update GMM parameters:

$$\omega_i = \frac{1}{N}\sum_{n=1}^N \gamma_{i,n}, \quad (21)$$

$$\boldsymbol{\mu}_i = \frac{1}{\sum_{n=1}^N \gamma_{i,n}}\sum_{n=1}^N \gamma_{i,n}\hat{\mathbf{y}}_{i,n}, \quad (22)$$

$$\mathbf{\Sigma}_i = \frac{1}{\sum_{n=1}^N \gamma_{i,n}}\sum_{n=1}^N \gamma_{i,n}\widehat{\mathbf{R}}_{\mathbf{yy},i,n} - \boldsymbol{\mu}_i\boldsymbol{\mu}_i^T. \quad (23)$$

4. EXPERIMENTS

4.1 Database

For our evaluation, we consider a subset of the RWC Popular Music Database [11] which has previously been considered in [10] for the same task. It consists of 40 songs sung by 10 singers, five of which were male (denoted by a to e) and the five others female (denoted by f to j). This set is then divided into the four groups of songs considered in [10], each containing one song by each singer .

Each of those songs is then split into segments of 10 seconds duration. Among those segments, only the ones where a singing voice is present (not necessarily during the whole duration of the segment) are kept unless otherwise stated.

Considering short duration segments instead of the whole song is done for two reasons. First, it makes the task more generic in the sense that multiple singers can also potentially be tracked within a same song. Second, it allows us to gain statistical relevance during the cross validation by enlarging the number of tests.

4.2 Methods

For each of those segments, features are computed and classified using the three methods depicted in Figures 1 to 3. The first one, named *mix*, consists in computing the features directly from the mixture, and serves as a baseline. The second method, termed *v-sep*, consider melody enhancement as a pre-processing step. The main melody enhancement system considered in this study is available under two versions: a version focusing on the voiced part of

Accuracy (%)	per 10 sec. singing segment					per song	
input	Fold 1	Fold 2	Fold 3	Fold 4	Total	all seg.	sung seg.
mix	51	53	55	38	49	57	64
v-sep	60	63	53	43	55	57	64
v-sep-uncrt	**71**	**72**	**84**	**83**	**77**	**85**	**94**

Table 1. Average accuracy of the tested methods per singing segment and per song, considering either all the segments or only those segments where singing voice is present in the latter case.

the singing voice and another version attempting to jointly enhance the voiced and the unvoiced parts of the singing voice (see [7] for details). In the following, only the results of the former are reported since the latter led to much smaller classification accuracy. When the estimated vocals signal has zero power in a given time frame, the resulting MFCCs may be undefined. Such frames are discarded. The last method, termed *v-sep-uncrt*, consists in exploiting the estimated uncertainty about the enhancement process.

For all those methods, we considered MFCC features and dropped the first coefficient, thus discarding energy information. Mixtures of 32 Gaussians are then trained using 50 iterations of the EM algorithm for each singer. For testing, the likelihood of each singer model is computed for each segment and the one with the highest likelihood is selected as the estimate.

4.3 Results

The aforementioned four groups of songs are considered for a 4-fold cross validation. For each fold, the selected group is used for testing and the data of the three remaining ones are used for training the models. The average detection accuracy are shown in Table 1. Compared to the baseline, *v-sep* and *v-sep-uncrt* achieve better performance while considering segments, indicating that focusing on the main harmonic source within the segment is beneficial for identifying the singer. That is, the level of feature invariance gained by the separation process more than compensates for the distortions it induces.

Considering the uncertainty estimate adds a significant level of improvement in the $v - sep$ case. We assume that this gain of performance is obtained because the use of uncertainty allows us to focus on the energy within the spectrogram that effectively belongs to the voice and that the use of the uncertainty allows us to robustly consider standard features (MFCCs).

Performing a majority vote over the all the segments (in this case the likelihood of each singer is taken into account even if no singing voice is present) of each song gives an accuracy of 85% and restricting the vote to only the sung segments gives a 94% accuracy. These numbers can respectively be considered as worst and best cases. It is therefore likely that a complete system that would incorporate a music model to discard segments with only music would achieve an accuracy that is between those bounds. Although a more formal comparison would be needed, we believe that those results compare favorably with the performances obtained in [10] using specialized features on the same dataset while standard MFCC features were used here. It is also interesting to notice that in this case of song-level decisions, considering the separation without uncertainty does not give any improvement compared to the *mix* baseline.

5. DISCUSSION

We have presented in this paper a computational scheme for extracting meaningful information in order to tackle a music retrieval task: singer identification. This is done by considering an enhanced version of the main melody that is more or less reliable in specific regions of the time/frequency plane. Instead of blindly making use of this estimate, we propose in this paper to consider how uncertain the separation estimate is during the modeling phase. This allows us to give more or less importance to the features depending on how reliable they are in different time frames, both during the training and the testing phases. For that purpose, we adopted the Gaussian uncertainty framework and introduced new methods to estimate the uncertainty in a fully automatic manner and to learn GMM classifiers directly from polyphonic data.

One should notice that the proposed scheme is not tied to the task considered in this paper. It is in fact completely generic and may be easily applied to other GMM-based MIR classification tasks where the prior isolation of a specific part of the music signal could be beneficial. The only part that would require adaptation is the derivation of VTS uncertainty propagation equations for other features than MFCCs. Uncertainty handling for other classifiers than GMM has also received some interest recently in the speech processing community.

The experiments reported in this paper provide us with encouraging results. Concerning this specific task of singer identification, we intend to exploit both the enhanced singing voice and accompaniment signals and to experiment on other datasets with a wider range of musical styles. In particular, we believe that the hip-hop/rap musicals genres would be an excellent testbed both from a methodological and application point of view, as many songs feature several singers: knowing which singer is performing at a given time is a useful piece of information. Finally, we would like to consider other content based retrieval tasks in order to study the relevance of this scheme for a wider range of applications.

6. ACKNOWLEDGMENTS

This work was partly supported by the Quaero project funded by Oseo and ANR-11-JS03- 005-01. The authors wish to thank Jean-Louis Durrieu for kindly providing his melody estimation source code to the community and Mathias Rossignol for his useful comments.

7. REFERENCES

[1] K. Adiloğlu and E. Vincent. An uncertainty estimation approach for the extraction of individual source features in multisource recordings. In *EUSIPCO, 19th European Signal Processing Conference*, 2011.

[2] S. Araki, F. Nesta, E. Vincent, Z. Koldovsky, G. Nolte, A. Ziehe, and A. Benichoux. The 2011 signal separation evaluation campaign (SiSEC2011): - audio source separation. In *Proc. Int. Conf. on Latent Variable Analysis and Signal Separation*, pages 414–422, 2012.

[3] 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), 2004.

[4] P. Chordia and A. Rae. Using source separation to improve tempo detection. In *Proc. 10th Intl. Society for Music Information Retrieval Conference*, pages 183–188, Kobe, Japan, 2009.

[5] M Cooke. Robust automatic speech recognition with missing and unreliable acoustic data. *Speech Communication*, 34(3):267–285, June 2001.

[6] L. Deng, J. Droppo, and A. Acero. Dynamic compensation of HMM variances using the feature enhancement uncertainty computed from a parametric model of speech distortion. *IEEE Transactions on Speech and Audio Processing*, 13(3):412–421, 2005.

[7] 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.

[8] J. Eggink and G. J. Brown. Application of missing feature theory to the recognition of musical instruments in polyphonic audio. In *Proc. 4th International Conference on Music Information Retrieval*, 2003.

[9] C. Févotte, N. Bertin, and J.-L. Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence. With application to music analysis. *Neural Computation*, 21(3):793–830, Mar. 2009.

[10] H. Fujihara, M. Goto, T. Kitahara, and H. G. Okuno. A modeling of singing voice robust to accompaniment sounds and its application to singer music information retrieval. *IEEE Trans. Audio, Speech and Language Processing*, 18(3):638–648, 2010.

[11] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: popular, classical, and jazz music databases. *Proc. International Conference for Music Information Retrieval (ISMIR)*, pages 287–288, 2003.

[12] T. Heittola, A. Klapuri, and T. Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. *in Proc. 10th Intl. Society for Music Information Retrieval Conference, Kobe, Japan*, pages 327–332, 2009.

[13] Y. E. Kim. Singer identification in popular music recordings using voice coding features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.

[14] Y. E. Kim, D. S. Williamson, and S. Pilli. Towards quantifying the album effect in artist identification. In *Proc. of Int.Conf. on Music Information Retrieval (ISMIR)*, pages 393–394, 2006.

[15] 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 Advances in Signal Processing*, 2007, 2007. article ID 51979.

[16] A. Mesaros and T. Virtanen. Singer identification in polyphonic music using vocal separation and pattern recognition methods. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 375–378, 2007.

[17] P. J. Moreno, B. Raj, and R. M. Stern. A vector Taylor series approach for environment-independent speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'96)*, volume 2, pages 733 – 736, 1996.

[18] A. Ozerov, M. Lagrange, and E. Vincent. GMM-based classification from noisy features. In *Proc. 1st Int. Workshop on Machine Listening in Multisource Environments (CHiME)*, pages 30–35, Florence, Italy, September 2011.

[19] A. Ozerov, E. Vincent, and F. Bimbot. A general flexible framework for the handling of prior information in audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 20(4):1118 – 1133, 2012.

[20] J. Reed, Y. Ueda, S. M. Siniscalchi, Y. Uchiyama, S. Sagayama, and C. H. Lee. Minimum classification error training to improve isolated chord recognition. In *Proc. 10th Intl. Society for Music Information Retrieval Conference*, pages 609–614, Kobe, Japan, 2009.

[21] R. C. Rose, E. M. Hofstetter, and D. A. Reynolds. Integrated models of signal and background with application to speaker identification in noise. *IEEE Trans. Speech and Audio Processing*, 2(2):245–257, April 1994.

[22] W.H. Tsai and H.M. Wang. Automatic singer recognition of popular music recordings via estimation and modeling of solo vocal signals. *IEEE Transactions on Audio Speech and Language Processing*, pages 1–35, 2006.

PREDOMINANT FUNDAMENTAL FREQUENCY ESTIMATION VS SINGING VOICE SEPARATION FOR THE AUTOMATIC TRANSCRIPTION OF ACCOMPANIED FLAMENCO SINGING

E. Gómez[1], F. Cañadas[2], J. Salamon[1], J. Bonada[1], P. Vera[2] and P. Cabañas[2]
[1] Music Technology Group, Universitat Pompeu Fabra, Spain
[2] Telecommunication Engineering Department, University of Jaen, Spain
emilia.gomez@upf.edu, fcanadas@ujaen.es, justin.salamon@upf.edu, jordi.bonada@upf.edu, pvera@ujaen.es, pcabanas@ujaen.es

ABSTRACT

This work evaluates two strategies for predominant fundamental frequency (f_0) estimation in the context of melodic transcription from flamenco singing with guitar accompaniment. The first strategy extracts the f_0 from salient pitch contours computed from the mixed spectrum; the second separates the voice from the guitar and then performs monophonic f_0 estimation. We integrate both approaches with an automatic transcription system, which first estimates the tuning frequency and then implements an iterative strategy for note segmentation and labeling. We evaluate them on a flamenco music collection, including a wide range of singers and recording conditions. Both strategies achieve satisfying results. The separation-based approach yields a good overall accuracy (76.81%), although instrumental segments have to be manually located. The predominant f_0 estimator yields slightly higher accuracy (79.72%) but does not require any manual annotation. Furthermore, its accuracy increases (84.68%) if we adapt some algorithm parameters to each analyzed excerpt. Most transcription errors are due to incorrect f_0 estimations (typically octave and voicing errors in strong presence of guitar) and incorrect note segmentation in highly ornamented sections. Our study confirms the difficulty of transcribing flamenco singing and the need for repertoire-specific and assisted algorithms for improving state-of-the-art methods.

1. INTRODUCTION

Flamenco is a music tradition originating mostly from Andalusia in southern Spain. The singer has a main role and is often accompanied by the guitar and other instruments such as claps, rhythmic feet and percussion. This research aims to develop a method for computing detailed note transcriptions of flamenco singing from music recordings, which can then be processed for motive analysis or further simplified to obtain an overall melodic contour that will characterize the style. In this study we focus on accompanied singing, and propose a method comprised of two stages: predominant f_0 estimation and note segmentation. For the first stage, two alternative strategies are evaluated and compared: in the first, we use a state-of-the-art predominant f_0 estimation algorithm, which estimates the f_0 of the predominant melody directly from the full audio mix. In the second, we propose a source separation approach to isolate the singing voice and perform monophonic f_0 estimation.

2. SCIENTIFIC BACKGROUND

Automatic transcription is a key challenge in the music information retrieval (MIR) field. It consists of computing a symbolic musical representation from an audio recording. In polyphonic music material, there is an interest in transcribing the predominant melodic line [1]. Although we find some successful approaches for singing transcription [2,3], the singing voice is still one of the most complex instruments to transcribe, given the continuous character of the human voice and the variety of pitch ranges and timbre. Additional challenges in flamenco arise from the quality of recordings, the acoustic and expressive particularities of flamenco singing, its ornamental and improvisational character and the yet to be formalized musical structures [4].

In [5], we proposed a melodic transcription system from a cappella flamenco singing and we evaluated it against manual annotations of 72 performances. The obtained overall accuracy was around 70% (50 cents tolerance), which was significantly lower than the one obtained for a small test collection of pop/jazz excerpts (∼85%). The study showed the importance of good monophonic f_0 estimation, and confirmed the difficulty of note segmentation for excerpts with unstable tuning or highly ornamented sections.

The goals of the present study are to apply this transcription system to accompanied singing and to perform a comparative evaluation of two alternative strategies for singing voice f_0 estimation. The first is to replace the monophonic f_0 detector by a predominant f_0 estimation method. The task of predominant f_0 estimation from polyphonic music (sometimes referred to simply as melody extraction) has received much attention from the research community in recent years, and state-of-the-art approaches yield an overall accuracy around 75% [6, 7]. A variety of different approaches have been proposed, based on tracking agents [8], classification [9], streaming rules [10] or pitch contour characterization [11]. The most common set of approaches are "salience based", i.e. they compute a pitch salience representation from the audio signal, and then select the melody out of the peaks of this representation over time [8, 9, 11].

The second strategy is to separate the singing voice from the guitar accompaniment using source separation and transcribe the separated track. Recent singing voice separation methods can be classified into three categories: spectrogram factorization [12–14], pitch-based inference [15, 16] and repeating-structure removal [17]. Spectrogram factorization methods decompose a magnitude spectrogram as a set of components that represent features such as the spectral patterns (basis) or the activations (gains) of the active sources along the time. Fitzgerald and Gainza [12] propose a non-negative partial cofactorisation sharing a common set of frequency basis functions. In [13], an accompaniment model is designed, from the non-vocal segments, to fit the musical instruments and attempt separation of the vocals. In [14], the basis of the vocal track is learned from the mixture by keeping the accompaniment spectra fixed. Pitch-based inference methods use information from the pitch contour to determine the harmonic structures of singing voice. In [16], separation of both the voiced and the unvoiced singing voice is presented by means of the combination of detected unvoiced sounds and a spectral subtraction method to enhance voiced singing separation [15]. Repeating-structure removal methods [17] use a pattern recognition approach to identify and extract accompaniment segments, without manual labeling, which can be classified as repeating musical structures.

3. TRANSCRIPTION METHOD

Our method relies on two main stages: low-level feature extraction (mainly f_0) and note segmentation. We present the two alternatives for f_0 estimation compared in this study followed by a summary of the note segmentation approach.

3.1 Singing voice f_0 estimation

3.1.1 Predominant f_0 estimation

For predominant f_0 estimation, we use [11], which obtained the highest overall accuracy in MIREX 2011 [6]. First, the audio signal is analyzed and spectral peaks (sinusoids) are extracted. This process is comprised of three main steps: first a time-domain equal loudness filter is applied, which has been shown to attenuate spectral components belonging primarily to non-melody sources [19]. Next, the short-time Fourier transform is computed with a 46 ms Hann window, a hop size of 2.9 ms and a 4 zero padding factor. At each frame the local maxima (peaks) of the spectrum are detected. In the third step, the estimation of the spectral peaks' frequency and amplitude is refined by calculating each peak's instantaneous frequency (IF) using the phase vocoder method and re-estimating its amplitude based on the IF. The detected spectral peaks are subsequently used to compute a representation of pitch salience over time: a salience function. The salience function is based on harmonic summation with magnitude weighting, and spans a 5-octave range from 55Hz to 1760Hz. Details are provided in [11]. In the next stage, the peaks of the salience function are grouped over time using heuristics based on auditory streaming cues. This results in a set of pitch contours, out of which the contours belonging to the melody need to be selected. The contours are automatically analyzed and a set of contour characteristics is computed. In the final stage of the system, the contour characteristics and their distributions are used to filter out non-melody contours. The distribution of contour salience is used to filter out pitch contours at segments of the song where the melody is not present. Next, we obtain a rough estimate of the melodic pitch trajectory by computing a per-frame salience-weighted average of the remaining pitch contours and smoothing it over time using a sliding mean filter. This rough pitch trajectory is used to minimise octave errors (contours with the correct pitch class but in the wrong octave) and remove pitch outliers (contours representing highly unlikely jumps in the melody). Finally, the melody f_0 at each frame is selected out of the remaining pitch contours based on their salience. For further details the reader is referred to [11].

In addition to computing the melody f_0 sequence using the default algorithm parameters (denoted *MTG*), we also computed the melody adjusting three parameters of the algorithm for each musical excerpt: the minimum and maximum frequency threshold and the strictness of the voicing filter (cf. [11] for details). The results using the per-excerpt adjusted parameters are referred to as *MTGAdaptedparam*.

3.1.2 Singing voice separation and monophonic f_0 estimation

Standard Non-negative Matrix Factorization (NMF) [20] is not able to determine if a frequency basis belongs to a percussive, harmonic or vocal sound. Our proposal attempts to overcome this limitation without using any clustering process. A mixture spectrogram X is factorized into three separated spectrograms, X_p (percussive), X_h (harmonic) and X_v (vocal). Using similar spectro-temporal features [21, 22], harmonic sounds are modeled by sparseness in frequency and smoothness in time. Percussive sounds are modeled by smoothness in frequency and sparseness in time. Vocal sounds are modeled by sparseness in frequency and sparseness in time. Although it is not necessary to discriminate between percussive and harmonic sounds in the accompaniment, our experimental results showed we obtain better vocal separation using this discrimination. The proposed singing voice separation is composed of three stages: segmentation, training and separation.

In the segmentation stage, the mixture signal $X = X_{nonvocal} \bigcup X_{vocal}$ is manually labelled into vocal X_{vocal} (vocal+instruments) and non-vocal $X_{nonvocal}$ (only instruments) regions. In the training stage, from non-vocal regions, the percussive W_p and harmonic W_h basis vectors are learned using an unsupervised NMF percussive/harmonic separation approach based on spectro-temporal features.

$$X_{nonvocal} \approx X_p + X_h = W_p \cdot H_p + W_h \cdot H_h \qquad (1)$$

In the separation stage, the vocal spectrogram X_v is extracted from the vocal regions by keeping the percussive

W_p and harmonic W_h basis vectors fixed from the previous stage.

$$X_{vocal} \approx X'_p + X'_h + X_v = W_p \cdot H'_p + W_h \cdot H'_h + W_v \cdot H_v \quad (2)$$

In this manner, the singing voice signal $v(t)$ is synthesized from the vocal spectrogram X_v. To obtain an f_0 sequence from the synthesized voice signal, the traditional difference function is computed for each time frame index t:

$$d(\tau, t) = \sum_{n=0}^{W-1} (v(t+n) - v(t+n+\tau))^2 \quad (3)$$

where W is the length of the summation window and τ is the candidate pitch period. From this function, the cumulative mean normalized difference function can be computed as defined in [23]:

$$d_n(\tau, t) = \begin{cases} 1, & \tau = 0 \\ d(\tau, t) / [\frac{1}{\tau} \sum_{j=1}^{\tau} d(j, t)] & \text{otherwise.} \end{cases} \quad (4)$$

Observe that the function $d_n(\tau, t)$ can be viewed as a cost matrix, where each element (τ, t) indicates the cost of having a pitch period equal to τ at time frame t. We estimate the whole f_0 sequence by computing the lowest-cost path through the matrix $d_n(\tau, t)$. This computation is accomplished with dynamic programming. The endpoints of the path are fixed only for the t-axis and the path is constrained to advance step-by-step along t, under the condition $|\tau_{t-1} - \tau_t| \leq 1$. This condition ensures a continuous and smooth f_0 contour. The obtained f_0 is denoted as *UJA*.

3.2 Note segmentation and labeling

Our approach for note segmentation and labeling is adapted from a transcription system for mainstream popular music [18]. After consulting a group of flamenco experts from the COFLA project [1], we took the following design decisions. First, we define an equal-tempered scale with respect to an estimated tuning frequency. Second, we assume a constant tuning frequency value for each analyzed excerpt. Third, we transcribe all perceptible notes, including short ornamentations, in order to cover both expressive nuances and the overall melodic contour. We summarize below the mains steps of the transcription algorithm and we refer to [5] and [18] for further details.

3.2.1 Tuning frequency estimation

From the obtained f_0 envelope, we perform an estimation of the tuning frequency used by the singer assuming an equal-tempered scale. The tuning frequency is assumed to be constant for a given excerpt. We compute the maximum of the histogram of f_0 deviations from an equal-tempered scale tuned to 440 Hz. We then map the f_0 values of all frames into a single semitone interval with a one-cent resolution.

In our approach, we give more weight to frames where the included f_0 is stable by assigning higher weights to

[1] http://mtg.upf.edu/research/projects/cofla

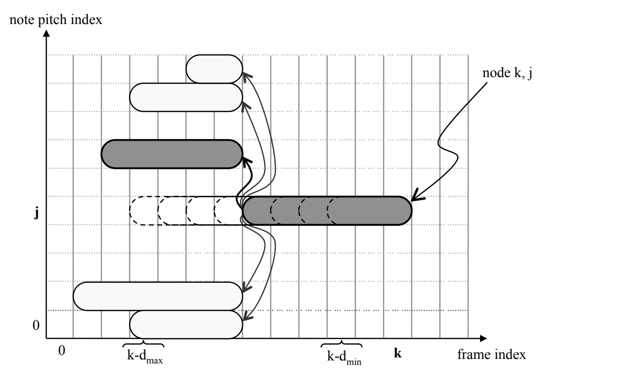


Figure 1. Matrix M used by the short note segmentation process, illustrating how the best path for a node with frame index k and note index j is determined. All possible note durations between d_{min} and d_{max} are considered, as well as all possible jumps to previous notes. The selected segmentation is marked with dark gray.

frames with low f_0 derivative. In order to smooth the resulting histogram and improve its robustness to noisy f_0 estimations, instead of adding a value to a single bin, we use a bell-shaped window that spans several bins. The maximum of this histogram (b_{max}) determines the tuning frequency deviation in cents from 440 Hz. The estimated tuning frequency in Hz then becomes

$$f_{ref} = 440 \cdot 2^{\frac{b_{max}}{1200}} \quad (5)$$

3.2.2 Short note transcription

The short note transcription step segments a single f_0 contour into notes. Using dynamic programming (DP), we find the note segmentation that maximizes a set of probability functions. The estimated segmentation corresponds to the optimal path among all possible paths along a 2-D matrix M (see Figure 1).

This matrix M has note pitches as rows and analysis frames as columns. Note pitches are quantized into semitones according to the estimated tuning frequency. Possible note pitches should cover the tessitura of the singer and include a $-\infty$ value for the unvoiced sections. Note durations are limited to a certain range $[d_{min}, d_{max}]$ of frames. The maximum duration d_{max} should be long enough so that it covers several periods of a vibrato with a low modulation frequency, e.g. $2.5Hz$, but also short enough to have good temporal resolution, e.g. avoid skipping short ornamentations.

Possible paths considered by the DP algorithm always start from the first frame, end at the last audio frame, and advance in time so that notes never overlap. A path p is defined by its sequence of N_p notes,
$p = \{n_{p0}, n_{p1}, \ldots, n_{pN_p-1}\}$, where each note n_{pi} begins at a certain frame k_{pi}, has a duration of d_{pi} frames and a pitch value of c_{pi}. The optimal path is defined as the path with maximum likelihood among all possible paths.

$$P = \arg\max_{p}\{L(p)\} \tag{6}$$

The likelihood $L(p)$ of a certain path p is determined as the product of likelihoods of each note $L(n_{pi})$ times the likelihood of each jump between consecutive notes $L(n_{pi-1}, n_{pi})$:

$$L(p) = L(n_{p0}) \cdot \prod_{i=1}^{N_{p-1}} L(n_{pi}) \cdot L(n_{pi-1}, n_{pi}) \tag{7}$$

In our approach, no particular characteristic is assumed a priori for the sung melody; therefore all possible note jumps have the same likelihood $L(n_{pi-1}, n_{pi}) = 1$. On the other hand, the likelihood of a note $L(n_{pi})$ is determined as the product of several likelihood functions based on the following criteria: duration (L_d), pitch (L_c), existence of voiced and unvoiced frames (L_v), and low-level features related to stability (L_s):

$$L(n_{pi}) = L_d(n_{pi}) \cdot L_c(n_{pi}) \cdot L_v(n_{pi}) \cdot L_s(n_{pi}) \tag{8}$$

Duration likelihood L_d is set so that it is small for short and long durations. Pitch likelihood L_c is defined so that it is higher the closer the frame f_0 values are to the note nominal pitch c_{pi}, giving more relevance to frames with low f_0 derivative values. The voicing likelihood L_v is defined so that segments with a high percentage of unvoiced frames are unlikely to be a voiced note, while segments with a high percentage of voiced frames are unlikely to be an unvoiced note. Finally, the stability likelihood L_s considers that a voiced note is unlikely to have fast and significant timbre or energy changes in the middle. Note that this is not in contradiction with smooth vowel changes, characteristic of flamenco singing.

3.2.3 *Iterative note consolidation and tuning frequency refinement*

The notes obtained in the previous step have a limited duration between $[d_{min}, d_{max}]$ frames, although longer notes are likely to have been sung. Therefore, it makes sense to consolidate consecutive voiced notes into longer notes if they have the same pitch. However, significant and fast energy or timbre changes around the note connection boundary may be indicative of phonetic changes unlikely to happen within a note, and thus may indicate that those consecutive notes are different ones. Thus, consecutive notes will be consolidated only if they have the same pitch and the stability measure of their connection L_s falls below a certain threshold.

Once notes are consolidated, it may be beneficial to use the note segmentation to refine the tuning frequency estimation. For this purpose, we compute a pitch deviation for each voiced note, and then estimate a new tuning frequency value from a one-semitone histogram of weighted note pitch deviations in similar way to that described in Section 3.2.1. The difference is that now we add a value for each voiced note instead of for each voiced frame. Weights are determined as a measure of the salience of each note, giving more weight to longer and louder notes. As a final step of this process, note nominal pitches are re-computed based on the new tuning frequency. This process is repeated until there are no more consolidations.

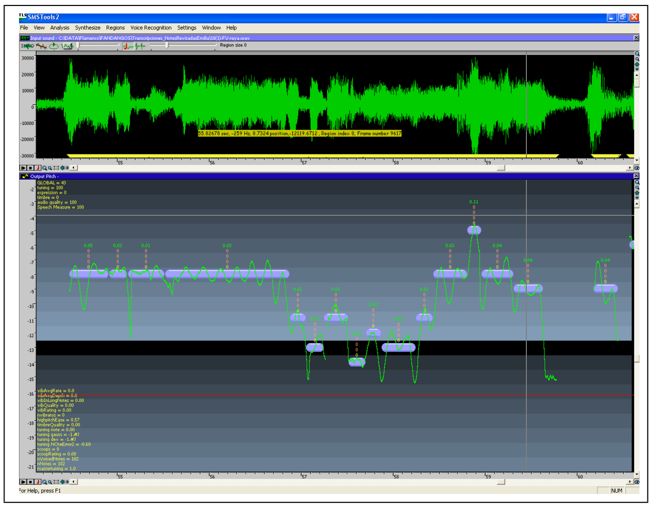


Figure 2. Visualization tool for melodic transcription. Audio waveform (top), estimated f_0 and pitch in a piano roll representation (bottom).

Figure 2 shows an example of a computed transcription. The system outputs both the extracted f_0 envelope and the estimated frame note pitch, according to an equal-tempered scale, as requested by flamenco experts for higher-level analyses.

4. EVALUATION STRATEGY

4.1 Music collection

We gathered 26.74 minutes of music, consisting of 30 performances of singing voice with guitar accompaniment (Fandango style). This collection has been built in the context of the COFLA project. It contains a variety of male and female singers and recording conditions. The average duration of the analyzed excerpts is 53.48 seconds and they contain a total of 271482 frames and 2392 notes.

4.2 Ground truth gathering

We collected manual note annotations from a musician with limited knowledge of flamenco music, so that there was no implicit knowledge applied in the transcription process. We provided him with the user interface shown in Figure 2. Since transcribing everything from scratch is very time consuming, we also provided the output of our transcription using the *MTGAdaptedParam* estimation as a guide. The annotator could listen to the original waveform and the synthesized transcription, while editing the melodic data until he was satisfied with the transcription. The criteria used to differentiate ornaments and pitch glides were discussed with two flamenco experts by collectively annotating a set of working examples, so that the annotator then followed a well-defined and consistent strategy.

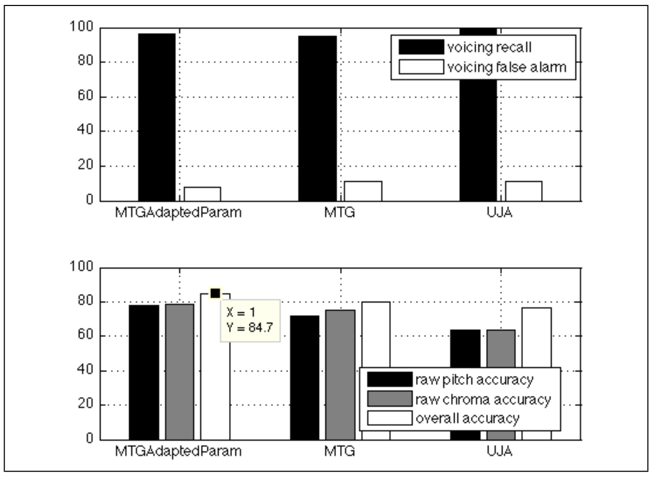


Figure 3. Frame-based accuracy measures (50 cents tolerance) for the considered approaches.

4.3 Evaluation measures

For evaluation we compute the measures used in the Audio Melody Extraction (AME) MIREX task [6]. The measures are based on a frame-by-frame comparison of the ground-truth to the estimated frequency sequence. Note that in our case we compare the ground truth to the frequency of the final note transcription, meaning any observed errors represent the combined errors introduced by the two stages of our method (f_0 estimation and note segmentation). Also, since we do not provide a pitch estimate for frames determined as unvoiced, incorrect voicing detection will also influence pitch accuracy (but not overall accuracy). We consider *Voicing recall*: % of voiced frames in the reference correctly estimated as voiced; *Voicing false alarm*: % of unvoiced frames in the reference mistakenly estimated as voiced; *Raw pitch accuracy*: % of voiced frames where the pitch estimate is correct within a certain threshold in cents (th); *Raw chroma accuracy*: same as the raw pitch accuracy except that octave errors are ignored; and *Overall accuracy*: total % of correctly estimated frames: correct pitch for voiced frames and correct labeling of unvoiced frames.

5. RESULTS

5.1 Frame-based pitch accuracy

Figure 3 shows the obtained accuracy measures for $th = 50$ cents. At first glance, we see that satisfying results are obtained for both strategies. The separation-based approach (*UJA*) yields good results (overall accuracy 76.81%, pitch accuracy 63.62%), as the guitar timbre can be accurately estimated from the instrumental segments. Nevertheless, these guitar segments have to be manually located. The predominant f_0 estimator (*MTG*) yields slightly higher overall accuracy (79.72%) and pitch accuracy (71.46%), and it does not require manual voicing annotation. Moreover, the overall accuracy increases to 84.68% (pitch accuracy 77.92%) if we adapt some algorithm parameters for each excerpt (*MTGAdaptedParam*). The observed voicing false alarm rate (around 10% for MTG and UJA) results from segments where the guitar is detected as melody.

Est.	Ref.	Vx Recall	Vx False Alarm	Raw pitch	Raw chroma	Overall accuracy
MTG	UJA	89.24	6.35	74.20	74.82	82.67
UJA	MTG	94.00	12.95	78.29	78.93	82.65

Table 1. Accuracy measures between f_0 estimations.

The obtained results are slightly higher than the ones obtained for a cappella singing [5] when considering the same note segmentation algorithm together with a monophonic f_0 estimator. This is due to two main reasons. Primarily, as the singer follows the tuning reference of the guitar, there are no tuning errors and the note labeling results are improved. Also, as the voice is very predominant with respect to the guitar, the predominant f_0 estimation method works very well for this material.

5.2 Agreement between f_0 estimations

We also estimate the agreement between both f_0 strategies by computing the evaluation measures with one estimator as ground truth and the other one as estimation. Results are presented in Table 1. We observe that in both cases the overall agreement is around 82.5%. The main difference between the approaches is in the determination of voiced sections. Whilst in *UJA* only large non-voiced sections were manually annotated, *MTGAdaptedParam* also attempts to automatically detect shorter unvoiced sections in the middle of the piece.

5.3 Error analysis

We observe that for the two considered strategies, transcription errors are introduced in both stages of the transcription process (f_0 estimation and note segmentation).

Regarding singing voice f_0 estimation, voicing seems to be the main aspect to improve. Voicing false positives occasionally appear during melodic guitar segments and in short unvoiced phonemes (e.g. fricatives). On the other hand, the singing voice f_0 is sometimes missed in the presence of strong instrumental accompaniment, resulting in voicing false negatives. Since the subsequent note segmentation stage relies on the voicing estimation, voicing errors during the f_0 estimation are bound to introduce errors in the note segmentation stage as well. Another type of error is fifth or octave errors at segments with highly predominant accompaniment. This occurs especially with the *UJA* method, as low harmonics of the singing voice might be erased from the spectrum during the separation process.

Regarding the note segmentation algorithm, most of the errors happen for short notes; either they are consolidated while the annotation consists of several close notes, or vice versa. This especially happens where the energy envelope also accounts for the presence of guitar, so that onset estimation becomes more difficult. Finally, some of the errors occur due to wrong pitch labeling of very short notes, as the f_0 contour is short and unstable. This demonstrates the difficulty of obtaining accurate note transcriptions for flamenco singing, given its ornamental character and the continuous variations of f_0, easily confused with deep vibrato or pitch glides. The great variability of the vocal f_0 contour can be observed in Figure 2.

6. CONCLUSIONS

This paper presents an approach for computer-assisted transcription of accompanied flamenco singing. It is based on an iterative note segmentation and labelling technique from f_0, energy and timbre. Two different strategies for singing voice f_0 estimation were evaluated on 30 minutes of flamenco music, obtaining promising results which are comparable to (and even better than) previous results for monophonic singing transcription. The main sources of transcription errors were identified: in the first stage (f_0 estimation) the main issue is voicing detection (e.g. identification of the guitar as voice), though we occasionally observe pitch errors (e.g. wrong f_0 in the presence of guitar) as well. In the second stage (note segmentation) we observed errors in segmenting short notes and labeling notes with an unstable f_0 contour. There is still much room for improvement. One limitation of this work is the small amount of manual annotations. This is due to the fact that manual annotation is very time consuming and difficult to obtain, and has a degree of subjectivity. We are currently expanding the amount of manual annotations. The second limitation is that we only have manual annotations on a note level (quantized to 12 semitones) and not the continuous f_0 ground truth, which would allow us to evaluate separately the accuracy of the two main stages of the algorithm. We plan to work on this issue. Finally, we plan to quantify the uncertainty of the ground truth information by comparing annotations in different contexts, and adapt the algorithm parameters accordingly.

7. ACKNOWLEDGEMENTS

The authors would like to thank the COFLA[1] team for providing the data set and expert knowledge in flamenco music. This work has been partially funded by AGAUR (mobility grant), the COFLA project (P09-TIC-4840 Proyecto de Excelencia, Junta de Andalucía) and the Programa de Formación del Profesorado Universitario of the Ministerio de Educación de España.

8. REFERENCES

[1] A. Klapuri, and M. Davy (Eds): "Signal Processing Methods for Music Transcription," Springer-Verlag, 2006.

[2] T. Mulder, J. P. Martens, M. Lesaffre, M. Leman, B. De Baets, and H. De Meyer: "An Auditory Model Based Transcriber of Vocal Queries," Proc. of ISMIR, 2003.

[3] M. P. Ryynänen: "Singing transcription," in Signal processing methods for music transcription (A. Klapuri and M. Davy, eds.), Springer, 2006.

[4] J. Mora, F. Gomez, E. Gómez, F. Escobar-Borrego, and J.M. Diaz-Bañez: "Characterization and melodic similarity of a Cappella flamenco cantes," Proc. of ISMIR 2010.

[5] E. Gómez, J. Bonada, and J. Salamon: "Automatic Transcription of Flamenco Singing from Monophonic and Polyphonic Music Recordings," Proc. of FMA, 2012.

[6] J. Salamon, E. Gómez: "Melody Extraction from Polyphonic Music: MIREX 2011," in Music Information Retrieval Evaluation eXchange (MIREX), 2011.

[7] G. E. Poliner, D. P. W. Ellis, F. Ehmann, E. Gómez, S. Streich, and B. Ong: "Melody transcription from music audio: Approaches and evaluation," IEEE Transactions on Audio, Speech and Language Processing, 15(4):1247:1256, 2007.

[8] 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, 43:311:329, 2004.

[9] G. E. Poliner and D.P.W. Ellis: "A Classification Approach to Melody Transcription," Proc. of ISMIR, 2005.

[10] K. Dressler: "An auditory streaming approach for melody extraction from polyphonic music," Proc. Of ISMIR, pp. 19:24, Miami, 2011.

[11] J. Salamon, and E. Gómez: "Melody Extraction from Polyphonic Music Signals using Pitch Contours Characteristics," IEEE Transactions on Audio, Speech and Language Processing, 20(6):1759-1770, August 2012.

[12] D. FitzGerald, and M. Gainza: "Single Channel Vocal Separation using Median Filtering and Factorisation Techniques," ISAST Transactions on Electronic and Signal Processing, 4(1):62-73, 2010 (ISSN 1797-2329)

[13] A. Ozerov, P. Philippe, F. Bimbot, and R. Gribonval: "Adaptation of Bayesian models for single-channel source separation and its application to voice/music separation in popular songs," IEEE Transactions on Audio, Speech, and Language Processing, 15(5): 1564:1578, July 2007.

[14] B. Raj, P. Smaragdis, M. V. Shashanka, and R. Singh: "Separating a foreground singer from background music," Proc. Int Symp. Frontiers Res. Speech Music (FRSM), India, 2007.

[15] Y. Li, and D. Wang: "Separation of singing voice from music accompaniment for monaural recordings," Proc. of ICASSP, 15(4):1475:1487, May, 2007.

[16] H. Chao-Ling, and R. Jyh-Shing: "On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset," IEEE Transactions on Audio, Speech, and Language Processing, 18(2):310:319, February 2010.

[17] Z. Rafii, and B. Pardo: "A Simple Music/Voice Separation Method based on the Extraction of the Repeating Musical Structure," Proc. of ICASSP, Prague, May, 2011.

[18] J. Janer, J. Bonada, M. de Boer, and A. Loscos: "Audio Recording Analysis and Rating," Patent pending US20080026977, Universitat Pompeu Fabra, 06/02/2008.

[19] J. Salamon, E. Gómez, and J. Bonada: "Sinusoid Extraction and Salience Function Design for Predominant Melody Estimation," Proc. of DAFX, Paris, 2011, pp. 73-80.

[20] D. Lee, and H. Seung: "Algorithms for Non-negative Matrix Factorization," Advances in NIPS, pp. 556-562, 2000

[21] N. Ono, K. Miyamoto, J. Le Roux, H. Kameoka, and S. Sagayama: "Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram," Proc. of EUSIPCO, 2008

[22] T. Virtanen: "Monaural Sound Source Separation by Non-Negative Matrix Factorization with Temporal Continuity and Sparseness Criteria," IEEE Transactions on Audio, Speech, and Language Processing, 3(15), March 2007.

[23] A. de Cheveigné, and H. Kawahara: "YIN, a Fundamental Frequency Estimator for Speech and Music," Journal of the Acoustic Society of America, 111(4):1971:1930, 2002.

Author Index

FEUP EDIÇÕES

ISBN:
978-972-752-144-9

ORGANISATION

INESCTEC
TECHNOLOGY & SCIENCE
ASSOCIATE LABORATORY
PORTUGAL

IN PARTNERSHIP WITH

TNSJ TEATRO NACIONAL SÃO JOÃO PORTO

GOVERNO DE PORTUGAL | SECRETÁRIO DE ESTADO DA CULTURA

O TNSJ é membro da

UNION DES THEATRES DE L'EUROPE

Mecenas TNSJ

ANA Aeroportos de Portugal

GOLD SPONSORS

gracenote

BRONZE SPONSORS

Google

SPECTRAL MIND audio intelligence

quaero

SUPPORT

MIRES

FEUP Universidade do Porto Faculdade de Engenharia

CATÓLICA PORTO ARTS

CITAR Research Center for Science and Technology in Art

U.PORTO

MAUS HABITOS Espaço de Intervenção Cultural

ableton

'74 cycling '74 • tools for media