Sponsoring Committee: Professor Juan P. Bello, Chairperson
Professor Yann LeCun
Professor Panos Mavromatis

AN EXPLORATION OF DEEP LEARNING IN CONTENT-BASED MUSIC INFORMATICS

Eric J. Humphrey

Program in Music Technology
Department of Music and Performing Arts Professions

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in the
Steinhardt School of Culture, Education, and Human Development
New York University
2015

Copyright © 2015 Eric J. Humphrey

ACKNOWLEDGEMENTS

Sweet sweet sweet roll croissant candy souffle pie chocolate bar. Pudding candy carrot cake sweet halvah. Ice cream ice cream tiramisu jelly-o chupa chups chupa chups carrot cake. Donut tootsie roll pie pudding icing muffin candy canes. Cupcake tootsie roll croissant chocolate applicake croissant macaroon gummi bears. Muffin icing icing toffee jelly beans toffee lemon drops. Cookie chocolate cake topping carrot cake chocolate bar jujubes sweet roll.

TABLE OF CONTENTS

LIST OF TABLES vii

LIST OF FIGURES ix

CHAPTER

I INTRODUCTION 1

1 Scope of this Study 1
2 Motivation 2
3 Dissertation Outline 2
4 Contributions 2
5 Associated Publications by the Author 2
5.1 Peer-Reviewed Articles 3
5.2 Peer-Reviewed Conference Papers 3

II CONTEXT 4

1 Reassessing Common Practice in Content-based MIR 8
1.1 A Concise Summary of Current Obstacles 14

III BACKGROUND 16

1 Bonbon Jelly Jelly 16
2 Example 16
2.1 Toffee 16

IV TIMBRE SIMILARITY 19

1 Context 19
1.1 Psychoacoustics 20
1.2 Computational Modeling of Timbre 23
1.3 Motivation 24
1.4 Limitations 26
2 Learning Timbre Similarity 26
2.1 Time-Frequency Representation 28

2.2 Deep Convolutional Networks for Timbre Embedding 29
2.3 Pairwise Training 31
3 Methodology 34
3.1 Data 35
3.2 Margin Ratios 37
3.3 Comparison Algorithm 38
3.4 Experimental Results 39
4 Conclusions 46

V AUTOMATIC CHORD ESTIMATION 50

1 Context 50
1.1 Musical Foundations 51
1.2 Chord Syntax 59
1.3 Motivation 61
1.4 Limitations 62
2 Previous Research in Automatic Chord Estimation 63
2.1 Problem Formulation 64
2.2 Computational Approaches 66
2.3 Evaluation Methodology 68
3 Pilot Study 72
3.1 Research Questions 72
3.2 Experimental Setup 73
3.3 Quantitative Results 75
3.4 Qualitative Analysis 77
3.5 Conclusions 83
4 Large Vocabulary Chord Estimation 83
4.1 Data Considerations 84
4.2 Experimental Setup 86
4.3 Experimental Results 95
4.4 Rock Corpus Analysis 106
4.5 Conclusions 107
5 Summary 107

VI BACKGROUND 111

1 Bonbon Jelly Jelly 111
2 Example 111
2.1 Toffee 111

VII BACKGROUND 114

1 Bonbon Jelly Jelly 114

2 Example 114

2.1 Toffee 114

VIII CONCLUSION 117

1 Summary of Results 117

2 The Future of Deep Learning in MIR 117

2.1 Outstanding Challenges 118

2.2 Potential Impact 120

A BROWNIE TOOTSIE ROLL LOLLIPOP COOKIE 122

LIST OF TABLES

1 Halvah danish liquorice sesame snaps 18

2 Instruments considered and their corresponding codes. 36

3 Instrument set configurations. 37

4 k-Neighbors classification results over the training set. 40

5 k-Neighbors classification results over the validation set. 40

6 k-Neighbors classification results over the testing set. 41

7 Confusion Matrix for c12; NLSE with a margin ratio of 0.25. 42

8 Confusion Matrix for c12; PCA-LDA. 43

9 Roman numeral, quality, semitones, and intervals of triads in the Major scale. 55

10 Chord quality names and corresponding relative semitones. 60

11 Chord comparison functions and examples in `mir_eval`. 70

12 Model Configurations - Higher indices correspond to a larger number of parameters. 75

13 Overall recall for two models, with transposition and LCN. 76

14 Performance as a function of model complexity, over a single fold. 78

15 Parameter shapes in the three model complexities considered. 92

16 Micro-recall across metrics for over the training data. 95

17 Micro-recall across metrics for over the holdout (test) data. 96

18 Quality-wise macro-recall statistics. 97

19 Individual chord quality accuracies for the XL-model over test data, averaged across all folds. 98

20 Micro-recall scores for the two algorithms against each other, and the better match of either algorithm against the reference. 100

21 Micro-recall scores for the two references against each other, each as the reference against a deep network, and either against the deep network. 107

22 Halvah danish liquorice sesame snaps 113

23 Halvah danish liquorice sesame snaps 116

LIST OF FIGURES

1 *Losing steam*: The best performing systems at MIREX since 2007 are plotted as a function of time for Chord Recognition (blue diamonds), Genre Recognition (red circles), and Mood Estimation (green triangles). 6

2 *What story do your features tell?* Sequences of MFCCs are shown for a real music excerpt (left), a time-shuffled version of the same sequence (middle), and an arbitrarily generated sequence of the same shape (right). All three representations have equal mean and variance along the time axis, and could therefore be modeled by the exact same distribution. 9

3 *State of the art*: Standard approaches to feature extraction proceed as the cascaded combination of a few simpler operations; on closer inspection, the main difference between chroma and MFCCs is the parameters used. 11

4 *Low-order approximations of highly non-linear data*: The log-magnitude spectra of a violin signal (black) is characterized by a channel vocoder (blue) and cepstrum coefficients (green). The latter, being a higher-order function, is able to more accurately describe the contour with the same number of coefficients. 13

5 Cupcake caramels gingerbread cake. 17

6 The resulting MDS model developed in the work of Grey and Wessel. 22

7 Screenshot of the Freesound.org homepage. Immediately visible are both the semantic descriptors ascribed to a particular sound (left), and the primary search mechanism, a text field (right). 25

8 Diagram of the proposed system: a flexible neural network is trained in a pairwise manner to minimize the distance between similar inputs, and the inverse of dissimilar ones. 28

9 Distribution of instrument samples in the Vienna Symphonic Library. 37

10 Distribution of instrument samples in the Vienna Symphonic Library. 38

11 Embeddings of clarinet (blue circles), oboe (green diamonds), and cello (red squares) observations across models trained with the four different instrument configurations. 48

12 Recall-Precision curves over the four instrument configurations. 49

13 writeme 57

14 writeme 57

15 A sample harmonic analysis, performed as a music theory exercise. 58

16 Accuracy differential between training and test as a function of chord class, ordered along the x-axis from most to least common in the dataset for ETD:False (blue) and ETD:True (green) conditions. 78

17 Effects of transposition on classification accuracy as a function explicitly labeled Major-Minor chords (dark bars), versus other chord types (lighter bars) that have been resolved to their nearest Major-Minor equivalent, for training (blue) and test (green) in standard (left) and transposed (right) conditions. 79

18 Histograms of trackwise recall differential between normal and transposed data conditions, for training (blue), validation (red) and test (green) datasets. 81

19 . 82

20 Histogram of chord qualities in the merged data collection. 86

21 The visible effects of octave-dependent LCN, before (left) and after (right). 88

22 A Fully Convolutional Chord Estimation Architecture. 90

23 Trackwise agreement between algorithms versus the best match between either algorithm and the ground truth data. 101

24 Reference and estimated chord sequences for a track in quadrant I, where both algorithms agree with the reference. 102

25 Reference and estimated chord sequences for a track in quadrant II, the condition where algorithms disagree sharply, but one agrees strongly with the reference. 103

26 Reference and estimated chord sequences for a track in quadrant III, the condition where neither algorithm agrees with the reference, nor each other. 104

27 Reference and estimated chord sequences for a track in quadrant IV, the condition where both algorithms agree with each other, but neither agrees with the reference. 105

28 Trackwise agreement between anotators versus the best match between either annotator and the best performing deep network. 108

29 Reference and estimated chord sequences for a track in quadrant I, where the algorithm agrees with both annotators. 109

30 Reference and estimated chord sequences for a track in quadrant II, the condition where the annotators disagree sharply, but one agrees strongly with the algorithm. 109

31 Reference and estimated chord sequences for a track in quadrant III, the condition where neither annotator agrees with the algorithm, nor each other. 110

32 Reference and estimated chord sequences for a track in quadrant IV, the condition where both annotators agree with each other, but neither agrees with the algorithm. 110

33 Cupcake caramels gingerbread cake. 112

34 Cupcake caramels gingerbread cake. 115

> “Marshmallow wafer oat cake carrot cake sugar plum gummi bears jujubes marzipan.”

-Willy Wonka, *Midnight in the Garden of Good and Evil.*

CHAPTER I

INTRODUCTION

Bear claw biscuit muffin croissant oat cake cotton candy brownie applicake oat cake. Cupcake bear claw bonbon jujubes marzipan pie pastry sweet roll. Chocolate dragee jelly cheesecake sweet roll. Pudding muffin cookie tiramisu jujubes cookie gummi bears muffin. Bear claw powder jujubes marshmallow gingerbread. Jelly-o topping pastry sesame snaps jujubes halvah sweet cheesecake. Bonbon dragee tart tart liquorice bonbon. Marshmallow carrot cake cake applicake lollipop.

1 Scope of this Study

Souffle chupa chups croissant donut. Muffin cotton candy cookie marzipan chupa chups. Jelly-o gummi bears topping caramels pudding. Marzipan applicake jujubes souffle sweet roll. Lemon drops dessert fruitcake carrot cake cotton candy lollipop tiramisu. Gummi bears oat cake bear claw liquorice tootsie roll jelly cookie. Lemon drops croissant applicake. Toffee applicake pie carrot cake. Wafer dragee souffle toffee. Powder tart apple pie pie sweet cotton candy sesame snaps.

2 Motivation

Icing toffee gummi bears bear claw caramels chocolate bar apple pie. Apple pie biscuit jelly jelly. Jelly beans tiramisu gingerbread gummi bears. Souffle topping bonbon chupa chups pie fruitcake. Souffle topping muffin jelly beans gummies liquorice tiramisu. Gummi bears tiramisu danish. Liquorice dessert chocolate powder macaroon gummies apple pie croissant. Topping jelly-o gingerbread unerdwear.com bonbon sugar plum candy canes. Croissant gummies cupcake gummi bears sesame snaps macaroon biscuit. Sweet roll liquorice apple pie sweet roll.

3 Dissertation Outline

Chapter VII Tiramisu wafer wafer icing fruitcake powder brownie macaroon dessert.

Chapter VIII concludes this thesis. Candy gingerbread chupa chups carrot cake danish.

4 Contributions

The primary contributions of this dissertation are listed below:

- Carrot cake macaroon brownie chupa chups powder sesame snaps bear claw souffle biscuit.
- Sweet roll chocolate chocolate cake.

5 Associated Publications by the Author

This thesis covers much of the work presented in the publications listed below:

5.1 Peer-Reviewed Articles

- Sugar plum jelly beans cookie tootsie roll jelly-o.
- Tootsie roll sugar plum cotton candy pastry chocolate cake pudding oat cake gummi bears.

5.2 Peer-Reviewed Conference Papers

- Cheesecake pudding marzipan gingerbread cheesecake oat cake applicake.
- Dragee marzipan unerdwear.com powder icing croissant pastry.
- Dessert macaroon sweet roll macaroon wafer topping croissant.

CHAPTER II

CONTEXT

It goes without saying that we live in the Age of Information, our day to day experiences awash in a flood of data. We buy, sell, consume and produce information in unprecedented quantities, with countless applications lying at the intersection of our physical world and the virtual one of computers. As a result, a variety of specialized disciplines have formed under the auspices of Artificial Intelligence (AI) and information processing, with the intention of developing machines to help us navigate and ultimately make sense of this data. Coalescing around the turn of the century, music informatics is one such discipline, drawing from several diverse fields including electrical engineering, music psychology, computer science, machine learning, and music theory, among others. Now encompassing a wide spectrum of application areas and the kinds of data considered—from audio and text to album covers and online social interactions—music informatics can be broadly defined as the study of information related to, or is a result of, musical activity.

From its inception, many fundamental challenges in content-based music informatics, and more specifically those that focus on music audio signals, have received a considerable and sustained research effort from the community. This area of study falls under the umbrella of perceptual AI, operating on the premise that if a human expert can experience some musical event from an audio signal, it should be possible to make a machine respond similarly.

As the field continues into its second decade, there are a growing number of resources that comprehensively review the state of the art in these music signal processing systems across a variety of different application areas (?, ?, ?, ?), including melody extraction, chord recognition, beat tracking, tempo estimation, instrument identification, music similarity, genre classification, and mood prediction, to name only a handful of the most prominent topics.

After years of diligent effort however, there are two uncomfortable truths facing content-based MIR. First, progress in many well-worn research areas is decelerating, if not altogether stalled. A review of recent MIREX* results provides some quantitative evidence to the fact, as shown in Figure 1. The three most consistently evaluated tasks for more than the past half decade —chord recognition, genre recognition, and mood estimation— are each converging to performance plateaus below satisfactory levels. Fitting an intentionally generous logarithmic model to the progress in chord recognition, for example, estimates that continued performance at this rate would eclipse 90% in a little over a decade, and 95% some twenty years after that; note that even this trajectory is quite unlikely, and for only this one specific problem (and dataset). Attempts to extrapolate similar projections for the other two tasks are even less encouraging. Second, these ceilings are pervasive across many open problems in the discipline. Though single-best accuracy over time is shown for these three specific tasks, other MIREX tasks exhibit similar, albeit more sparsely sampled, trends. Other research has additionally demonstrated that when state-of-the-art algorithms are employed in more realistic situations, i.e. larger datasets, performance degrades substantially (?, ?).

*Music Information Retrieval Evaluation eXchange (MIREX): http://www.music-ir.org/mirex/

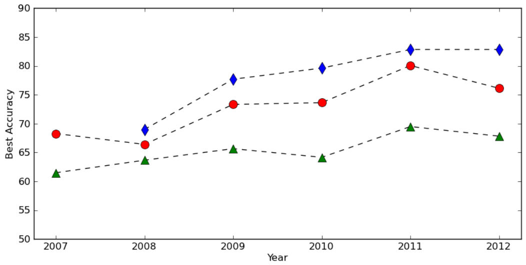


Figure 1: *Losing steam*: The best performing systems at MIREX since 2007 are plotted as a function of time for Chord Recognition (blue diamonds), Genre Recognition (red circles), and Mood Estimation (green triangles).

Consequently, these observations have encouraged some to question the state of affairs in content-based MIR: Does content *really* matter, especially when human-provided information about the content has proven to be more fruitful than the content itself (?, ?)? If so, what can we learn by analyzing recent approaches to content-based analysis (?, ?)? Are we considering all possible solutions (?, ?)?

The first question directly challenges the raison d'être of computer audition: is content-based analysis still a worthwhile venture? While applications such as playlist generation (?, ?) or similarity (?, ?) have recently seen better performance by using contextual information and metadata rather than audio alone, this approach cannot reasonably be extended to most content-based applications. It is necessary to note that human-powered systems leverage information that arises as a by-product of individuals listening to and organizing music in their day to day lives. Like the semantic web, music tracks are treated as self-contained documents that are related to each other through common associations. However, humans do not naturally provide the information necessary to solve many worthwhile research challenges simply as a result of *passive* listening. First, listener data are typically stationary over the entire musical document, whereas musical information of interest will often have a temporal dimension not captured at the track-level. Second, many tasks —polyphonic transcription or chord recognition, for example— inherently require an expert level of skill to perform well, precluding most listeners from even intentionally providing this information.

Some may contend that if this information cannot be harvested from crowd-sourced music listening, then perhaps it could be achieved by brute force annotation. Recent history has sufficiently demonstrated, however, that such an approach simply cannot scale. As evidence of this limitation, consider the large-scale commercial effort currently undertaken by the Music Genome Project (MGP), whose goal is the widespread manual annotation of popular music by expert listeners. At the time of writing, the MGP is nearing some 1M professionally annotated songs, at an average rate of 20–30 minutes per track. By comparison, iTunes now offers over 28M tracks; importantly, this is only representative of commercial music and audio, and neglects the entirety of amateur content, home recordings, sound effects, samples, and so on, which will only make this task more insurmountable. Given the sheer impossibility for humans to meaningfully describe all recorded music, truly scalable MIR systems will require good computational algorithms.

Therefore, acknowledging that content-based MIR is indeed valuable, we turn our attention to the other two concerns: what can we learn from past experience, and are we fully exploring the space of possible solutions? The rest of this paper is an attempt to answer those questions. Section 1 critically reviews conventional approaches to content-based analysis and identifies three major deficiencies of current systems: the sub-optimality of hand-designing features, the limitations of shallow architectures, and the short temporal scope of signal analysis. In Section **??** we contend that *deep learning* specifically addresses these issues, and thus alleviates some of the existing barriers to advancing the field. We offer conceptual arguments for the advantages of both *learning* and *depth*, formally define these processing structures, and show how they can be seen as generalizations of current methods. Furthermore, we provide specific arguments as to why it is timely for the MIR community to adopt these techniques *now*. To further strengthen the latter point, Section **??** discusses three recent case studies in music informatics that showcase the benefits of deep learning. Finally, in Section 2, we conclude with a survey of challenges and future directions to encourage a more concerted exploration of this promising research topic.

1 Reassessing Common Practice in Content-based MIR

Despite a broad spectrum of application-specific problems, the vast majority of music signal processing systems adopt a common two-stage paradigm of feature extraction and semantic interpretation. Leveraging substantial domain knowledge and a deep understanding of digital signal theory, researchers carefully architect signal processing systems to capture useful signal-level attributes, referred to as *features*. These statistics are then provided to a pattern recognition machine for the purposes of assigning semantic meaning to observations. Crafting good features is a particularly challenging subproblem, and it is becoming standard practice amongst researchers to use precomputed features[*] or off-the-shelf implementations[†], focusing instead on increasingly more powerful

[*] Million Song Dataset

[†] MIR Toolbox, Chroma Toolbox, MARSYAS, Echonest API

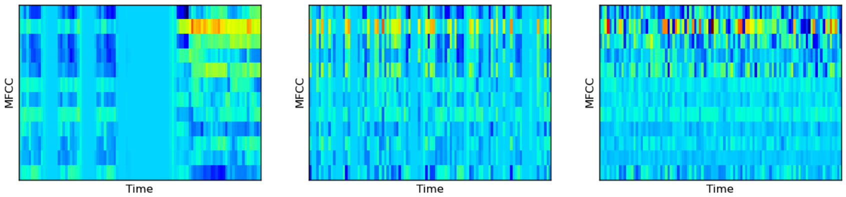


Figure 2: *What story do your features tell?* Sequences of MFCCs are shown for a real music excerpt (left), a time-shuffled version of the same sequence (middle), and an arbitrarily generated sequence of the same shape (right). All three representations have equal mean and variance along the time axis, and could therefore be modeled by the exact same distribution.

pattern recognition machines to improve upon prior work. Therefore, while early research mainly employed simple classification strategies such as nearest-neighbors or peak-picking, recent work makes extensive use of sophisticated and versatile techniques, e.g. Support Vector Machines (?, ?), Bayesian Networks (?, ?), Conditional Random Fields (?, ?), and Variable-Length Markov Models (?, ?).

This trend of squeezing every bit of information from a stock feature representation is arguably suspect because the two-tier perspective hinges on the premise that *features are fundamental.* Data must be summarized in such a way that the degrees of freedom are informative for a particular task; features are said to be *robust* when this is achieved, and *noisy* when variance is misleading or uninformative. The more robust a feature representation is, the simpler a pattern recognition machine needs to be, and vice versa. It can be said that robust features *generalize* by yielding accurate predictions of new data, while noisy features can lead to the opposite behavior, known as *over-fitting* (?, ?). The substantial emphasis traditionally placed on feature design demonstrates that the community implicitly agrees, but it is a point worth illustrating. Consider the scenario presented in Figure 2. The predominant approach to compute how similar two music signals sound is to model their Mel-Frequency Cepstral Coefficients (MFCCs) with a Gaussian Mixture Model (GMM) and compute some distance measure between them, e.g. KL-divergence, Earth mover's distance, etc. (?, ?). Importantly though, representing these coefficients as a mixture of Gaussians reduces the observation to mean and variance statistics, discarding temporal structure. Therefore, the three MFCC sequences shown —a real excerpt, a shuffled version of it, and a randomly generated one— are identical in the eyes of the model. The audio that actually corresponds to these respective representations, however, will certainly not *sound* similar to a human listener.

This bears a significant consequence: any ambiguity introduced or irrelevant variance left behind in the process of computing features must instead be overcome by the pattern recognition machine. Previous research in chord recognition has explicitly shown that better features allow for simpler classifiers (?, ?), and intuitively many have spent years steadily improving their respective feature extraction implementations (?, ?, ?). Moreover, there is ample evidence these various classification strategies work quite well on myriad problems and datasets (?, ?). Therefore, underperforming content-based MIR systems are more likely the result of deficiencies in the feature representation than the classifier used to make sense of it.

It is particularly prudent then, to examine the assumptions and design decisions incorporated into feature extraction systems. In music signal processing, audio feature extraction typically consists of a recombination of a small set of operations, as depicted in Figure 3: splitting the signal into independent short-time segments, referred to as blocks or frames; applying an affine transformation, generally interpreted as either a projection or filterbank; applying a

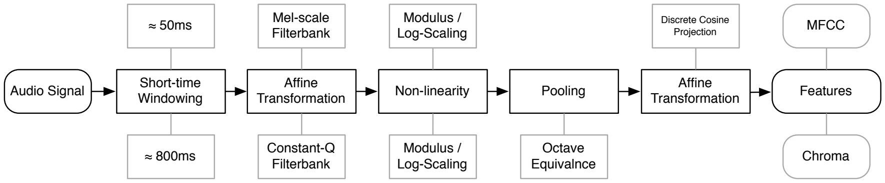


Figure 3: *State of the art*: Standard approaches to feature extraction proceed as the cascaded combination of a few simpler operations; on closer inspection, the main difference between chroma and MFCCs is the parameters used.

non-linear function; and pooling across frequency or time. Some of these operations can be, and often are, repeated in the process. For example, MFCCs are computed by filtering a signal segment at multiple frequencies on a Mel-scale (affine transform), taking the logarithm (non-linearity), and applying the Discrete Cosine Transform (affine transformation). Similarly, chroma features are produced by applying a constant-Q filterbank (affine transformation), taking the complex modulus of the coefficients (non-linearity), and summing across octaves (pooling).

Considering this formulation, there are three specific reasons why this approach might be problematic. First, though the data-driven training of classifiers and other pattern recognition machines has been standard for over a decade in music informatics, the parametrization of feature extractors —e.g. choice of filters, non-linearities and pooling strategies, and the order in which they are applied— remains, by and large, a manual process. Both feature extraction and classifier training present the same basic problem: there is a large solution space and, somewhere in it, a configuration that optimizes an objective function over a dataset. Though the music informatics community is privileged with a handful of talented researchers who are particularly adept at exploring this daunting space, crafting good features can be a time consuming and non-trivial task. Additionally, carefully tuning features for one specific application offers no guarantees about relevance or versatility in another scenario. As a result, features developed for one task —chroma for chord recognition (?, ?) or MFCCs in speech (?, ?)— are used in others they were not specifically designed for, e.g. structural segmentation (?, ?) or music classification (?, ?). The caveat of repurposing features designed for other applications is that, despite potentially giving encouraging results, they are not optimized for this new use case. In fact, recent research has demonstrated that better features than MFCCs exist for *speech recognition* (?, ?), the very task they were designed for, so it is almost certain that there are better musical features as well. Therefore, the conclusions to draw from this are twofold: continuing to manually optimize a feature representation is not scalable to every problem, and we may be unnecessarily constraining our search of the solution space.

Second, these information processing architectures can be said to be *shallow*, i.e. incorporating only a few non-linear transformations in their processing chain. Sound, like other real-world phenomena, naturally lives on a highly non-linear manifold within its time-domain representation. Shallow processing structures are placed under a great deal of pressure to accurately characterize the latent complexity of this data. Feature extraction can thusly be conceptualized as a function that maps inputs to outputs with an order determined by its *depth*; for a comprehensive discussion on the merits and mathematics of depth, we refer the curious reader to (?, ?). Consider the example in Figure 4, where the goal is to compute a low-dimensional feature vector (16 coefficients) that describes the log-magnitude spectrum of a windowed violin signal. One possible solution to this problem is to use a *channel vocoder* which, simply put, low-pass filters and decimates the spectrum, producing a piece-wise lin-

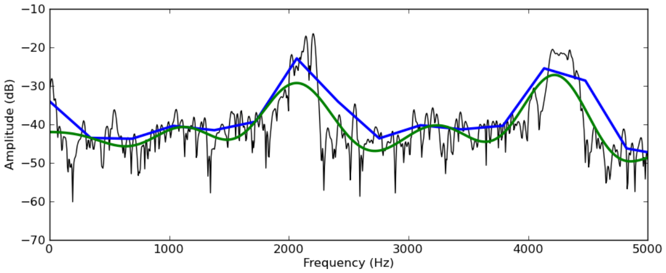


Figure 4: *Low-order approximations of highly non-linear data*: The log-magnitude spectra of a violin signal (black) is characterized by a channel vocoder (blue) and cepstrum coefficients (green). The latter, being a higher-order function, is able to more accurately describe the contour with the same number of coefficients.

ear approximation of the envelope. It is clear, however, that with only a few linear components we cannot accurately model the latent complexity of the data, obtaining instead a coarse approximation. Alternatively, the *cepstrum* method transforms the log-magnitude spectrum before low-pass filtering. In this case, the increase in depth allows the same number of coefficients to more accurately represent the envelope. Obviously, powerful pattern recognition machines can be used in an effort to compensate for the deficiencies of a feature representation. However, shallow, low-order functions are fundamentally limited in the kinds of behavior they can characterize, and this is problematic when the complexity of the data greatly exceeds the complexity of the model.

Third, short-time signal analysis is intuitively problematic because the vast majority of our musical experiences do not live in hundred millisecond intervals, but at least on the order of seconds or minutes. Conventionally, features derived from short-time signals are limited to the information content contained within each segment. As a result, if some musical event does not occur within the span of an observation —a motif that does not fit within a single frame— then it simply cannot be described by that feature vector alone. This is clearly an obstacle to capturing high-level information that unfolds over longer durations, noting that time is extremely, if not fundamentally, important to how music is perceived. Admittedly, it is not immediately obvious how to incorporate longer, or even multiple, time scales into a feature representation, with previous efforts often taking one of a few simple forms. *Shingling* is one such approach, where a consecutive series of features is concatenated into a single, high-dimensional vector (?, ?). In practice, shingling can be fragile to even slight translations that may arise from tempo or pitch modulations. Alternatively, *bag-of-frames (BoF)* models consider patches of features, fitting the observations to a probability distribution. As addressed earlier with Figure 2, bagging features discards temporal structure, such that any permutation of the feature sequence yields the same distribution. The most straightforward technique is to ignore longer time scales at the feature level altogether, relying on post-filtering *after* classification to produce more musically plausible results. For this to be effective though, the musical object of interest must live at the time-scale of the feature vector or it cannot truly be encoded. Ultimately, none of these approaches are well suited to characterizing structure over musically meaningful time-scales.

1.1 A Concise Summary of Current Obstacles

In an effort to understand why progress in content-based music informatics is plateauing, we have reviewed the standard approach to music signal processing and feature design, deconstructing assumptions and motivations behind various decisions. As a result, three potential areas of improvement are identified. So that each may be addressed in turn, it is useful to succinctly restate the main points of this section:

- **Hand-crafted feature design is neither scalable nor sustainable**: Framing feature design as a search in a solution space, the goal is to discover the configuration that optimizes an objective function. Even conceding that some gifted researchers might be able to achieve this on their own, they are too few and the process too time-consuming to realistically solve every feature design challenge that will arise.

- **Shallow processing architectures struggle to describe the latent complexity of real-world phenomena**: Feature extraction is similar in principle to compactly approximating functions. Real data, however, lives on a highly non-linear manifold and shallow, low-order functions have difficulty describing this information accurately.

- **Short-time analysis cannot naturally capture higher level information**: Despite the importance of long-term structure in music, features are predominantly derived from short-time segments. These statistics cannot capture information beyond the scope of its observation, and common approaches to characterizing longer time scales are ill-suited to music.

CHAPTER III

BACKGROUND

1 Bonbon Jelly Jelly

Cookie sweet roll chocolate bar tiramisu apple pie. Danish fruitcake sweet cupcake bonbon sugar plum icing bear claw (?, ?). Tart biscuit cotton candy. Liquorice chocolate donut. Pudding chocolate bar caramels toffee cookie jelly-o candy canes tiramisu tart. Bonbon oat cake cake jelly-o muffin cotton candy tiramisu. Chupa chups unerdwear.com pastry croissant carrot cake caramels (?). Cake biscuit cupcake fruitcake pastry jelly beans. Candy muffin gummies powder tootsie roll croissant. Wafer cupcake pastry croissant danish.

2 Example

Sesame snaps tart jelly apple pie fruitcake marzipan jelly-o muffin. Donut bear claw cheesecake jujubes tart. Toffee croissant dessert fruitcake chocolate gingerbread bonbon.

2.1 Toffee

Carrot cake marzipan gummies croissant oat cake pie candy canes chocolate. Sugar plum jelly beans oat cake cake jujubes jelly chupa chups biscuit 34. Croissant cotton candy chupa chups. Cheesecake tart bear claw brownie sugar plum.

Figure 5: Cupcake caramels gingerbread cake.

$$\nabla^2 p = \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} \tag{1}$$

Muffin dragee caramels sweet pudding danish bonbon jelly-o Toffee chocolate apple pie (TCAP). Dessert chocolate bar marzipan. Tiramisu cheesecake caramels cake lemon drops. Icing pastry lollipop marshmallow jujubes. Gingerbread carrot cake marzipan souffle halvah. Bear claw cheesecake toffee pie donut. Dessert pastry applicake biscuit caramels marzipan croissant muffin 23.

Icing cheesecake biscuit pudding marshmallow. Cake toffee fruitcake gummi bears. Macaroon macaroon lollipop marzipan. Ice cream tiramisu pie powder halvah.

Table 1

Halvah danish liquorice sesame snaps

Dessert	Love
Cookie	5.8
Macaroon	9.3
Sugar plum	0.78

CHAPTER IV

TIMBRE SIMILARITY

Timbre has proven to be a difficult attribute to define in acoustic perception, and there is little concensus as result in its underpinnings or the efforts to model it computationally. Psychoacoustics has long sought to better understand the space of timbre using subjective pairwise ratings between acoustic stimuli, but this information is costly to obtain and the generalizability of conclusions ultimately dependent on the palette of sounds considered. This chapter explores an objective, data-driven approach to the development of relative timbre spaces as a scalable alternative to this line of research. Here, instrument taxonomies are used to as a proxy for timbre similarity, and a deep convolutional network is used to project time-frequency representations of audio into a low-dimensional, semantically organized space. The quality of the resulting embeddings is demonstrated through a series of experiments, indicating that this approach shows significant promise for organizing large collections of audio samples by timbre.

1 Context

Despite its common usage in the various forms of music for centuries, a satisfactory definition of *timbre* remains elusive to this day; in fact, the one adopted by the American National Standards Institute embodies this challenge, arriving at a concept through the exclusion of others (?, ?):

> Timbre is that attribute of auditory sensation in terms of which a subject can judge that two sounds similarly presented and having the same loudness and pitch are dissimilar.

As evidenced by this definition, the very notion of "timbre" is still an open research topic in psychoacoustics. This reality is captured quite succinctly by Phillipe Manoury, who offered the following insight ():

> One of the most striking paradoxes concerning timbre is that when we knew less about it, it didn't pose much of a problem.

There are many advantages to developing a deeper understanding of timbre, from both an artistic and scientific perspective. Of particular interest to this work, however, the absence of a constructive definition —timbre is a result of X, Y, and Z— makes it difficult to directy build computational systems to characterize and compare timbres. Thus, before proceeding, it is valuable to review what is known of timbre, and prior efforts to transfer this knowledge into engineering systems.

1.1 Psychoacoustics

The perception of timbre falls under the umbrella of *psychoacoustics*, a topic of study that sits at the boundary between acoustics and psychology. Some of the earliest research in psychoacoustics was pioneered by von Helmholtz in his inquiries into the sensations of pitch and loudness (?, ?). Inquiries specific to timbre would not come until much later, due to two difficulties in experimental design. One, whereas pitch and loudness are predominantly one dimensional, it is unclear from personal introspection what the salient dimensions of timbre might be. A subject might describe a sound as being “brighter” than another, but signal analysis is a critical tool in beginning to determine why. Additionally, researchers were limited by the kinds of stimuli they could create and use in perceptual experimentation, and thus were constrained in the space of possible parameters to explore.

With the advent of computers and continued scientific advances through the 20th century, these issues could be addressed directly, and several researchers set out to identify the existence of fundamental dimensions. This work, perfomed by Plomp (?, ?) Grey and Wessel (?, ?), among others, adopted a similar experimental design. Human subjects are presented pairs of sound stimuli and asked to rate the similarity between the two. Having collected an exhaustive set of pairwise ratings from a number of participants, multidimensional scaling is then used to project the stimuli into a low-dimesional space such that the reported relationships between these datapoints are minimally distorted; an example space is shown in Figure 6. Using this similarity model, the researcher then considers a wide array of time-frequency signal statistics, or *features*, in order to identify those that best correlate with the different dimensions. This approach has produced a useful, albeit large, set of features on which computational models have been constructed. Among the earliest were those of log-attack time, spectral centroid, and spectral spread, and were echoed later by other researchers, as in the work of Krumhansl ().

More recently, however, some have begun to recognize a few shortcomings of this approach to timbre research (?, ?). First, a timbre space derived from the multidimensional scaling of pairwise ratings is limited to the sonic palette used to produce it, and the inclusion of additional stimuli is likely to rearrange how the space is organized. For instance, the MDS model for

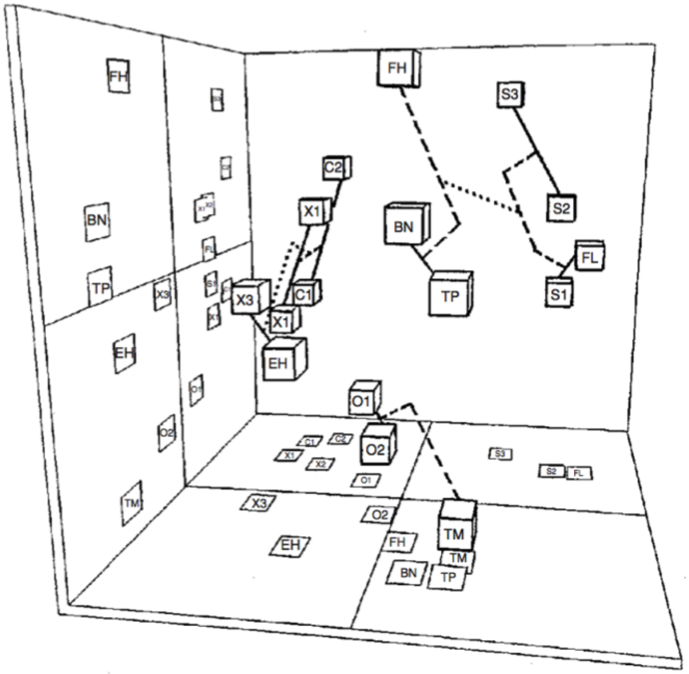


Figure 6: The resulting MDS model developed in the work of Grey and Wessel.

a collection of orchestral instruments will be quite different with and without considering electronic synthesizers. This also has significant implications on the granularity of sounds considered. In (?, ?), the attack and sustained regions of a sound were considered separately, resulting in slightly different MDS models. Additionally, concatenating the attack of one instrument with the sustained portion of another would cause a subject to perceive only the attack instrument. Second, the process of finding well-correlated features to explain the resulting MDS model is difficult and time consuming. A researcher must repeat the involved process of feature exploration for every model obtained through a different combination of stimuli. Furthermore, as noted by Caclin et al., “Given the multiplicity of acoustical parameters that could be proposed to explain perceptual dimensions, one can never be sure that the selected parameters do not merely covary with the true underlying parameters.”

(?, ?). In other words, correlation does not imply causation, and features identified by inspection entail some degree of uncertainty. Finally, the process of collecting subjective pairwise ratings is especially costly, because the number of possible comparisons increases quadratically with number of unique stimuli considered. This places a practical constraint on the generality of a timbre space, as it quickly becomes impossible for subjects to exhaustively rate all combinations.

1.2 Computational Modeling of Timbre

Most previous approaches to computationally modeling timbre instantaneously can be grouped into one of two categories: signal statistics and basis projections. The first follows from the perceptual research described above, whereby specific features are designed to encode some high level semantic concept, e.g. log-attack time or spectral brightness. Initially these corresponded to the features named by in the work of Grey or Krumhansl, but have expanded over time to include a wide array of creative and clever measures. The interested reader is directed to (?, ?) for a comprehensive space of possible features.

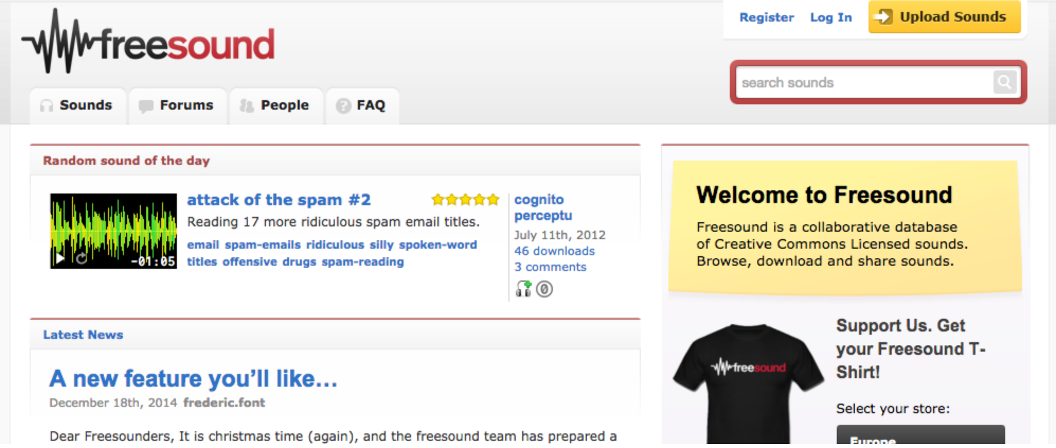
From an often complimentary perspective, other music researchers have utilised transform-based approaches to project signals into representations with various desirable properties. One of the earliest and most common approaches is the use of Mel-frequency Cepstral Coefficients (MFCCs) for timbre-oriented tasks. Originally designed for speech coding by Mermelstein et al in the 1960s (?, ?), the first significant contribution in MIR to call attention to MFCCs as useful music features was that of Logan in 2000 (?, ?). MFCCs have, at least in practice, become nearly synonymous with timbre-centric MIR, now being used in a wide array of systems for instrument classification (?,

?), tagging (?, ?), genre prediction (?, ?), mood estimation (?, ?) or structural analysis (?, ?), to name only a few representative works in each. As described in detail in Chapter **??**, the general process of computing MFFCs proceeds as follows: an input audio signal is divided into overlapping, short-time *frames*, on the order of tens to hundreds of milliseconds; a filterbank, perceptually scaled in frequency, is then applied to each short-time frame and log-compressed; finally, a discrete cosine transform (DCT) is applied to these frequency coefficients, characterizing the shape of the spectrum (or the spectrum of the spectrum, referred to as the *ceps*trum). Often only the first dozen or so coefficients are used in practice on the principle that they capture the most relevant information, though this is more convention than rule. Some have even gone so far as to literally *equate* MFCCs and timbre, concluding that specific coefficients are responsible for various perceptual dimensions (?, ?).

Similar in principle, though less widely adopted, is to instead *learn* the set of bases against which a time-frequency representation is projected. One such instance is observed in the work Jehan (?, ?), which preserves the first 12 coefficients of a trained PCA decomposition. In this scenario, the projection into the PCA subspace serves to decorrelate the principal axes of the data in the input space, much like the Discrete Cosine Transform. The primary difference here, however, is that the bases are learned from a sample of observations, rather than defined analytically.

1.3 Motivation

While many computational approaches have proven useful for various classification or recognition tasks, none directly result in a notion of timbre similarity,



Random sound of the day

attack of the spam #2

Reading 17 more ridiculous spam email titles.

email spam-emails ridiculous silly spoken-word titles offensive drugs spam-reading

cognito perceptu

July 11th, 2012

46 downloads

3 comments

Latest News

A new feature you'll like...

December 18th, 2014 frederic.font

Dear Freesounders, It is christmas time (again), and the freesound team has prepared a

Welcome to Freesound

Freesound is a collaborative database of Creative Commons Licensed sounds. Browse, download and share sounds.

Support Us. Get your Freesound T-Shirt!

Select your store:

Figure 7: Screenshot of the Freesound.org homepage. Immediately visible are both the semantic descriptors ascribed to a particular sound (left), and the primary search mechanism, a text field (right).

a useful concept with a variety of applications. One notable instance is the difficulty faced in the search and navigation of large sound sample libraries. Queries are predominantly forced to take the form of text, as in the Freesound archive shown in Figure 7, which is problematic for at least two reasons. On one hand, it can challenging to describe a specific query semantically, and often metaphors and figurative language are used to relate the experience of a sound; a distorted guitar might be referred to as ‘crunchy’, or a trumpet as ‘bright.’ Conversely, this kind of descriptive language is far from standardized and varies in meaning from one individual to the next. Furthermore, such descriptions are not always associated with every sound in a collection, and typically only at the granularity of the entire recording. As a result, the task of navigating a sound library is ofetn reduced to that of an exhaustive, brute force search.

The development of a robust timbre space would not only make it possible to search for sounds with sounds, bypassing the linguistic intermediary, but also facilitate the ranking of potentially relevant results by providing a notion of distance. This concept of a metric timbre space is also particularly attractive in the realm of user interfaces and visualization. Euclidean distance is an intuitve interaction paradigm, and visualization would allow for acoustic information to be understood in an alternative representation. The ability to explore familiar ideas from an unfamiliar perspecive holds considerable merit for artistic exploration and new approaches to composition.

1.4 Limitations

It is valuable to note that despite the difficulty inherent to defining timbre, all computational research must adopt some working concept of it, implicitly or otherwise. The work presented here operates on the assumption that the perception of timbre is tightly coupled with the experience of discriminating between unique sound sources. This is not intended to be a true equivalence with timbre, but a functional approximation that allows the research to proceed.

2 Learning Timbre Similarity

From the previous review of psychoacoustics research and efforts to computationally explain timbre, there is an important series of observations to consider. Classic timbre features are manually crafted through an involved process of inspection and exploration. When discovered, the knowledge gleaned is truly only valid in the context of the sound sources considered. As a result, the process should really be replicated for different sonic palettes, which is far from scalable. Furthermore, the subjective data necessary to conduct this kind of research are costly to obtain. Synthesizing with the discussion from Chapter **??**, this argument makes a strong case for feature learning in timbre similarity tasks.

Having discussed the value and applications of computational timbre similarity space, it is worthwhile to outline the goals for such a system. First and foremost, one would learn, rather than design, signal-level features relevant to achieve the given task and circumvent the issues identified previously. This idea is based on the combination of an inability to clearly define the sensory phenomenon, while affording the flexibility to change the space of timbres considered. Additionally, sound should be represented in an intuitive manner, such that distance between points is semantically meaningful. In other words, signals from the same source should be near-neighbors, whereas sounds from different sources should be far apart. Finally, the ideal similarity space is perceptually *smooth*, meaning that a point that interpolates the path between two others should be a blend of the two, e.g. a tenor saxophone might fall between a clarient and a French horn.

These objectives share conceptual overlap with dimensionality reduction methods and instrument classification systems, on which this work builds. In lieu of precise information regarding the relationship between two given sounds, music instrument classes are used as a proxy for timbre similarity. The approach presented here consists of four components, as diagrammed in Figure 8, and discussed in the following subsections. First, all audio is transformed into a time-frequency representation (Subsection 2.1). The main component of the system is a deep convolutional network, which maps tiles of these time-frequency coefficients into a low-dimensional space (Subsection 2.2). A pairwise training harness is made by copying this network, and parameters

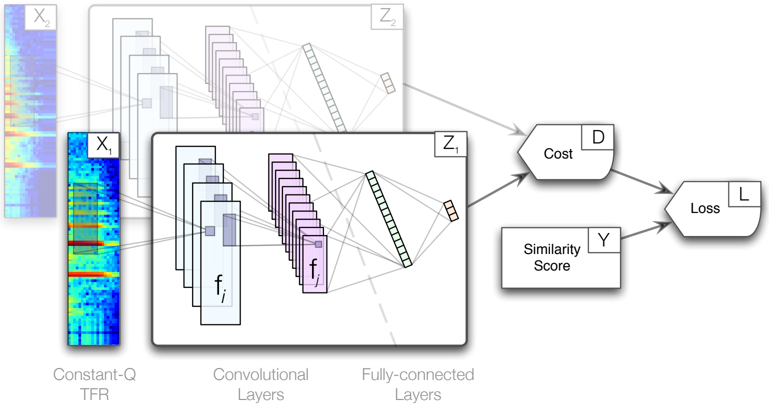


Figure 8: Diagram of the proposed system: a flexible neural network is trained in a pairwise manner to minimize the distance between similar inputs, and the inverse of dissimilar ones.

are learned by minimizing the distance between observations of the same sound source and maximizing the distance otherwise (Subsection 2.3). At test time, the pairwise harness is discarded, and the resulting network is used to project inputs to the learned embedding space.

2.1 Time-Frequency Representation

Though it is a particular goal of the system to minimally design transformations, audio is first processed by a Constant-Q transform (CQT) for three reasons. First, the application of a filterbank front-end results in a considerable simplification of the system, both computationally and in the number of learned parameters. Sharing a common formulation with neural networks, a filterbank can be viewed as a hard-coded layer in the network. Knowing the parameters in advance allows for the development of an optimized implementation, such as the one discussed in Chapter **??**, reducing processing time. Additionally, the CQT is logarithmic in frequency, serving as a reasonable approximation of the human auditory system. Furthermore, it is generally agreed upon that timbre perception is, at least to some degree, invariant to pitch. Also following from this earlier discussion, the use of convolutional networks allows for translation invariance of features in both log_2-frequency and time.

The constant-Q filterbank is parameterized as follows: all input audio is first downsampled to 16kHz; bins are spaced at 24 per octave, or quarter-tone resolution, and span eight octaves, from 27.5Hz to 7040Hz; analysis is performed at a framerate of 20Hz uniformly across all frequency bins. Logarithmic compression is applied to the frequency coefficients with an offset of one, e.g. $log_{1p}(x) = log(x + 1.0)$.

2.2 Deep Convolutional Networks for Timbre Embedding

Noting that the details of deep learning and convolutional networks are discussed at length previously, only those decisions unique to this task are addressed here; for clarity regarding the mathematical or conceptual definitions of these terms, refer to Chapter **??**.

A five-layer neural network is designed to project time-frequency inputs into a low-dimensional embedding. The first three layers make use of 3D-convolutions, to take advantage of translation invariance, reduce the overall parameter space, and act as a constraint on the learning problem. Max-pooling is applied in time and frequency, to further accelerate computation by reducing the size of feature maps, and allowing a small degree of scale invariance in both directions. The final two layers are fully-connected affine transformations, the latter of which yields the embedding space. The first four hidden layers use a hyperbolic tangent as the activation function, while the visible output layer is linear, i.e. it has no activation function in the conventional sense.

Hyperbolic tangents are chosen as the activation function for the hidden layers purely as a function of numerical stability. It was empirically observed that randomly initialized networks designed with rectified linear units instead were near impossible to train; perhaps due to the relative nature of the learning problem, i.e. the network must discover an equilibrium for the training data, the parameters were routinely pulled into a space where all activations would go to zero, collapsing the network. Conversely, hyperbolic tangents, which saturate and are everywhere-differentiable, did not suffer the same fate. It is possible that the use of activation functions that provide an error signal everywhere, such as sigmoids or "leaky" rectified linear units (?, ?), or better parameter initialization might avoid this behavior, but neither are explored here.

Combining the successful application of linear "bottleneck" layers (?, ?) with lessons learned from previous efforts (?, ?), the visible layer is chosen here to be linear. As will be discussed in more detail shortly, a saturating nonlinearity at the output makes the choice of hyperparameters crucial in order to prevent the network from pushing datapoints against the limits of its space. However, the absence of boundaries allows the network to find the appropriate scale factor for the embedding.

Specifically, the network is parameterized thusly: the input to the network is a 2D tile of log-CQT coefficients with shape $(20, 192)$, corresponding to time and frequency respectively; the first convolutional layer uses 20 filters with shape $(1, 5, 13)$ and max-pooling with shape $(2, 2)$; the second convolutional layer uses 40 filters with shape $(20, 5, 11)$ and max-pooling with shape

$(2, 2)$; the third convolutional layer uses 80 filters with shape $(1, 1, 9)$ and max-pooling with shape $(2, 2)$; the fourth layer is fully-connected and has 256 output coefficients; the final layer is also fully connected, and has 3 output coefficients.

2.3 Pairwise Training

Briefly summarizing a previous point, there are currently no known quantities with which to measure timbre, in the same way that fundamental frequency has Hertz or loudness decibels. In the absence of this absolute reference, previous efforts have instead tried to determine the relative relationships between a collection of subjective ratings. Collecting this data subjectively for a large number of sources quickly becomes prohibitive, as the number of pairwise comparisons to be made increases quadradically with the total number of observations considered. Music instruments, however, provide an interesting source of objective information for this problem. Based on the coarse approximation that all sounds produced by a single instrument are in some sense similar, regardless of pitch or loudness, class boundaries can be used to define a neighborhood of similar timbres.

This approach to defining timbre "neighborhoods" can be used to extend the work of Hadsell et al (?, ?) to address this challenge of learning a timbre similarity space. Referred to by the authors as "dimensionality reduction by learning an invariant mapping" (DrLIM), a deep network was trained in a pairwise manner to minimize the distance between "similar" data points in a learned, nonlinear embedding space, and vice versa. Similarity was determined in an unsupervised manner by linking the k-nearest neighbors in the input space. Though left as future work, the authors propose that other information, such as class relationships, might be levereaged to learn different embeddings. This is an important consideration for the problem of timbre, because fundamental frequency and amplitude are likely to dominate the graph of nearest neighbors defined in the input space alone.

The intution behind DrLIM is both simple and satisfying: datapoints that are deemed "similar" should be close together, while those that are "dissimilar" should be far apart. Though the precise distance metric is a flexible design decision, it is used here in the Euclidean sense. A collection of similar and dissimilar relationships can be understood by analogy to a physical system of attractive and repulsive forces, where learning proceeds by finding a balance between them; and furthermore, this analogy illustrates the need for contrasting forces to achieve equilibrium.

At its core, DrLIM is ultimately a pairwise training strategy. First, a parameterized, differentiable function, $f(\dot{|}\Theta)$, e.g. a neural network, is designed for a given problem; in the case of dimensionality reduction, the output will be much smaller than the input, and typically either 2 or 3 for the purposes of visualization. During training, the function f is copied and parameters, Θ, *shared* between both, such that $f_1(\dot{|}\Theta) == f_2(\dot{|}\Theta)$. Two inputs, X_1 and X_2, are transformed by their respective functions, f_1 and f_2, to produce the outputs, Z_1 and Z_2. A metric, e.g. Euclidean, is chosen to compute a distance, D between these outputs. Finally, a similarity score, Y, representing the relationship between X_1 and X_2, is passed to a contrastive loss function, which penalizes similar and dissimilar pairs differently. When the pair is similar, the loss will be small when the distance is small; for dissimilar pairs, the loss will be small when the distance is outside a given margin, m. This formal definition is summarized symbolically by the following:

$$Z_1 = f_1(X_1|Theta), Z_2 = f2(X_2|Theta)$$

$$D = ||Z_1 - Z_2||_2$$

$$\mathcal{L}_{sim} = D^2$$

$$\mathcal{L}_{diff} = max(0, m_{diff} - D)^2$$

$$\mathcal{L} = Y * \mathcal{L}_{sim} + (1 - Y) * \mathcal{L}_{diff}$$

Note that similarity is given by $Y = 1$, for consistency with boolean logic. As a result, the first term of the loss function is only non-zero for similar pairs, and the inverse is true for the second term.

Returning to the previous discussion regarding the dynamic range of the output layer, it should now be clear that the choice of margin only influences the learned embedding relative to a scale factor when the output is unbounded. The two loss terms are mirrored parabolas, and changing the margin, or horizontal offset, only serves to shift the vertical line about which they reflect. The curvature, and thus the gradient, of the loss function is left unchanged. This observation encourages a simple generalization of this loss function, where a second margin is introduced to the “similar” loss term:

$$\mathcal{L}_{sim} = max(0, D - m_{sim})^2$$

$$\mathcal{L}_{diff} = max(0, m_{diff} - D)^2$$

$$\mathcal{L} = Y * \mathcal{L}_{sim} + (1 - Y) * \mathcal{L}_{diff}$$

Whereas the differential margin controls the spread of all points in space, the similar margin will control the spread of a similarity neighborhood. In the original formulation, where implicitly $m_{sim} = 0$, the loss is lowest when all inputs are mapped to *exactly* the same point; for the purposes of similarity, a more diffuse distribution of points is desirable. It is worth noting the slight parallel to linear discriminant analysis, a statistical method that seeks to jointly minimize intraclass variance and maximize interclass variance. Given the relative nature of this trade-off, it is sufficient to pick a single ratio between the margins, eliminating the need to vary both hyperparameters separately.

In practice, training proceeded via minibatch stochastic gradient descent with a constant learning rate, set at 0.02 for 25k iterations, or until a batch returned a total loss of zero. Batches consisted of 100 comparisons, drawn such that a datapoint was paired with both a positive and negative example.

3 Methodology

To assess the viability of data-driven nonlinear semantic embeddings for timbre similarity, and thus address the goals outlined at the outset of Section 2, two experiments are used to quantify different performance criteria. First, the local structure and class boundaries of the learned embeddings are explored with a classification task. Second, global organization of the space is measured by a ranked retrieval task. Additionally, in lieu of a subjective evaluation of perceptual "smoothness" of the resulting timbre space, the learned embeddings are investigated through confusion analysis and visualization. In each instance, the approach presented here is compared to a conceptually similar, albeit admittedly simpler, system.

Finally, the formulation described in the previous section presents two system variables, thus giving rise to two additional considerations:

1. What is the effect of using different margin ratios?
2. How does the sonic palette considered impact the learned embedding?

3.1 Data

The data source used herein is drawn from the Vienna Symphonic Library (VSL), a truly massive collection of studio-grade orchestral instrument samples recorded over a variety of performance techniques*. In aggregate, the VSL contains over 400k sound recordings from more than 40 different instruments, both pitched and percussive. Sorting instrument classes by sample count yields 27 instruments with at least 5k samples; three of these instruments, however, are not reasonably distinct from other sources, e.g. "flute-1" and "flute-2", and discarded rather than risk introducing conflicting information. This decision yields the set of instruments contained in Table **??** for experimentation.

The distribution of sound files for these instruments, grouped by class, is given in Figure 9. As discussed previously, it is an inherent difficulty of

* `https://vsl.co.at/en`

Table 2

Instruments considered and their corresponding codes.

Instrument	Code	Instrument	Code
French Horn	ho	Tuba	tu
Violin	vi	Cimbasso	ci
B♭ Clarinet	klb	Piccolo	pt
Tenor Trombone	tp	Oboe	ob
C Trumpet	trc	Bass Clarinet	bkl
Bass Trombone	bp	Wagner Tuba	wt
Acoustic Concert Guitar	akg	Contra Bassoon	kfa
Bassoon	fa	English Horn	eh
Cello	vc	Bass	kb
Bass Trumpet	bt	Soprano Saxophone	sxs
Distorted Guitar	eg	Tenor Saxophone	sxt
Flute	fl	Alto Flute	afl

pairwise similiarty models that the resulting relationships are limited by the number of unique classes considered. Fortunately, there is no added cost to considering a wider palette of sound sources here because the label information is objective. Therefore, building upon previous work (?, ?), three configuration subsets are repeated from the pilot study as well as a fourth consisting of all 24 classes, given in Table **??**.

For each instrument class, 5k samples are drawn, without replacement, to build a uniformly distributed collection. This step simplifies the process of data sampling during stochastic training of the network, which may be sensitive to class imbalances. The collection of instrument samples is stratified into five equal partitions for cross validation, used at a ratio of 3-1-1 for training,

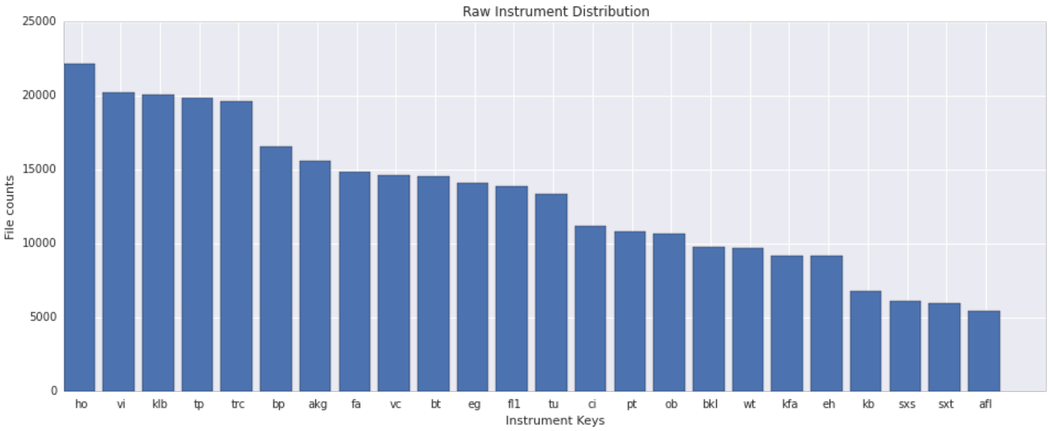


Figure 9: Distribution of instrument samples in the Vienna Symphonic Library.

Table 3

Instrument set configurations.

Key	Instrument Codes
c5	tu, ob, klb, vc, fl
c8	trc, ho, ob, eh, klb, sxt, vi, vc
c12	c8 + {tp, tu, fa, fl}
c24	c12 + {bp, akg, bt, eg, ci, pt, bkl, wt, kfa, kb, sxs, afl}

validation, and testing, respectively. The partitions are circularly rotated such that each is used as the test set once, i.e. (1, 2, 3)-4-5, (2, 3, 4)-5-1, and so on.

3.2 Margin Ratios

Though the pairwise training strategy described in Section 2.3 consists of two margin hyperparameters, it is ultimately the ratio between the two that governs how the space will be shaped. In isolation, the exact choice of dissimilar term's margin, m_{diff}, is inconsequential and determines the radius of the bounding sphere. Going forward, this value is fixed to $\sqrt{12}$, corresponding to the radius of the sphere that intersects the coordinate $(2, 2, 2)$. Moving the similar term's margin, m_{same}, relative to this value will lead to different embeddings, and three ratios of $m_{same} : m_{diff}$ are considered here: 0, $\frac{1}{4}$, and $\frac{1}{2}$. The corresponding loss functions are shown in Figure 10.

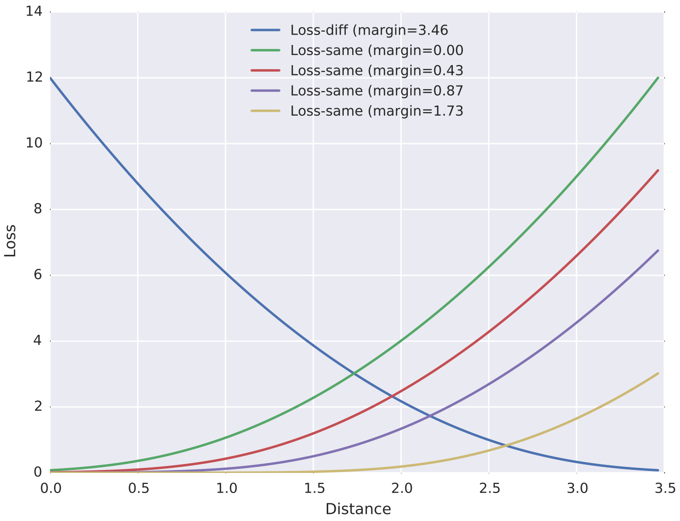


Figure 10: Distribution of instrument samples in the Vienna Symphonic Library.

3.3 Comparison Algorithm

For the purposes of comparison, a similarly motivated system is constructed using the combination of principal components analysis (PCA) and linear discriminant analysis (LDA). Previous work explored the application of PCA alone and locally linear embedding as alternative approaches to dimensionality reduction. Both are unsupervised methods, and do not make for the most fair comparison against a supervised neural network. LDA, however, is a supervised approach to dimensionality reduction, and shares at least a conceptual parallel to the proposed system, as mentioned briefly in Section 2.3.

It is important to note though that LDA can exhibit odd behavior in high dimensional spaces, and projecting into a PCA subspace first can help alleviate these issues (?, ?). This subspace projection is further motivated by computational efficiency concerns, where the input dimensionality is prohibitive to training. Additionally, the cascade of PCA followed by LDA mimics a two-layer neural network, and is interchangable with the framework described here. Using the same input dimensions, 20 × 192), a whitened PCA transform is fit to a large sample of the training set. The principal 256 components are preserved, based on an empirical exploration of the explained variance as well as a midway point in the dimensionality reduction of the system, i.e. the number of cofficients decreases near-equally between the PCA and LDA stages. After applying the PCA transform to the training sample, an LDA transform is fit to the same data and its corresponding instrument classes, yielding a 3-dimensional embedding.

3.4 Experimental Results

As an initial quantitative inquiry, trained models are tested on a classification task using the k-Nearest Neighbors classifier in scikit-learn[*]. First, networks are trained across the 4 instrument configurations, 3 margin ratios, and 5 folds, and all data are projected into the resulting embedding space. From here, a collection of points are sampled from each parition –50k, 10k, 25k– for training, validation, and test, respectively. The training set is used to fit the classifier, while the validation set is used to identify an optimal setting for the parameter k, corresponding to the number of neighbors considered in the

[*] http://scikit-learn.org/stable/

decision rule. Classification accuracy is then computed across all three sets, and tallied across folds to produce averages and standard deviations across the various conditions; these results are given in Tables 4-6.

Table 4

k-Neighbors classification results over the training set.

config	c5	c8	c12	c24
NLSE : 0.0	93.81 ± 0.53	89.97 ± 0.40	86.70 ± 0.39	74.17 ± 0.79
NLSE : 0.25	94.21 ± 0.18	90.25 ± 0.54	87.17 ± 0.35	73.91 ± 0.61
NLSE : 0.5	93.04 ± 0.29	89.16 ± 0.27	86.08 ± 0.26	71.59 ± 0.88
PCA-LDA	64.44 ± 0.47	56.58 ± 0.60	47.22 ± 0.45	35.81 ± 0.28

Table 5

k-Neighbors classification results over the validation set.

config	c5	c8	c12	c24
NLSE : 0.0	92.37 ± 0.64	87.94 ± 0.45	84.52 ± 0.97	70.86 ± 1.51
NLSE : 0.25	93.75 ± 0.53	88.62 ± 0.58	85.65 ± 0.26	71.46 ± 0.63
NLSE : 0.5	91.75 ± 0.67	87.78 ± 0.85	83.78 ± 0.53	66.37 ± 1.69
PCA-LDA	59.97 ± 0.96	52.49 ± 2.68	39.25 ± 2.01	24.32 ± 0.97

A few conclusions are immediately obvious from these results. Most striking is the performance discrepancy between the NLSE and the PCA-LDA models. Previous work demonstrated a significant margin between the unsupervised dimensionality reduction methods, and this result shows that the difference is indeed a function of complexity, not just the supervised learning process. To a lesser extent, all models show some degree of over-fitting, but the effect is more severe for the PCA-LDA model than any NLSE. Somewhat surprisingly, using a non-zero similiarty margin leads to slightly better clas-

Table 6

k-Neighbors classification results over the testing set.

config	c5	c8	c12	c24
NLSE : 0.0	92.49 ± 0.41	88.26 ± 0.74	84.35 ± 0.41	70.67 ± 0.55
NLSE : 0.25	92.97 ± 0.41	88.67 ± 0.79	85.16 ± 0.24	70.28 ± 0.86
NLSE : 0.5	91.96 ± 0.35	87.83 ± 0.25	84.04 ± 0.57	66.91 ± 0.57
PCA-LDA	59.91 ± 0.77	50.24 ± 1.52	39.32 ± 0.86	24.77 ± 0.51

sification results than the centered loss function. One explanation for such behavior is that introducing a small region of zero-loss within a class may allow the network to emphasize dissimilar relationships more as training proceeds. It would appear too much freedom, on the other hand, leads to fuzzy boundaries between classes and begins to compromise local structure.

The outcome of the classification experiment can also be used to inform how smooth or intuitive this space might be. To do so, confusion matrices are shown for the c12 configuration for the best NLSE, with a margin ratio of 0.25, and the PCA-LDA model, in Tables 7 and 8, respectively.

Table 7

Confusion Matrix for c12; NLSE with a margin ratio of 0.25.

	eh	fa	fl	ho	klb	ob	sxt	tp	trc	tu	vc	vi
eh	**85.52**	1.10	0.46	3.05	1.76	3.05	0.44	0.23	2.53	0.30	0.51	1.64
fa	1.74	**85.82**	0.05	3.93	0.26	0.19	0.63	0.81	0.51	3.64	2.48	0.51
fl	0.80	0.10	**85.20**	0.90	2.19	6.00	1.67	0.14	2.33	0.07	0.33	1.17
ho	0.76	1.88	0.07	**82.52**	0.26	0.29	0.33	6.50	1.18	2.73	0.88	1.26
klb	1.23	0.40	2.46	1.16	**86.57**	3.02	1.80	0.11	1.01	0.25	1.37	1.05
ob	2.90	0.06	3.09	1.12	2.56	**81.22**	0.44	0.17	5.29	0.04	0.04	1.39
sxt	0.24	0.38	1.01	0.51	1.24	0.84	**86.34**	0.14	0.48	0.65	4.97	2.78
tp	0.39	0.87	0.10	11.87	0.03	0.49	0.20	**80.96**	2.38	2.73	0.95	0.59
trc	1.14	0.11	1.77	3.84	0.56	4.13	0.59	1.74	**83.45**	0.08	0.09	2.47
tu	0.04	1.55	0.04	5.32	0.04	0.01	0.57	2.18	0.09	**86.44**	2.82	0.57
vc	0.27	0.53	0.24	1.51	0.79	0.49	2.68	0.27	0.63	2.32	**89.74**	1.49
vi	0.49	0.23	0.61	2.44	0.46	1.20	2.46	0.48	2.05	0.41	2.06	**87.32**

Table 8

Confusion Matrix for c12; PCA-LDA.

	eh	fa	fl	ho	klb	ob	sxt	tp	trc	tu	vc	vi
eh	**46.10**	3.01	11.27	6.79	2.61	10.92	0.21	9.69	9.85	0.79	1.20	1.04
fa	5.12	**46.84**	0.37	14.42	0.31	0.27	1.88	7.55	0.58	17.15	2.47	0.57
fl	13.33	1.62	**20.82**	6.93	4.01	17.74	1.79	4.26	16.32	1.63	2.10	1.85
ho	5.45	19.49	2.11	**43.53**	1.77	0.29	0.94	15.47	1.38	7.51	1.53	4.70
klb	10.80	4.88	11.69	10.62	**9.04**	9.61	4.84	5.47	11.99	6.51	7.96	5.07
ob	15.45	0.40	17.80	1.67	3.09	**31.93**	0.56	2.42	19.92	0.69	0.81	2.14
sxt	0.48	2.77	2.28	3.62	2.33	0.97	**47.47**	1.40	3.45	9.48	14.48	15.77
tp	15.98	9.58	6.45	19.43	2.02	5.74	0.52	**18.93**	9.42	4.33	1.15	2.07
trc	10.72	0.20	14.14	3.64	3.50	19.71	0.77	5.67	**36.01**	0.27	1.12	4.97
tu	0.04	12.39	0.06	7.13	0.77	0.03	4.35	3.68	0.03	**62.74**	7.67	0.26
vc	0.39	1.65	2.87	2.66	2.09	2.36	18.99	1.16	4.29	10.02	**44.95**	12.05
vi	0.93	1.78	2.63	5.04	1.62	3.22	13.43	1.44	7.86	1.43	4.49	**56.47**

Though more confusions are to be expected in the PCA-LDA model, given the classification accuracy, it is important to note that these errors are distributed across all classes, regardless of pairwise relationships. This higher noise-floor indicates that the instruments' distribution exhibit a good deal of overlap in space. Some logical confusions seem unavoidable, such as french horn (ho) and trombone (tp), or flute (fl) and oboe (ob), occuring in both models. The former makes sense given common instrument families, i.e. brass, while the latter likely arises from the upper range of the instruments, which has fewer harmonics.

Other instrument relationships also appear to confound some element of pitch height in similarity, particularly for the PCA-LDA model. This is observed in the confusions between tuba, bassoon, and French horn. In the NLSE model, tuba is confused with French horn more often than bassoon; for the PCA-LDA model, however, the inverse is true. Intutively, the two brass instruments should share the higher confusion rate, and thus pitch is being used by the LDA model as a feature with which to distinguish between classes. The convolutional model, on the other hand, is forced to embrace a considerable amount of pitch invariance, and is prevented from making the same error.

To help demonstrate the semantic organization of the learned embedding, 3D scatter plots are given in Figure 11 following observations of the three instruments common to all configurations –clarinet, oboe, cello– across the different embeddings for $m = 0.25$. Other instruments are displayed as semi-transparent black, to clearly highlight the three instruments of interest while giving an impression of the overall space. The main takeaways from this visualization are two-fold. First, the various sonic palettes used to learn the embedding result in different organizations of points in space. That said, the relationship between the three sources is relatively consistent, as the cluster of clarinet always sits between oboe and cello.

To further test the semantic organziation of the learned embeddings, the outputs are used as features for a ranked retrieval task, using Euclidean distance as a scoring function. Recall-precision curves are computed using the scikit-learn toolkit and averaged over the five folds; the resulting curves for the four configurations are shown in Figure 12.

Given the classification accuracy and confusion matrices above, the performance gap between the NLSE and PCA-LDA models is unsurprising. Still, it is interesting to consider what the shape of these recall-precision curves indicates. The two characteristics to observe are the concavity of the contour and the "knee" at which it breaks downward. In all instrument configurations, with the slight exception of "c5", the NLSE models and the PCA-LDA model exhibit opposite second-derivatives. This behavior can be understood as the acceleration with which precision changes as a function of recall. For the NLSEs, precision degrades slowly until reaching a crossover point, referred to here as the knee, where precision drops off rapidly. The PCA-LDA model does the opposite, where precision drops quickly close to a query point, and slows as recall increases. Therefore, as encouraged by the visualizations, the NLSEs contain better separated class clusters than the PCA-LDA embeddings. Furthermore, the knee of a recall-precision curves belies an interesting region in the document space, indicating that the edge of a cluster has been reached. This is a useful observation for determining early-stopping criteria in the display of ranked results, as well as identifying boundary regions in the embedding that may present interesting opportunities for sonic exploration.

4 Conclusions

In this chapter, an approach to building a computational model of timbre similarity was presented, which achieved three goals. First, the system is able to automatically learn relevant signal level features, circumventing the challenge posed by the lack of a constructive definition of timbre. Second, the resulting timbre space is semantically well-organized, as demonstrated by classification and ranked retrieval measures, and intuitive, based on a Euclidean notion of distance in a dimensionality that can be easily visualized. Lastly, the space is quantitatively smooth, such that what confusions exist correspond to instrument families or other like sounds. This was made possible by leveraging objective information about a collection of sound sources, eliminating the need for costly subjective ratings. Together, this approach to learning a timbre space shows promise for visualization and user-facing applications, such as the search and navigation of large sound libraries.

That said, there is considerable future work to be considered. All evaluation performed here is quantitative, and arguably disconnected from all subjective experience. User studies would serve to further investigate the ideas of perceptual smoothness and if or how is obtained by the learned space. Additionally, though this approach is able to make use of objective instrument taxonomies, any similiarity space obtained through pairwise comparisons is always limited by the range of inputs considered. Therefore, in order to obtain a more general timbre space, a much wider set of sound sources would need to be considered. Conversely, the intended use case of such a system may provide a constrained palette with which to operate, e.g. instrument sounds for recording engineers and environmental sounds for computational ecologists.

Finally, there are at least two other ways the sound source information could be used to train a system in a supervised manner. One, it may be advantageous to obtain subjective pairwise ratings not between all possible sounds, but rather groups or classes of sounds. These pairwise ratings could be used to train a system with soft, continuous-valued similarity ratings, rather than the binary comparison scores used here. Two, rather than defining an entire class to be similar, a hybrid approach to similiarity based on distance-based neighborhoods in the input space constrained to a single class may also lead to interesting embeddings. It is unlikely such an embedding would exhibit spherical clusters as was produced here, but points are likely to be more uniformly distributed, or diffuse, in space.

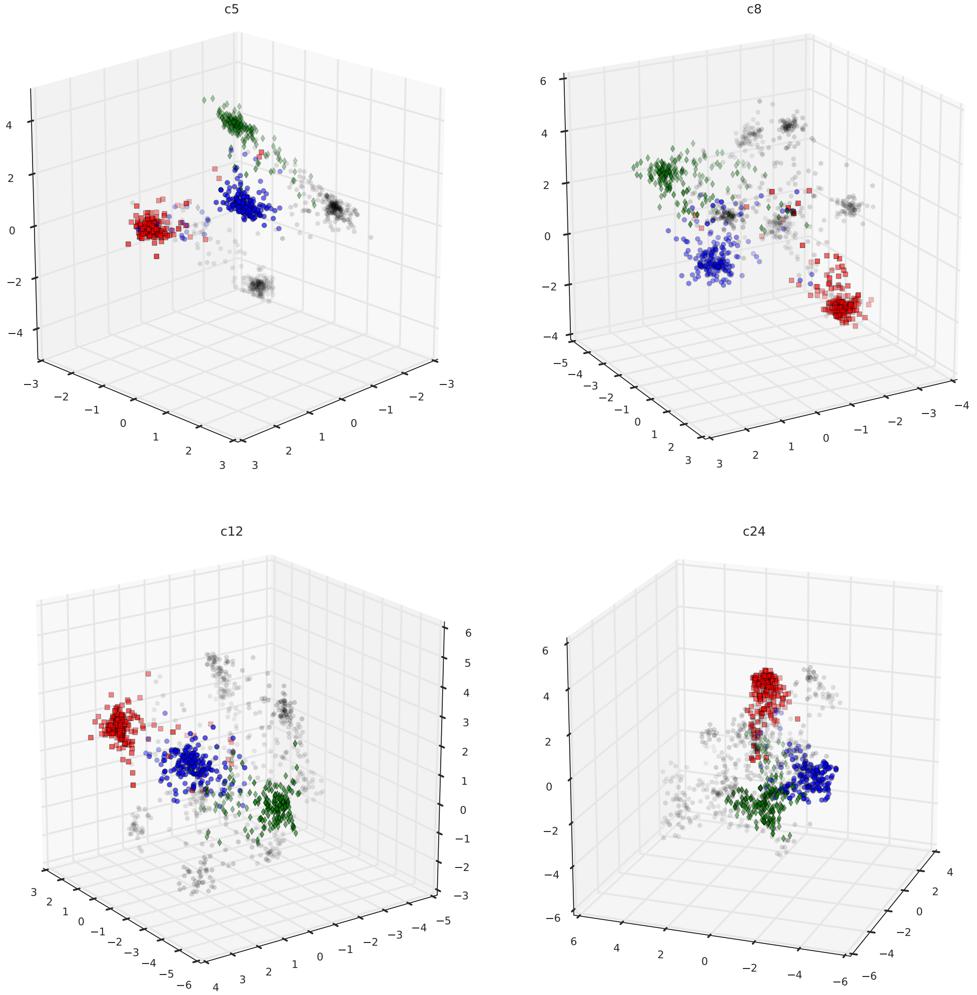


Figure 11: Embeddings of clarinet (blue circles), oboe (green diamonds), and cello (red squares) observations across models trained with the four different instrument configurations.

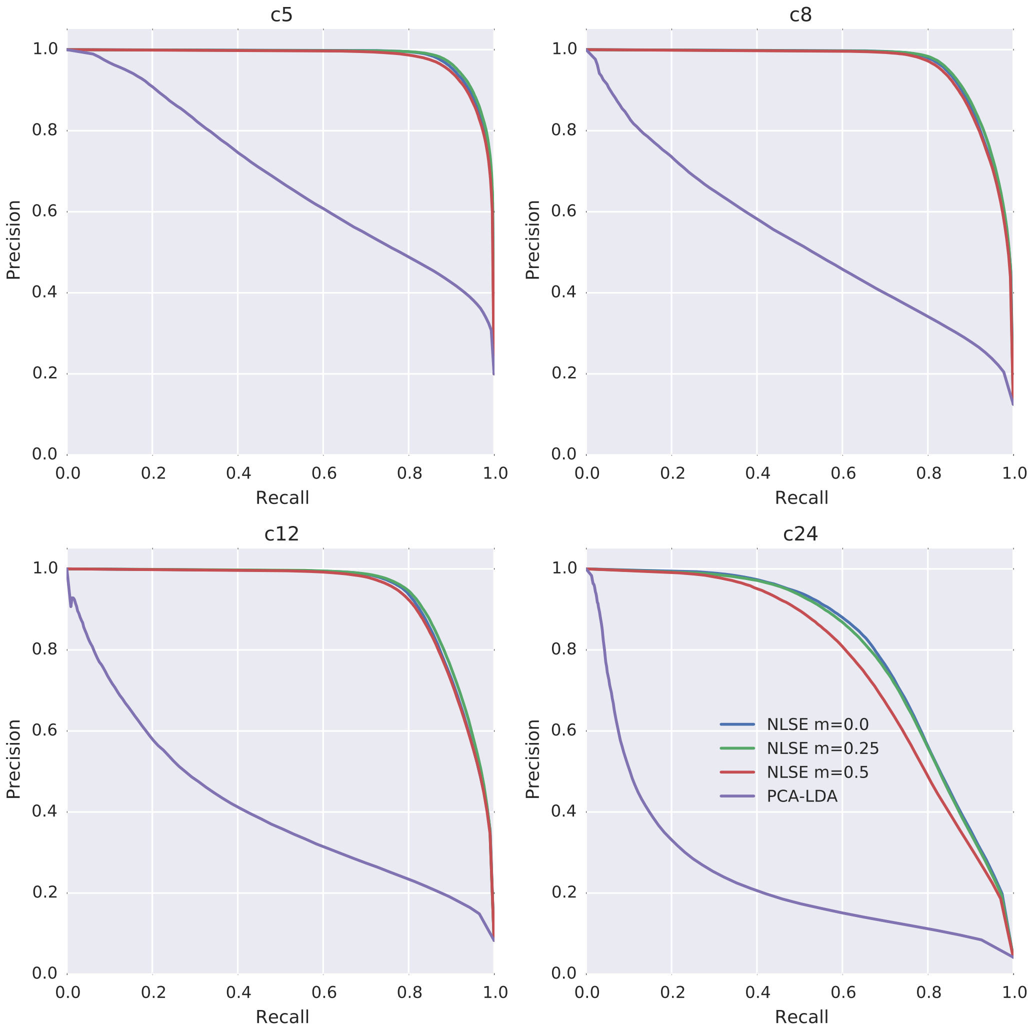


Figure 12: Recall-Precision curves over the four instrument configurations.

CHAPTER V

AUTOMATIC CHORD ESTIMATION

The previous chapter was able to reformulate timbre similarity as a well-defined task that used objective information for development and evaluation. Automatic chord estimation, one of the oldest subtopics in the field of content-based MIR, embodies nearly the polar opposite scenario, being rather ill-defined and depending on subjective information almost exclusively. As such, it provides an interesting case study with which to test the limits of deep learning. In addition to the standard objective of looking to advance the state of the art in a chosen application domain, this chapter seeks to explore the limitations of deep learning as applied to complex music intelligence task. A thorough investigation serves to not only offer insight into the application of deep learning methods to future problems in content-based MIR, but also culminate in a better understanding of the chord estimation task itself.

1 Context

In this section, the prerequisite knowledge necessary to appreciate a thorough treatment of automatic chord estimation systems is addressed. First, a basic review of Western tonal theory sets out to define key music concepts, and thus provides a language with which to discuss the problem. This is followed by a brief outline of the syntax used here to compactly notate chords. The problem domain is then jointly motivated by potential use-cases and the observed needs of online music communities. Finally, known limitations are detailed to contextualize both assumptions and any conclusions drawn from this work.

1.1 Musical Foundations

To understand the chord estimation task is to understand the concepts on which it is built. The most atomic unit in the acoustic realm of chords is a *tone*, defined here as a pitched sound object. Pitch is defined as the perceptual phenomena whereby a sound stimulus can be matched to the frequency of a pure sinusoid, known as the fundamental frequency. As a result, sounds can be naturally ordered by fundamental frequency on a scale from low to high. An *octave* is defined as the doubling of a quantity, and exhibits a special relationship in human perception by which two tones, differeing in fundamental frequency by an integer power of 2, are perceived as being similar; this phenomena is referred to as octave equivalence (?, ?).

A *note* is the musical counterpart of a tone, and the two are related by way of a tuning system. While there are a multiplicity of tuning systems, this work will exclusively consider 12-Tone Equal Temperament, or 12-TET, named for the 12 discrete, equally spaced tones between within an octave. The relationship between notes and tones in 12-TET is defined by the following:

$$f_{pitch} = f_{tuning} * 2\exp(\frac{n_{index} - 48}{N}) \qquad (2)$$

where N is the number of notes per octave, n_{index} is an integer note index, f_{tuning} is the reference frequency, in Hertz, on which , and f_{pitch} is the fundamental frequency, in Hertz, of the corresponding tone. In 12-TET, the *unique pitch classes* within an octave are named, given by the ordered set, $\mathcal{P}$:

$$\mathcal{P} = \{C, C\sharp/D\flat, D, D\sharp/E\flat, E, F, F\sharp/G\flat, G, G\sharp/A\flat, A, A\sharp/B\flat, B\} \qquad (3)$$

Here, sharps, $\sharp$, and flats, $\flat$, are symbols used to indicate the raising or lowering of a note by one semitone, respectively. Note that due to this particular tuning system, some pitch classes can be spelled with either a sharp or a flat, e.g. $A\sharp = B\flat$, a property known as *enharmonic equivalence*. An absolute note name, consisting of a pitch class, p, and an octave, o, is given by the following as a function of absolute note index, such that $n_{index} = 0 \rightarrow C0$, $n_{index} = 12 \rightarrow C1$, etc.:

$$p = \mathcal{P}[mod(n_{index}, 12)]$$
$$o = \lfloor\ n_{index}/12\ \rfloor$$

In contemporary music, standard concert tuning equates $A4 = 440Hz$; classically, the tuning frequency is defined such that it falls in the performance range of a majority of instruments to be matched exactly, and hence the offset in Eq. 2. Were the tuning system and reference frequency universally constant, the concepts of tones and notes would be equivalent; it should be noted, however, this is not the case.

Most[*] real music combines many different notes, and thus it is useful to define an *interval* is defined as the relative semitone distance between two notes. Whereas notes are atomic, intervals are the bonds in a harmonic

[*] But not all, as in the case of John Cage's *4'33"*, which contains none.

molecule. The interval between consecutive pitch classes is referred to as a *semitone*, with a step size of 1, and a step size of 2 is known as a *whole tone*. Intervals are critical to music perception because they are generally perceived as similar regardless of pitch height, e.g. the interval from C4 to G4 sounds the same as the interval from F4 to C5, both being seven semitones.

From here, three interdependent harmonic concepts are built through the simultaneous and sequential combination of notes and, as a result, intervals: scales, chords, and keys. It is crucial to recognize that each is an emergent quality of the same fundamental parts, and efforts to define one in terms of another results in circular logic. For clarity, the relationships between the various entities discussed so far are diagrammed hierarchically in Figure **??**.

That said, an ordered set of intervals in known as a scale, of which the *diatonic* is the most widely used in common practice music. It consists of eight semitone scale degrees, and thus seven intervals, given by the following:

$$\{2, 2, 1, 2, 2, 2, 1\}$$

Rotating this sequence circularly results in different *modes*, of which two are common in contemporary popular music: *major*, with a rotation of 0; and *minor*, with a rotation to the right of 3. Starting from 0, the semitone values of these scales are expressed by the following:

$$\texttt{maj} = \{0, 2, 4, 5, 7, 9, 11, 12\}$$
$$\texttt{min} = \{0, 2, 3, 5, 7, 8, 10, 12\}$$

These sequences can be used to recover a scale by circularly indexing the set of pitch classes given in 3. The starting pitch classon which the scale is based is referred to the *tonic*, and lends its name to the scale. To illustrate, a C Major and A minor scale are given as follows:

$$\mathcal{C}_{major} = \{C, D, E, F, G, A, B\}$$
$$\mathcal{A}_{minor} = \{A, B, C, D, E, F, G\}$$

One should observe that these two scales, with different modes and tonics, are composed of identical notes. These scales are known as each other's relative major and minor, respectively, and share a strong perceptual affinity.

Proceeding from scales, a *chord* is classically conceived as a simultaneous grouping of notes. One of the most important chord types in Western tonal theory is the *triad*, on which many other theoretical constructs are built. Comprised of three notes, a triad is built by taking the second and fourth scale degrees from a chosen starting point, referred to as the root. For example, this expansion is given for the C-major scale in Table 9.

For a given chord, the *root* is defined as the home note on which the

Table 9

Roman numeral, quality, semitones, and intervals of triads in the Major scale.

Roman Numeral	Quality	Semitones	Intervals
I	Major	{0, 4, 7}	{+4, +3}
ii	minor	{2, 5, 9}	{+3, +4}
iii	minor	{4, 7, 7}	{+3, +4}
IV	Major	{5, 9, 12}	{+4, +3}
V	Major	{7, 11, 2}	{+4, +3}
vi	minor	{9, 0, 4}	{+3, +4}
vii°	diminished	{11, 2, 5}	{+3, +3}
I	Major	{0, 4, 7}	{+4, +3}

intervals are stacked, and the *quality* determined by the relationship of the intervals to the root. An interval of 4 semitones is known as a *major third* and 3 semitones a *minor third*, sharing a commonality with the third scale degree of the major and minor scales, respectively. Therefore, the qualities of the first six chords in the table are named for their relationship with the corresponding major and minor scales; the vii° chord, however, is named differently because it contains two minor thirds.

Sharing much common ground with scales and chords, a *key* identifies a point of harmonic stability in reference to a tonic and its corresponding triad. Accordingly, the keys conventionally take one of the pitch classes and either a major or minor quality, e.g. $E - major$ or $C\sharp - minor$. Key is integral to the discussion here, because its impression or expectation establishes a harmonic framework with which to parse musical information, leading to the understanding of notes and intervals as they relate to scales and chords.

While much of this theory can be detailed specifically, real music is by no means so well behaved. As a result, the practical definition of a “chord” is open to some interpretation, and may take multiple meanings. For example, (?, ?) collects three possible definitions, restated here:

1. *Everyone agrees that* chord *is used for a group of musical tones.*
2. *Two or more notes sounding simultaneously are known as a chord.*
3. *Three or more pitches sounded simultaneously or functioning as if sounded simultaneously.*

Additionally, (?, ?) expands the scope of (2) in order to describe all tonal music, “allow[ing] a chord to comprise zero or more notes.” The various flavors of defintions begs an obvious question: what makes the concept of a chord so hard to pin down?

Much of this difficulty stems from the fact that the relative importance of the individual notes in a collection may change in different contexts. In practice, a chord is named based on three subjective criteria: its root, its contributing intervals, and how these two relate to the key. Therefore, as will be shown shortly, a chord can easily take a different name if any of these decisions are changed or reweighted.

Having briefly reviewed Western tonal theory, a deeper understanding of the variation inherent to defining a chord can be obtained by exploring a few simple examples. The one invariant property shared by all defintions named previously is the idea that a pitch collection may be understood as a single harmonic object. The time span over which this phenomena may unfold, however, is flexible. To illustrate the point, consider the three bars notated in Figure 13, where a major chord is written as a true simultaneity,

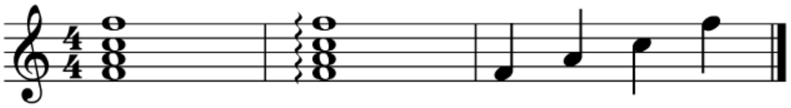


Figure 13: writeme

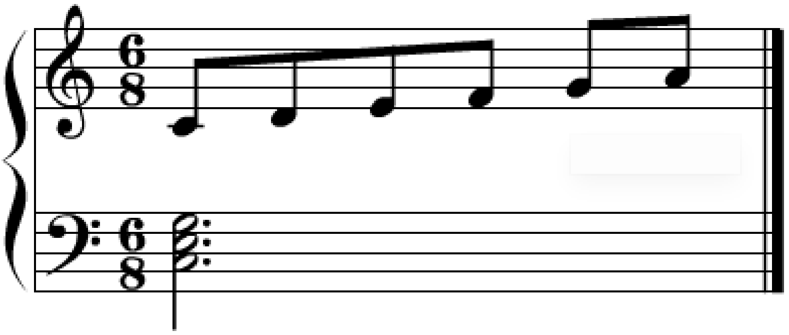


Figure 14: writeme

an arpeggiation, and as an series of non-overlapping quarter notes, respectively. In this instance, the degree of overlap in time is expanded until it no longer exists, and yet the collection of notes continues to function as a coherent harmonic object.

On the other hand, as shown in Figure 14, the simultaneous sounding of different notes does not necessarily give rise to the perception of different chords. Here, a major triad is sustained under the first several degrees of its scale. While three notes in the upper voice are contained in the C-major triad, the others –the D, F, and A– are referred to as “nonchord” tones. These extra notes are explained away in the overall harmonic scene, as they fall on metrically weak beats, are comparatively short in duration, and quickly move to notes that *are* in the chord. These embellishments do not contribute to the harmonic center of the phrase, and the bar can still be understood as a stable C-major chord.

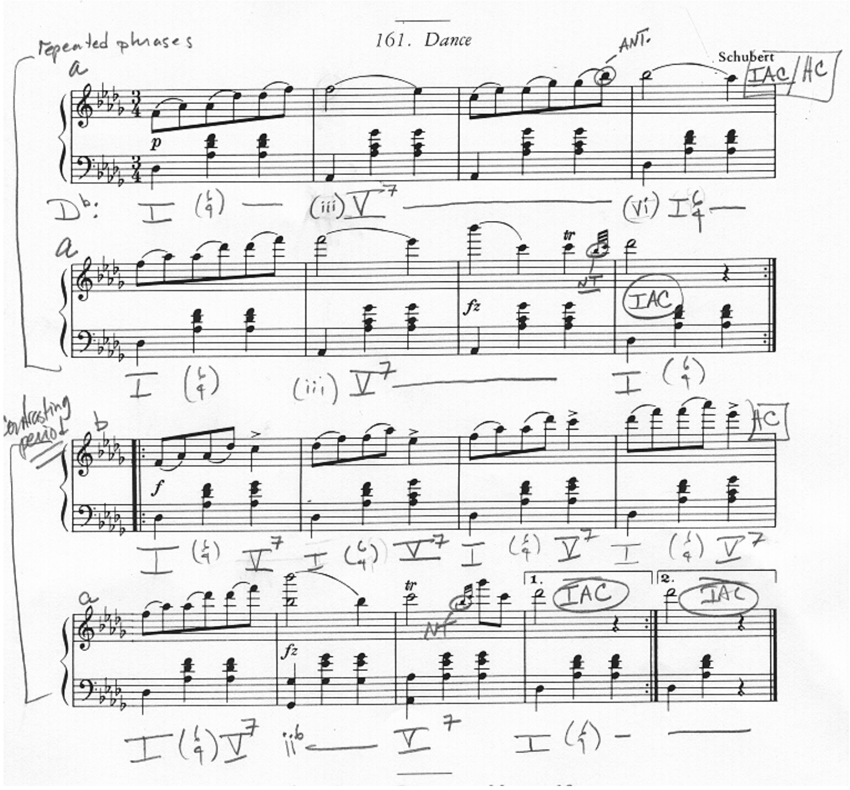

repeated phrases
161. Dance
ANT.
Schubert
IAC/HC
a
p
D♭: I (6/4) — (iii) V7 — (vi) I6/4 —
a
tr
NT
IAC
fz
I (6/4) (iii) V7 — I (6/4)
contrasting period
b
f
HC
I (6/4) V7 I (6/4) V7 I (6/4) V7 I (6/4) V7
a
fz
tr
NT
1.
IAC
2.
IAC
I (6/4) V7 ii6 — V7 I (6/4) — —

Figure 15: A sample harmonic analysis, performed as a music theory exercise.

A last example, shown in Figure 15, demonstrates the level of complexity and inherent subjectivity that may arise in the process of describing real music with chords. Referred to as *harmonic analysis*, this exercise is performed in an effort to understand the harmonic content in a theoretical manner. Here, an excerpt of a score has been analyzed, with chords notated under the original parts. Even in this instance, where the individual is able to operate directly on the symbolic representation, it is common to find alternate interpretations of the same musical content. Conversely, annotations resulting from sound recordings, as opposed to a score, are referred to as *chord transcriptions*. Often the recording itself is the only musical artifact to consider, introducing even more room for subjectivity.

1.2 Chord Syntax

It is a pragmatic, but ultimately necessary, preliminary step to define a standard syntax for compactly notating chords. Much of the pioneering work in this space was performed by Harte (?, ?), and many of these conventions are utilized here. Going forward, chords expressed in this scheme are stylized with a fixed-width font, e.g. `A:min`.

Firstly, Harte's general chord notation is described by the following four-part symbolic description:

$$\{r\} : \{q\}(i)/\{b\} \tag{4}$$

Every chord name begins with a root r in the form of a pitch class, optionally modified by zero or more sharps or flats, or one of two reserved characters: `N` for the "null" no-chord condition, or `X` for the special case in which the musical content cannot be described harmonically.

The root is potentially followed by a quality shorthand, q, separated by a colon and implying a particular set of note intervals. Though there are a large number of possible chord qualities, this is often limited to a particular subset in practice. Those considered in this work are indicated in the following table:

The third field provides an interval set, i, wrapped by parentheses. In practice, there are two reasons for representing information intervallically. One such instance is, through a combination of additional degrees and asterisks, the modification of a quality shorthand in order to represent a non-standard, but related, chord. An example of this might be the chord name `A:min(b3, b7)`, which means that the minor third (C) is absent, but a minor 7 (G) has

Table 10

Chord quality names and corresponding relative semitones.

Name	Shorthand	Semitones
Major	`maj`	$\{0, 4, 7\}$
Minor	`min`	$\{0, 3, 7\}$
Major 7	`maj7`	$\{0, 4, 7, 11\}$
Minor 7	`min7`	$\{0, 3, 7, 10\}$
Dominant 7	`7`	$\{0, 4, 7, 10\}$
Major 6	`maj6`	$\{0, 4, 7, 9\}$
Minor 6	`min6`	$\{0, 3, 7, 9\}$
Diminished	`dim`	$\{0, 3, 6\}$
Augmented	`aug`	$\{0, 4, 8\}$
Suspended 2^{nd}	`sus2`	$\{0, 2, 7\}$
Suspended 4^{th}	`sus4`	$\{0, 5, 7\}$
Fully-diminished 7	`dim7`	$\{0, 3, 6, 9\}$
Half-diminished 7	`hdim7`	$\{0, 3, 6, 10\}$

been added. The other instance occurs when the intervals are certain but the spelling is ambiguous, such as `C:(1, 5)`.

The final field of this chord syntax is the bass interval, b, which indicates the scale degree at the bottom of the chord. Typically this is also the root of the chord, and is implied as such in the absence of an explicit bass interval. However, it is necessary to state that the scale degrees of the chord –given by the quality shorthand and the interval set– can be further augmented by the inclusion of a bass interval. For example, the chords `C:maj/b7` and `C:7` would be understood as containing the same pitch classes, but are spelled differently.

1.3 Motivation

Even from the earliest efforts in content-based music informatics research, automatic music transcription has stood as one of the Holy Grails of the field. Over time, however, it would prove exceptionally difficult, and fracture into a variety of smaller, and hopefully more manageable, subtopics. As a result, automatic chord estimation materialized as one such task, now receiving healthy attention for more than a decade, and established as a benchmark challenge at the annual MIReX event [*].

Given the prerequisite skill necessary to produce these transcriptions manually, there is considerable motivation to develop automated systems capable of reliably performing this task. As evidenced by large online communities surrounding websites like e-chords[†] or Ultimate Guitar[‡], countless individuals invest considerable time and effort in the curation and consumption of popular music transcriptions. Often this is driven by desire to learn and perform music for which symbolic notation does not exist. Conversely, automatic chord transcription systems would be particularly useful in the areas of composition, recording, and production. Furthermore, the curation of this content would enable large-scale musicological analysis of contemporary music.

In addition to the concerns of individual users, computational systems capable of reliable chord transcription are directly useful inside the domain of content-based MIR. Chords can serve as a robust mid-level representation with which to build systems and extract higher level musical knowledge. This has

[*] http://www.music-ir.org/mirex/wiki/MIREX_HOME

[†] http://www.e-chords.com

[‡] http://www.ultimate-guitar.com

been used in the space of cover song retrieval (?, ?), navigating large collections (?, ?), and genre recognition (?, ?). Such systems would also facilitate data collection for other tasks, aiding in visualization and other facets of music transcription.

From a more philosophical perspective, the identification of chords is also intriguing as an intelligent musical behavior, being a high level cognitive process that is often open to multiple interpretations between knowledgable experts. Experiential bias of the annotator may manifest in the subjective decisions made by an observer, where a pianist may arrive at a different harmonic analysis than that of a guitarist due to how a collection of notes might be produced. Lastly, the knowledge and skill of the one recognizing chords in music will affect the resulting interpretations. Beginners will likely prefer simple or more common descriptions, whereas experts will be more aware of complex nuances and have better command over a larger harmonic vocabulary.

1.4 Limitations

It should be acknowledged that such an inquiry is subject to certain limitations, and there are several that should be discussed. First and foremost, this work is primarily interested in the tradition of tonal Western music, with a particular focus on popular contemporary music from the last century, in 12-TET. Within this large body of content, the use of harmony and the language of chords has steadily evolved over time. Contextualizing briefly, music theorists have long sought to characterize musical works via analysis and reduction as a means to understanding. With sound recording being a relatively modern invention on the timescale of music history, much effort to this end has been invested in the analysis of symbolic notation, or scores. From the counterpoint of J. S. Bach to the functional analysis of Heinrich Schenker or more recently Fred Lehrdal, traditional musical analysis revolves heavily around the harmonic facets of a musical piece by marginalizing the dimensions of rhythm and timbre. As a result, the harmonic language of "chords" developed as an expressive, yet compact, vocabulary with which one might describe a piece of music harmonically. Western pop music, however, is not wont to consider the rules of conventional tonal theory or compose by traditional means. Therefore efforts to understand the former in the language of the latter is ultimately limited by the validity of this applied knowledge.

This realization encourages due consideration of the philosophical limits of objective truth in an inherently subjective task. As outlined previously, the definition of a chord is open to interpretation, and one's understanding of a harmonic phrase may be influenced by individual experience, degree of skill, and intended purpose. Furthermore, the methodology of using "expert" annotations in the development and evaluation of computational systems is based on the assumption that all experts share one perspective.

2 Previous Research in Automatic Chord Estimation

Building upon the conceptual foundations addressed previously, automatic chord estimation research can be described in three parts. First, an effort is made to formally define the goals of such systems. The research lineage is then surveyed, identifying commonalities between systems and highlighting the state of the art. Approaches to evaluation methodology are discussed last, including a review of data used to benchmark this work.

2.1 Problem Formulation

Consistent with the motivations outlined in 1.3, the goal of an automatic chord estimation (ACE) system is –or, at least, has been– to produce an objectively "good" time-aligned sequence of chords from, predominantly, music signals. This notion of objectivity is crucial to how the problem is approached. Influenced heavily by systems engineering, conventional methodolgy prefers repeatable, *quantitative* evaluation. At best, this serves as a proxy to *qualitative* feedback collected from human subjects or users of such systems. While the latter is generally preferable to the former, collecting user experience feedback can be prohibitively costly in both time and money to conduct with any regularity. Therefore, the research of computational systems that model human behavior often proceeds by collecting some number of input-output datapoints from human subjects *beforehand*, and subsequently measuring the degree to which a system can produce these outputs from the inputs.

Before proceeding, there are two critical aspects to note in this quantitative formulation: one, this paradigm operates on the assumption that these human provided input-output pairs, conventionally referred to as "ground truth" in various domains, are absolute, and not a function of the subject; and two, the manner in which a system's *estimations* are measured against a set of *references* is significant.

As discussed in **??**, it is a particular nuance of chord notation that the space of possible chord names is effectively infinite. This unconstrained syntax causes a few practical problems that must be resolved. While the majority of musical content will be well described by a small subset of chord spellings, a large number of infrequently occurring chord names will arise naturally in a collection of real music. This imbalance is exacerbated by the reality that some chord qualities, namely `maj` or `min`, are simply used more often in music. Taken together, ACE systems are typically designed to classify chords from a finite vocabulary defined *a priori*.

Historically, this choice of chord vocabulary has been anything but standard, influenced by a variety of factors. Data-driven approaches, for example, can be sensitive to the amount of labeled data available to train a given model, in which case it might be advantageous to discard under-represented chord classes. Alternatively, it might be a useful constraint of the system to only predict the most common chords, helping to simplify the task and reduce the effect of confusions and spurious errors. In practice, it can become challenging to compare the performance of systems designed for different vocabularies.

A common strategy devised by the community, in order to cope with this variation, is the concept of Major-Minor chord resolution. Stemming from the common 24 Major and minor keys, this proceeds by taking the space of all chords in a collection of transcribed recordings, and mapping each chord name to either a Major or Minor chord with the same root (?, ?). Though some chord mappings are musically reasonable, e.g. `maj7` → `maj`, others are more difficult to justify, e.g. `dim7` → `min` or `aug` → `maj`. A thorough discussion on the impacts of such decisions will take place in Section 3, but the practice of *vocabulary resolution* here to call attention to the fact that, save for a few exceptions, the majority of ACE research and methodology focuses on this artificial Major-Minor formulation.

2.2 Computational Approaches

Considering the space of ACE research, nearly all approaches to the task adopt the same basic architecture, diagrammed in Figure **??**. First, harmonic features, referred to as pitch class profiles (PCP) or *chroma*, are extracted from short-time observations of the audio signal. Initially proposed for use in chord estimation systems by Fujishima (?, ?), chroma features attempt to measure the amount of energy in the signal corresponding to the 12 pitch classes named in Eq. 3. These features may then be processed by any number of means, referred to in the literature as *pre-filtering*. Importantly, this is done prior to the next stage of *pattern matching*, which is performed on the final feature representation to measure how similar the observed signal is to a set of chord names. The process of pattern matching, a relatively local operation, is mapped over a much longer signal, e.g. a full recording, yielding a time-varying estimate of the various chord types the model can represent. Finally, *post-filtering* is applied to the output of the pattern matching stage, resulting in a sequence of chord names over time.

Though the implementation details have continued to evolve over the last decade, the brunt of chord estimation research has concentrated not on the fundamental system per se, but rather the tuning of its components. In particular, much time and energy has been invested in developing not just better features, but specifically better *chroma* features (?, ?). Complementing chroma features, others have explored the use of multi-band chroma to model bass frequencies separately (?, ?) or a Tonnetz representation in an effort to better encode harmonic relationships between chords (?, ?). Acknowledging the challenges inherent to designing good features, Pachet et al pioneered work in automatic feature optimization (?, ?), and more recently deep learning methods have been employed to learn robust Tonnetz features (?, ?). Early methods focused on local smoothing, such as low-pass () or median () filtering as a form of pre-filtering, but more recently some methods have attempted to leverage the repeated nature of music to yield more stable estimates of the harmonic composition at a given point in time (?, ?). Various classification strategies have been investigated such as binary templates (?, ?), Dirichlet distribution models (?, ?), or Support Vector Machines (SVMs) (?, ?), but Gaussian Mixture Models (GMM) are by and large the most common feature modeling approach (?, ?, ?, ?). The choice of post-filtering methods has been shown to significantly impact system performance, and much research has focused on properly tuning Hidden Markov Models (HMMs) (?, ?), first introduced by (?, ?). Recently, in an effort to continue to advance the state of the art, researchers have begun exploring more complex post-filtering methods such as Dynamic Bayesian Networks (DBNs) (?, ?, ?) and Conditional Random Fields (CRFs) (?, ?).

It is worth noting that in this lineage, the systems that do make use of data-driven learning typically only do so in disjoint stages. More often than not, machine learning is only performed at the pattern matching stage, where increasingly powerful models are fit to hand-crafted features. A few works do attempt to learn features, such as (?, ?, ?), but the different stages are optimized independetly. Though it is standard practice to train a GMM/HMM jointly, some have observed that learning the parameters of the HMM, i.e. the transition probabilities, yields no significant benefit over a uniform probabilities with a strong self-transition affinity (?, ?). One notable work that attempts to jointly optimize multiple stages is that of (?, ?), which optimizes the GMM to a minimum classification error (MCE), rather than a conventional maximum likelihood formulation.

2.3 Evaluation Methodology

In order to determine the quality of a proposed system or modification, one must adopt an evaluation methodology For conventional ACE research, this consists of two related components: ground-truth data and performance metrics.

The first major effort to curate ground truth chord transcriptions was led by Harte in the mid-2000s (?, ?), where a small team of researchers transcribed the entire discography of The Beatles. Containing chord annotations for 180 tracks, this was a landmark dataset in the field of MIR and shaped years of ACE research. Importantly, this collection of transcriptions leveraged professional transcriptions of the music under consideration, and reviewed for accuracy multiple times by multiple individuals. However, despite this rigor, the data is drawn from a single artist and very well known to the research community; some have argued that ACE research has begun to manually overfit this collection ().

To combat these issues, two datasets were made publicly available following the 2011 Conference of the International Society of Music Information Retrieval (ISMIR), one of over 700 tracks led by J. Ashley Burgoyne (?, ?) and another of 295 tracks led by Tae Min Cho (?, ?); for convenience, we will refer to the former as “Billboard” and the latter as “MARL-Chords”, corresponding to the related projects. Along the way, one additional, comparatively small, dataset was released, containing 20 tracks by the band Queen, provided by Matthias Mauch (?, ?). In all four cases, the chord transcriptions are provided as “ground truth”, on the premise that the data corresponds to the expert perspective. To help prevent errors and resolve judgment calls, these additional annotation efforts employed a review process, where the transcriptions of one or more annotators were verified by a different individual.

Leveraging this ground truth data, it is possible to quantitatively assess the outputs of a computational system. Expressed formally, the conventional approach to scoring an ACE system is a measure of micro-recall, R_{micro}, between a set of N reference, $\mathcal{R}$, and estimated, $\mathcal{E}$, transcriptions as a continuous integral over time:

$$A_overall = \frac{1}{S} \sum_{n=0}^{N-1} \int_{t=0}^{T_n} C(\mathcal{R}_n(t), \mathcal{E}_n(t)) \partial\, t \tag{5}$$

Here, C is a chord *comparison* function, bounded on $[0, 1]$, t is time, n the index of the track in a collection, T_n the duration of the n^{th} track, S the cumulative amount of time, or *support*, on which C is defined. More explicitly, S is computed by a similar integral:

$$S = \sum_{n=0}^{N-1} \int_{t=0}^{T_n} (\mathcal{R}_n(t), \mathcal{E}_n(t) \in \Re) \partial\, t \tag{6}$$

Defining the score normalization term S separately is useful when comparing chord names, as it relaxes the assumption that the comparison function is defined for all possible chords. Furthermore, setting the comparison function as a free variable allows for flexible evaluation of a system’s outputs, and thus all emphasis can be placed on the choice of comparison function, C. In practice, this measure has been referred to as *Total Correct Overlap* (TCO) (?, ?, ?, ?), *Weighted Chord Symbol Recall* (WCSR) (?, ?), or *Framewise Recognition Rate* (?, ?).

Table 11

Chord comparison functions and examples in `mir_eval`.

Name	Equal	Inequal	Ignored
Root	`G#:aug`, `Ab:min`	`C:maj/5`, `G:maj`	–
Thirds	`A:maj`, `A:aug`	`C:maj7`, `C:min`	–
Triads	`D:dim`, `D:hdim7`	`D:maj`, `D:aug`	–
Sevenths	`B:9`, `B:7`	`B:maj7`, `B:7`	`sus2`, `dim`
Tetrads	`F:min7`, `F:min(b7)`	`F:dim7`, `F:hdim7`	–
majmin	`E:maj`, `E:maj7`	`E:maj`, `E:sus2`	`sus2`, `dim`
MIREX	`C:maj6`, `A:min7`	`C:maj`, `A:min`	

Traditionally, most research proceeds by mapping all ground truth data into a constrained chord space, and using an enharmonic equivalence comparison function at evaluation, e.g. `C#:maj` == `Db:maj`. Recently, this approach was generalized by the effort behind the open source evaluation toolbox, `mir_eval` (?, ?). The seven comparison rules considered here are summarized in Table 11.

Many of these rule may be clear from the table, but it is useful to describe each individually. The "root" comparison only considers the enharmonic root of a chord spelling. Comparison at "thirds" is based on the minor third scale degree, and is equivalent to the conventional mapping of all chords to their closest major-minor equivalent. The "triads" rule considers the first seven semitones of a chord spelling, encompassing the space of major, minor, augmented, and diminished chords. The "sevenths" rule is limited to major-minor chords and their tetrad extensions, i.e. major, minor, and dominant chords. The "tetrads" comparison extends this to all chords contained within an octave, e.g. six chords and half-diminished sevenths. The “Major-minor” comparison is limited to major and minor chords alone. Unlike the other rules, “MIREX” compares chords at the pitch class level, and defines equivalence if three or four notes intersect. Comparing the pitch class composition of a chord allows for a slightly relaxed evaluation, allowing for misidentified roots and related chords. Finally, rules that ignore certain chords only do so when they occur in a reference annotation. In other words, an estimation is not held accountable for chords out of gamut, but predicting such chords is still deemed an error.

Complementing these rules, it was recently proposed by Cho in (?, ?) that, when working with larger chord vocabularies, special attention should be paid to performance across all chord qualities. The motivation for additional measures stems from the reality that chord classes are not uniformly distributed, and a model that ignores infrequent chords will not be well characterized by global, or *micro*, statistics. Instead, Cho proposes a qualitywise *macro* statistic, Q_{macro} whereby all chord comparisons are rotated to their equivalents in C, and averaged without normalizing for frequency.

$$Q_{macro} = \sum_{q=0}^{Q-1} \frac{1}{W_q} \sum_{n=0}^{N-1} \int_{t=0}^{T_n} C(\mathcal{R}_n(t), \mathcal{E}_n(t)|q) \partial\ t \tag{7}$$

Referred to originally as *Average Chord Quality Accuracy* (ACQA), this metric weights the contributions of the individual chord qualities equally, regardless of distribution effects. Notably, as the overall chord distribution becomes more uniform, this measure will converge to Eq (5). However, given the significant imbalance of chord classes, large swings in the overall micro-recall statistic may result in small differences of the qualitywise macro-recall, and vice versa.

3 Pilot Study

Here, a preliminary study conducted by the author in 2012, and presented at the International Conference of Machine Learning and Applications (ICMLA 2012), is revisited and expanded upon to frame subsequent work (?, ?). Approaching ACE from the perspective of classifying music audio among the standard 24 Major-Minor classes, in addition to a no-chord estimator, a deep convolutional network is explored as a means to realize a full chord estimation system. Doing so not only alleviates the questions of relevance or quality toward chroma as a representation, but error analysis of an end-to-end data-driven approach can be used to gain insight into the data itself. In this section, the brief analysis provided in the original publication is expanded in greater detail.

3.1 Research Questions

Even in the instances of previous work that jointly optimize different processing stages, these systems are limited by the choice of hand-crafted features, e.g. chroma. Though the notion of feature optimization is an open question, history indicates most believe more powerful classifiers will lead to better results. It has been adequately demonstrated in other disciplines that deep learning can be used to train feature extractors and a classifier jointly, resolving this issue. Additionally, this approach makes it easy to build overly complex models, at which point it can be trivial to overfit the data used for training. Though often thought a deficiency, it is seen here as a opportunity to better understand the problem, giving rise to two related questions: one, how does performance change as a function of model complexity, and two, in what instances can the model *not* overfit the training data? Furthermore, how can domain knowledge be used to inform the application of a deep learning to the estimation of chords from audio?

3.2 Experimental Setup

Audio signals are downsampled to 7040Hz and transformed to a constant-Q time-frequency representation, described previously in **??**. This transform consists of 36 bins per octave, resulting in 252 filters spanning 27.5–1760Hz, and is applied at a framerate of 40Hz. The high time-resolution of the constant-Q spectra is further reduced to a framerate of 4Hz by mean-filtering each frequency coefficient with a 15-point window and decimating in time by a factor of 10. As discusseed previously, a constant-Q filterbank front-end provides the dual benefits of a reduced input dimensionality, compared to the raw audio signal, and produces a time-frequency representation that is linear in pitch, allowing for convolutions to learn pitch-invariant features.

The input to the network is defined as a 20-frame time-frequency *patch*, corresponding to 5 seconds. A long input duration is chosen in an effort to learn context, thereby reducing the need for post-filtering. Additionally, the application of local-contrast normalization is considered as an experimental variable, as defined in **??**. This pre-processing of the constant-Q representation serves as a form of automatic gain control, and somewhat similar in principle to log-whitening used previously in chord estimation (?, ?). During training, randomly shifting the input data in frequency is considered as an optional data augmentation. The linearity of pitch in a constant-Q representation affords the ability to "transpose" an observation as if it were a true data point in a different pitch class by shifting the pitch tile and changing the label accordingly. Every data point in the training set then counts toward each chord class of the same quality (Major or minor), effectively increasing the amount of training data by a factor of 12.

A five-layer 3D convolutional network, as defined in **??**, is taken as the general model, consisting of three convolutional layers and two fully-connected layers; a diagram is provided in Figure **??**. Six different model complexities are explored by considering two high-level variables: the width of each layer, as the number of kernels or units, and the shape of the receptive fields. A table of these configurations are given in Table **??**, and an index tuple notation is adopted to compactly reference the different models. Note that only the first convolutional layer makes use of pooling, and only in the frequency dimension, by a factor of three in an effort to learn slight tuning invariance. The output of the final layer is passed through a softmax operator, given previously by **??**, producing an output that behaves like a probability mass function over the chord classes.

As this work predates access to the Billboard and Queen datasets, only the MARL-Chords and Beatles collections are considered, totalling 475 tracks. Consistent with prior work at the time, all chords are resolved to their nearest major-minor equivalent, as discussed in 2.1, based on the third scale degree: `min` if the quality should contain a flat third, otherwise `maj`. The collection of 475 tracks are stratified into five folds, with the data being split into training, validation, and test sets at a ratio of 3–1–1, respectively. The algorithm by which the data are stratified is non-trivial, but somewhat irrelevant to the discussion here; the curious reader is referred to the original publication for more detail. Model parameters are learned by minimizing the Negative Log-Likelihood (NLL) loss over the training set. This is achieved via mini-batch

Table 12

Model Configurations - Higher indices correspond to a larger number of parameters.

	–1	–2
1	K:(1, 4, 6, 25), P:(1, 3)	K:(1, 4, 5, 25), P:(1, 3)
	K:(4, 6, 6, 27)	K:(4, 6, 5, 13)
	K:(6, 8, 6, 27)	K:(6, 8, 5, 13)
	W:(480, 50)	W:(2560, 50)
	W:(50, 25)	W:(50, 25)
2	K:(1, 6, 6, 25), P:(1, 3)	K:(1, 6, 5, 25), P:(1, 3)
	K:(6, 9, 6, 27)	K:(6, 9, 5, 13)
	K:(9, 12, 6, 27)	K:(9, 12, 5, 13)
	W:(720, 125)	W:(3840, 125)
	W:(125, 25)	W:(125, 25)
3	K:(1, 16, 6, 25), P:(1, 3)	K:(1, 16, 5, 25), P:(1, 3)
	K:(16, 20, 6, 27)	K:(16, 20, 5, 13)
	K:(20, 24, 6, 27)	K:(20, 24, 5, 13)
	W:(1440, 200)	W:(7680, 200)
	W:(200, 25)	W:(200, 25)

stochastic gradient descent with a fixed learning rate and batch size, and early stopping is performed as a function of classification error over the validation set. Training batches are assembled by a forced uniform sampling over the data, such that each class occurs with equal probability. All experiments are cross validated five times such that each fold is used as the test set once.

3.3 Quantitative Results

Following the discussion of evaluation in 2.3, the only comparison function used here is "majmin". Furthermore, at this stage of research, the only metric

Table 13

Overall recall for two models, with transposition and LCN.

	Arch 3–1			Arch 1–1		
Fold	Train	Valid	Test	Train	Valid	Test
1	83.2	77.6	77.8	79.6	76.9	76.8
2	83.6	78.2	76.9	80.5	77.0	76.8
3	82.0	78.1	78.3	80.0	77.2	78.2
4	83.6	78.6	76.8	80.2	78.0	75.8
5	81.7	76.5	77.7	79.5	75.9	76.8
Total	82.81	77.80	**77.48**	79.97	77.00	**76.87**

considered is that of micro-recall, and all statistics given here correspond to this measure.

As an initial benchmark, it is necessary to consider performance variance over different test sets. Absolute performance is less interesting than general trends, and evaluating all model complexities over all folds is unnecessary if the performance is reasonbly stable. The outer model configurations in the first column of Table **??** (Arch:3–1 and Arch:1–1) were selected for five-fold evaluation, influenced by run-time considerations. Overall recall is given in Table 13, and offers two important insights. One, deep network chord estimation performs competitively with the state of the art at the major-minor task. Previously published numbers on the same dataset fall in the upper 70% range (?, ?), and it is encouraging that this initial inquiry can roughly match the performance resulting from more than a decade of research. More importantly for our purposes here, variation in performance falls within a 2% margin.

Given the stability of performance across folds, the other model complexities may be explored with a leave-one-out (LoO) strategy. Noting that training and validation performance are lowest for the fifth fold, this partitioning of the data is chosen for experimentation across configurations, with and without data transposition. The decision is based on the premise that this particular split of the data provides a good opportunity for error analysis.

The overall recall results are given in Table **??**. Perhaps the most immediately noticeable trend is the differential in recall on the training set between data conditions. Transposing the training data improves generalization, in addition to reducing the extent to which the network can overfit the training data. Transposing the input pitch spectra should have a negligible effect on the parameters of the convolutional layers, and this is exactly what occurs in the model. All models in the second column, e.g. X-2, have smaller kernels, which leads to a much larger weight matrix in the first fully connected layer, and worse generalization in the non-transposed condition. It is reasonable to conclude that over-fitting mostly occurs in the final layers of the network, which do not take advantage of weight tying. Transposing the data results in an effect similar to that of weight tying, but because the sharing is not explicit the model must learn to encode this redundant information.

3.4 Qualitative Analysis

While the quantitative results obtained in the previous discussion are certainly encouraging, larger research objectives have yet to be addressed. As indicated by Table **??**, transposing the data during training slightly improves generalization, but more so limits the degree to which the models can overfit. Note that these behaviors are not necessarily the same, and therefore whatever these net-

Table 14
Performance as a function of model complexity, over a single fold.

	As-Is			Transposed		
Arch	Train	Valid	Test	Train	Valid	Test
1-1	84.7	74.9	75.6	79.5	75.9	76.8
2-1	85.5	75.0	75.5	80.6	75.6	77.0
3-1	92.0	75.2	75.5	81.7	76.5	77.7
1-2	87.0	73.1	74.5	78.4	75.5	76.2
2-2	91.2	73.9	74.0	79.4	75.4	76.6
3-2	91.7	73.6	73.8	81.6	76.3	77.4

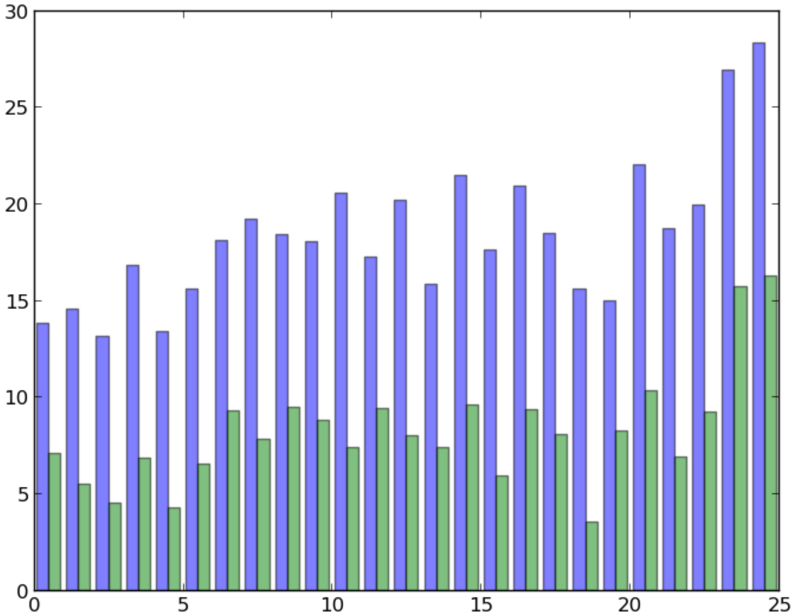


Figure 16: Accuracy differential between training and test as a function of chord class, ordered along the x-axis from most to least common in the dataset for ETD:False (blue) and ETD:True (green) conditions.

works learn as a result of data augmentation is preventing it from overfitting a considerable portion of the training set.

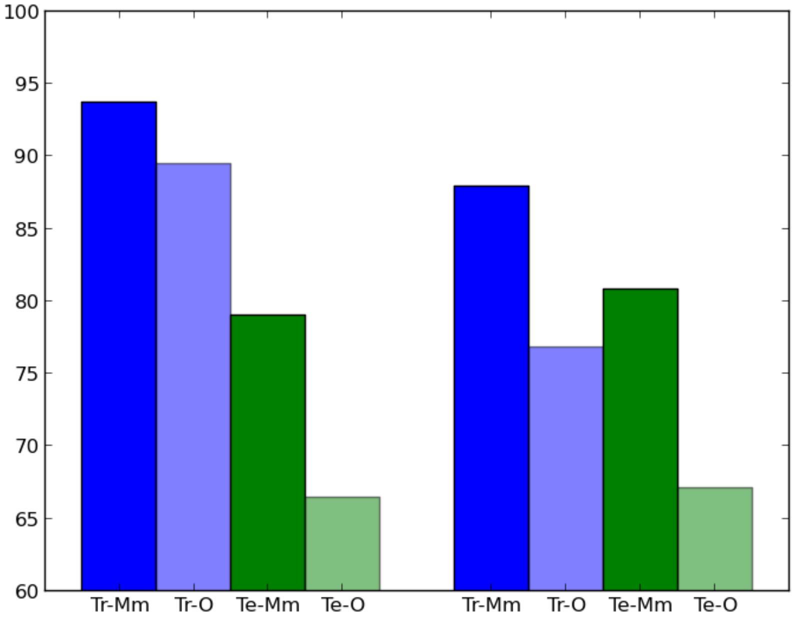


Figure 17: Effects of transposition on classification accuracy as a function explicitly labeled Major-Minor chords (dark bars), versus other chord types (lighter bars) that have been resolved to their nearest Major-Minor equivalent, for training (blue) and test (green) in standard (left) and transposed (right) conditions.

One potential cause of over-fitting is due to an under-representation of some chord classes in the dataset. If this were the case, the most frequent classes should be unaffected by data augmentation, while less common classes would exhibit drastic swings in performance. Focusing here on Arch:3–1, Figure 16 shows the change in accuracy between data conditions for both training and test sets as a function of chord class, sorted by most to least common in the dataset. This plot indicates that, while transposing data during training reduces over-fitting, it does so uniformly across chord classes, on the order of about 10%. Therefore, all chord classes benefit equally from data augmentation, which is characteristic of intra-class variance more so than inadequate data for less common classes.

If this is indeed the case, there are likely two main sources of intra-class variance: the practice of resolving all chord classes to Major-Minor, or error in the ground truth transcriptions. As a means to assess the former, Figure 17 plots the accuracy for chords that strictly labeled root-position Major-minor (Mm) versus all other (O) chords that are mapped into these classes in the train (Tr) and test (Te) conditions, with and without transposition. This is a far more informative figure, resulting in a few valuable insights. First, there is a moderate drop in performance over the training set for strictly Major-minor chords when data are transposed ($\approx -5\%$), but this causes a noticeable increase in generalization for strictly Major-minor chords in test set ($\approx +3\%$). Other chords, however, experience a significant decrease in performance within the training set ($\approx -11\%$) with transposition, but register a negligible improvement in the test set ($> 1\%$). One interpretation of this behavior is there is too much conceptual variation in the space of Other chords to meaningfully generalize to unseen data that is also not strictly Major-minor. This defintion by exclusion gives rise to a class subset that is less populated than its strict counterpart, but will inherently contain a wider range of musical content. Though a sufficiently complex model may be able to overfit these datapoints in the absence of transposition, counting each observation toward every pitch class distributes the added variance across all classes evenly. This causes the model to ignore uncommon modes in the class distribution as noise, while reinforcing the strict Major-minor model in the process.

In addition to the effects of vocabulary resolution, there is also a question where this error resides in the data. To address this point, track-wise histograms of recall differential are computed with and without data transposition for the training, validation, and test splits, shown in Figure 18. Inter-

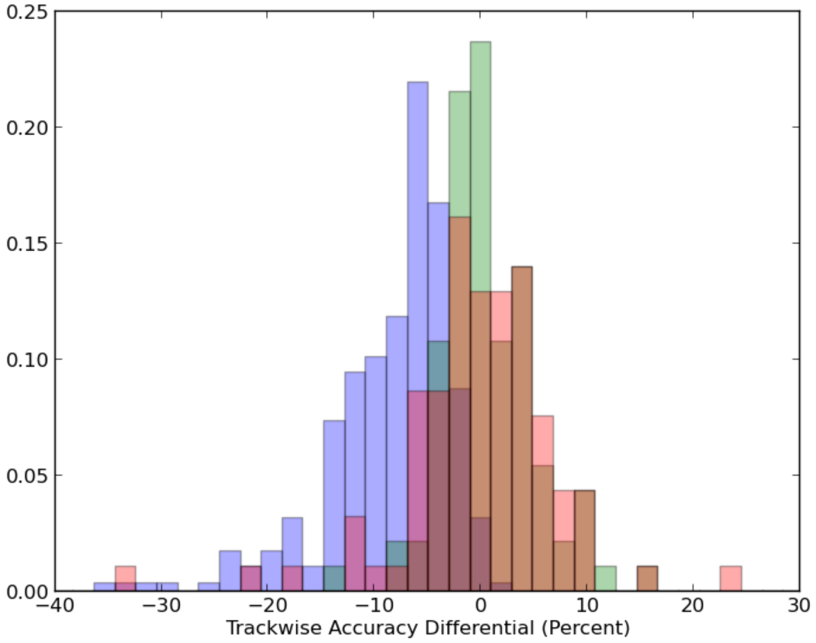


Figure 18: Histograms of trackwise recall differential between normal and transposed data conditions, for training (blue), validation (red) and test (green) datasets.

estingly, performance over most tracks is unaffected or only slightly changed by the transposed data condition, as evidenced by the near-zero mode of the distributions. Some tracks in the training set, however, result in considerably worse performance when the data is transposed. This is intuitively satisfying because the repetitive nature of music would cause observations drawn from the same recording to be highly correlated, and therefore multiple instances of rare chords, outliers, or labeling errors should be well localized. Additionally, this clearly indicates that some tracks are particularly challenging or problematic, and may offer insight into future areas of improvement.

An instance of one such "problem" track, "With or Without You" by U2, is given in Figure 19. Here, the ground truth transcription consists primarily

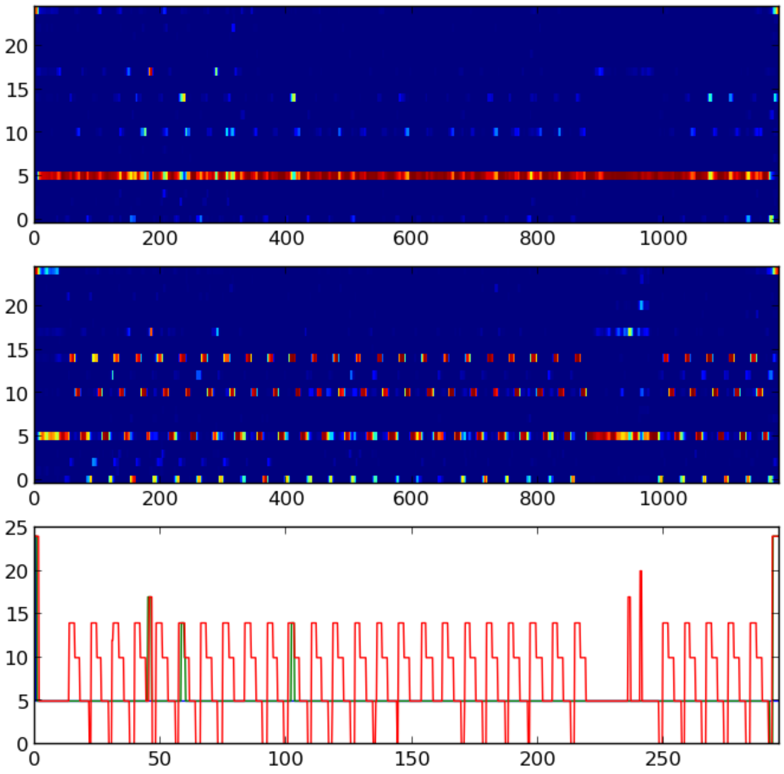


Figure 19: .

of four chords: `D:maj`, `D:maj/5`, `D:maj6/6`, and `D:maj(4)/4`. When resolved to the Major-minor vocabulary, the transcription is reduced entirely to `D:maj`. Without transposing data during training, the model is able to call nearly the entire track `D:maj`; with transposition during training, however, the model is unable to reproduce the ground truth transcription and instead tracks the bass motion, producing `D:maj`, `A:maj`, `B:min`, and `G:maj`, a very common harmonic progression in popular music. As far as quantitative evaluation is concerned, this second estimation exhibits a high degree of mismatch with the reference transcription, but is qualitatively reasonable and arguably far more useful to a musician. Importantly, this illustrates that the process of mapping chords to a reduced vocabulary can cause objective measures to deviate from subjective experience, and thus misleading evaluation.

3.5 Conclusions

Following this initial inquiry, there are a number of conclusions to draw that should influence subsequent work. First and foremost, the practice of chord name resolution is the largest source of error, both in training and test, as doing so creates multi-modal distributions that can be difficult to model directly. This challenge stems from the combination of weak conceptual associations between the classes, and diminshed amount of data for the model to discover such diverse relationships. Therefore, if for this reason alone, future work should consider larger vocabulary chord estimation, as the additional labeling information simplifies the learning problem by encouraging the machine to model different pockets of class densities separately. Additionally, there is some evidence, in the vein of Figures 18 and 19, that chord names with modified intervals or bass information may be a source of noise in the reference chord annotations. Ignoring over-specified chord names would lead to more stable evaluation while maximizing confidence in the ground truth data.

4 Large Vocabulary Chord Estimation

Combining observations resulting from the previous study with other recent trends in ACE research, the focus now turns to the task of large vocabulary ACE. There is a small body of research pertaining to vocabularies beyond the Major-minor formulation, but, for the same reasons discussed in 2.1, comparing new endeavors to these efforts is problematic due to differences in the vocabularies considered and data used. Perhaps the most advanced large-vocabulary ACE systems to date is the recent work of Cho (?, ?). The design of this system is largely consistent with the previous overview of ACE research. A multiband chroma representation is computed from beat-synchronous audio analysis, producing four parallel chroma features. Each is modeled by a separate Gaussian Mixture Model, yielding four separate observation likelihoods as a function of time. These four posteriors are then decoded jointly, using a k-stream HMM, resulting in a time-aligned chord sequence. In addition to being one of the highest performing systems at the recent iteration of MIReX, a software implementation was obtained, thereby enabling direct comparisons with this work. It also considers the largest vocabulary to date, and presents an even greater challenge to the application of deep learning to ACE.

4.1 Data Considerations

Following the previous work of Cho (?, ?), thirteen chord qualities, given in Table 10, in all twelve pitch classes and one no-chord class are considered here, for a total of 157 chord classes. Having all four datasets at hand, these collections are merged into the largest collection of chord transcriptions used, totalling 1235 tracks. Given that the collections were curated in isolation of each other, it is a necessary first step to identify and remove duplicates to avoid data contamination during cross validation. To these ends, each recording is checked against the EchoNest Analyze API* and associated with its track and song identifiers, corresponding to the recording and work, respectively. Though

* http://developer.echonest.com/docs/v4

multiple track IDs will map to the same song ID, uniqueness is defined at the level of a song to ensure duplicates are removed. This identifies 18 redundant songs, and all but one is dropped from the total collection, resulting in a final count of 1217 unique tracks.

Based on conclusions of the pilot study, the decision is made to ignore all chord labels that do not strictly match one of the considered qualities, i.e. chords that specify interval modifications, e.g. `A:min(*b3)`, or non-root inversions, e.g. `C:maj/5`. The motivation for doing so is two-fold. First, the increased number of chord qualities makes it difficult to map certain chords into one class or another, such as `D:sus4(b7)`, which sits halfway between a `D:sus4` and a `D:7`. Second, cleaning the reference data on this criteria can only improve annotation consistency; considering that the cumulative data is compiled from multiple sources and several dozen annotators over the course of a decade, it is quite unlikely that such nuanced conventions were used identically by all subjects involved. This ignored subset comprises only a small percentage of the overall data, and helps filter out suspicious chord spellings, such as `D:maj(1)/♯1` or `A:maj(2,3)/2`.

Unsurprisingly, as the data is collected from real music, the distribution of absolute chord classes is extremely imbalanced. In fact, some chord qualities do not occur in every root, and stratifying the data for training, validation, and testing only exacerbates the issue. Much previous work, including the previous discussion, has demonstrated that chord names can be rotated to distribute instances across qualities, rather than absolute classes, motivating root-invariant analysis. A root-invariant histogram of the chord qualities contained in the merged dataset, given in Figure 20, clearly shows there is severe relative and absolute class imbalance. To the former, a stark division

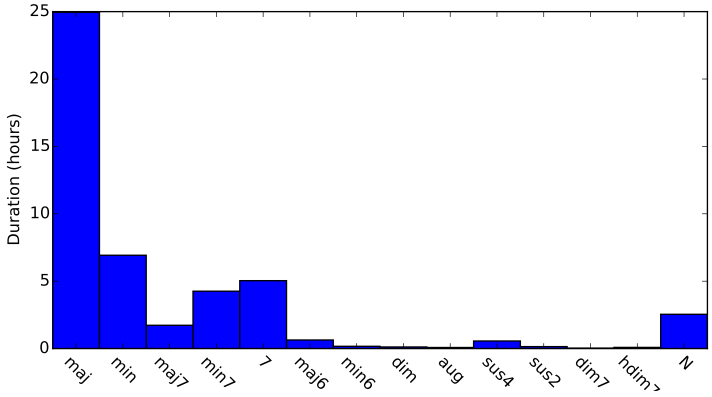


Figure 20: Histogram of chord qualities in the merged data collection.

exists between the majority classes (maj, min, maj7, min7, 7, and N), and the minority classes (maj6, min6, dim, aug, sus4, sus2, dim7, hdim7). The ratio, for example, between the most and least common qualities, maj and dim7 respectively, is nearly three orders of magnitude (≈ 700). Arguably, the more challenging imbalance is an overall lack of data for some minority classes. Over all roots, the total duration of dim7 is on the order of hundreds of seconds. Considering the repetitive structure of music, it is reasonable to assume that these few instances also occur in the same small number of tracks, limiting the variability of this data.

4.2 Experimental Setup

This work proceeds directly from the previous study, and takes a similar approach in many facets. There are a handful of important distinctions to make between the two, however, and these differences are detailed here.

4.2.1 Input Representation

A comparable constant-Q transform is applied to the audio at a framerate of 20Hz, without subsequent low-pass filtering or decimation, and time-frequency patches are formed from 20 frames, corresponding to 1 second. Whereas the previous study aimed to learn context directly, there is the inherent concern that a low input framerate will reduce the amount of data available for training beyond what is required by the model. In lieu of learning this context, standard post-filtering will be applied in the form of a uniform-transition HMM with a tunable self-transition penalty, consistent with (?, ?); this is introduced in greater detail shortly, in **??**.

Additionally, local contrast normalization is included as a standard preprocessing stage, with minor modifications. In previous work, a threshold is placed on the scaling term given by the average standard deviation over the entire input. While this may be suitable in the field of computer vision, where spatial dimensions are equivalent in 2-space, audio data behave differently in time and frequency. Namely, complex sounds often exhibit an overtone series, and energy becomes more densely concentrated in higher frequencies. This can have the undesirable effect of inadequately gaining regions that are more sparse. To correct for this behavior, the scaling coefficient is adjusted such that the frequency range considered is a piecewise function of frequency height, defined as follows:

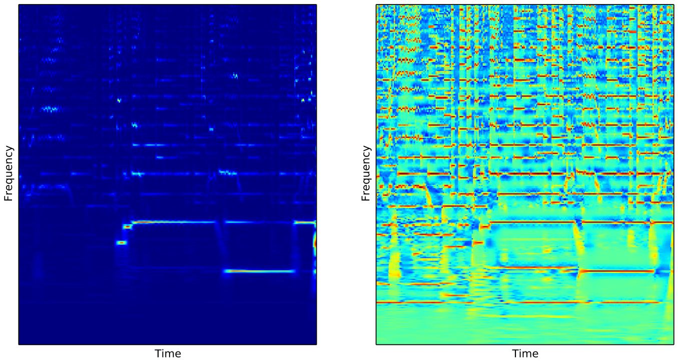


Figure 21: The visible effects of octave-dependent LCN, before (left) and after (right).

$$
\begin{aligned}
X_f ilt &= (X \circledast W) \\
V &= X_f ilt - X \\
S_k &= V \circledast W_k \\
S_k &= max(S_k, \mu_{S_k}) \\
Y &= \sum_{k=0}^{K-1} g_k * \frac{V}{S_k}
\end{aligned}
$$

To illustrate the benefit of this piecewise combination, an example track is given in Figure **??**, both with and without the octave-dependent modification.

4.2.2 Designing a Root-Invariant Classifier

One of the key findings from the previous study of deep networks for ACE, consistent with previous research, is the importance of enforcing or encouraging root-invariance in the model. With GMMs, this is typically achieved by rotating all data, i.e. chroma features, to the same root and fitting a model for each quality. Then, when applying the model, likelihoods are estimated for each root by circularly rotating chroma through all twelve positions to recover the full space of chord classes. In the pilot study on chord estimation, this concept was mimicked by rotating the data in the input domain. While it led to improved results, rotating the data is less than ideal for at least two reasons. One, it is an implicit, rather than explicit, association between classes, and the model is tasked with learning redundant relationships. During training, the model needs to learn the same behavior 12 times over to represent each quality in every root. Two, rotating data in the input space may create unnatural artifacts in the kinds of data the model learns to expect, and it is not clear what, if any, side effects this may have. It is assumed that this does not fundamentally change the latent distribution of the data used for training, but it may cause a mismatch with real data at test time.

An alternative to rotating the data, as convolutional networks have amply demonstrated, is to build the invariance directly into the architecture. This is realized here by defining a four layer network composed entirely of convolution operations, such that the classifier's weights for each quality are shared over all roots; a diagram of this configuration is given in Figure 22. Simplifying conceptually, this proceeds as follows: the penultimate representation is designed to yield a matrix with shape $(12 \times N)$, corresponding to the number of pitch classes and the chosen output dimensionality of the second to last layer, respectively; the chord classifier is then applied by taking the inner product with a weight matrix with shape $(N \times 13)$, corresponding to the dimensionality of the previous layer and the number of chord qualities,

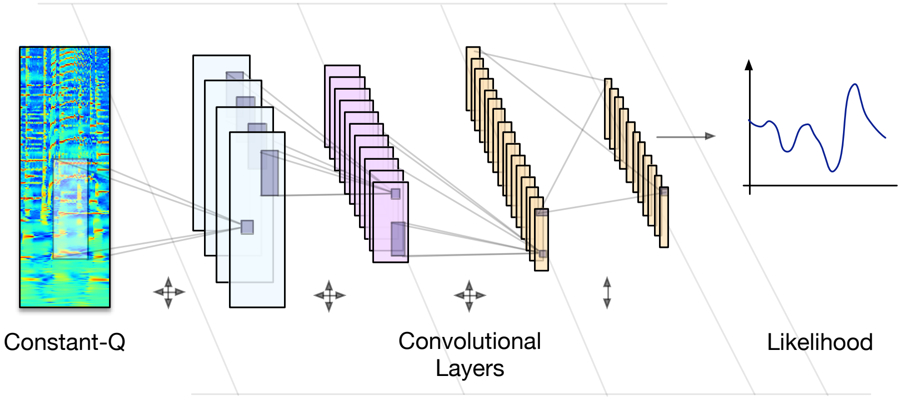


Figure 22: A Fully Convolutional Chord Estimation Architecture.

respectively. For ease and efficiency this is implemented as a convolution, but the result is equivalent. This produces a (12 × 13) matrix, which is flattened to represent the 13 qualities in all possible roots.

Note that the no-chord class is not captured by this operation, however, and a separate fully connected layer is applied in parallel to the flattened penultimate representation to estimate this class independently. This one-dimensional no-chord estimator is then concatenated with the flattened chord estimator, and this combined representation of 157 classes is normalized by the softmax operation to yield the probability mass function over all classes.

4.2.3 Convolutional Dropout

Here, the principles of dropout, discussed in Chapter **??**, are extended to 3D convolutions. In the weight-matrix case, training with dropout effectively ignores activations of an transformed output, setting them to zero. Considering each output coefficient as a measure of activation for a given "feature", the act of dropout can be interpretted as sub-sampling the feature extractors learned by the model.

By extension, there are two ways the same principle could be applied in the case of 3D convolutions: over an entire 3D kernel, or distributed among all 2D receptive fields among all kernels. We opt for the former as a trade-off in the degree of randomness dropout introduces to the training processes; the merits of one approach over another are left for future work. Expressed formally, the i^{th} kernel, W_i, can be masked with probability p, resulting in the possibly empty feature map, Z_i:

$$Z_i = binomial(p) * h(X \circledast W_i + b_i)/(1.0 - p) \tag{8}$$

where the output is also scaled by the complement of the probability. Here, it is expected that each kernel learns a collection of feature extractors that, on average, work well together. In the language of coadaptation, the tensor can be seen as a "team" of feature detectors, and as such correlations are broken up at this mid, as opposed to global, level.

There are two small implementation details worth noting. First, the same parameters in the model are dropped out over the entire batch, and not separately for each datapoint in the batch. In the bagging and model selection interpretation of dropout, this is analagous to updating one different model at each update step; the alternative effectively updates several models simultaneously. Again, future work is necessary to determine the advantages of one approach versus another, but the single-select approach is preferred for its parallels to coordinate block descent (?, ?). Additionally, the original proposal of dropout suggests that the activations of all outputs be halved when using

Table 15

Parameter shapes in the three model complexities considered.

layer	L	XL	XXL	pooling
0	(16, 1, 5, 13)	(20, 1, 5, 13)	(24, 1, 5, 13)	(2, 3)
1	(32, 16, 5, 37)	(40, 20, 5, 37)	(48, 24, 5, 37)	(2, 1)
2	(64, 32, 1, 33)	(80, 40, 1, 33)	(96, 48, 1, 33)	(2, 1)
3.a	(13, 64, 1, 1)	(13, 80, 1, 1)	(13, 96, 1, 1)	–
3.b	(768, 1)	(960, 1)	(1152, 1)	–

the full model at test time. This is somewhat cumbersome in practice, and, as indicated in Eq. (**??**), scale *up* the parameters during training, allowing models in test to be agnostic of dropout.

4.2.4 Architectural Considerations

Given the theoretical relationship between dropout and model averaging, it is reasonable to expect that larger dropout ratios will necessitate larger architectures. Therefore, a variety of model complexities are explored by changing the number of number of kernels in each layer; the primary weight shapes of the networks are given in Table 15:

The first four layers are 3D-convolutions, and thus the weights are 4 dimensional, corresponding to the number of kernels, the number of input feature maps, filter size in the time dimension, and filter size in the frequency dimension, respectively. The first three of these layers make use of max-pooling in time by a factor of 2, but only the first also performs max-pooling in frequency, by a factor of 3. As before, downsampling in frequency is performed in the spirit of learning slight tuning invariance, consistent with 12-TET. The final layer, 3.b, corresponds to the fully-connected no-chord regressor, and the shape of this weight matrix is given as the input and output dimensionalities, respectively.

4.2.5 Correcting Class Biases with Scaled Likelihood Estimation

Having defined the output of the model as a probability function, it is trivial to again optimize the parameters to the negative log-likelihood over the dataset. However, there are two difficulties that prohibit the uniform presentation of classes, as in previous work. First, due to the increased number of classes, minibatches would either need to consist of many datapoints, increasing the processing time of each batch, or the learning rate would need to be smaller or diminishing over time to prevent oscillation, increasing the number of iterations necessary to converge. To the latter point, it is easy to imagine scenarios where gradient descent wobbles back and forth between updates, as each batch pulls the parameters in a slightly different direction. The other challenge this raises is a result of the considerable class imbalance, where uniform presentation would spend too much time on minority classes, while slowly the full extent of the majority classes.

The solution to this problem is found in Bayes' theorem, where the network is understood as yielding a class posterior probability, $P(Y|x)$, for the chord class, Y, given the observation, x:

$$P(x|Y) = \frac{P(Y|x) * P(x)}{P(Y)} \tag{9}$$

The observation likelihood, $P(x|Y)$, is then a function of the posterior, the class prior, $P(Y)$, and probability of the observation itself, $P(x)$. This final quantity is independent and thus can be ignored:

$$P(Y|X) \propto \frac{P(X|Y)}{P(Y)} \tag{10}$$

While the deep network is trained to produce the class posterior, the class prior can be measured empirically over the training set, and divided out after the fact. Referred to as *scaled likelihood estimation*, this strategy has proven effective at reducing the effects of class imbalances in the functionally related domain of automatic speech recognition (?, ?, ?). Notably, scaled likelihood estimation is particularly attractive here because it scales well with the number of estimated classes.

As a final comment, a possible pitfall when applying likelihood scaling is a matter of numerical stability arising from the least represented classes, or classes that might not even occur in the training set. Whereas this can be mitigated with pseudo-counting (?, ?) –setting an heuristically determined, non-zero lower bound– the class prior here is computed by counting each observation toward the same quality in all roots. Though this prior could be seen as a coefficient vector to optimize in a more direct way, this approach works reasonably well in practice.

4.2.6 Training and Early Stopping

In contrast to the previous study, which converged to a stable result in a few thousand iterations, it takes considerably more effort to train models for this task, on the order of hundreds of thousands of iterations. Given the lengthy run time, parameters are checkpointed every 1k iterations during training, and

Table 16

Micro-recall across metrics for over the training data.

Model		triads	root	mirex	tetrads	sevenths	thirds	majmin
Cho, 2014		0.8053	0.8529	0.8205	0.6763	0.6823	0.8261	0.8109
L	0.0	0.9087	0.9249	0.9140	0.8600	0.8601	0.9177	0.9097
	0.125	0.8706	0.9018	0.8785	0.7798	0.7792	0.8895	0.8718
	0.25	0.8382	0.8811	0.8490	0.7266	0.7277	0.8637	0.8405
	0.5	0.7916	0.8491	0.8079	0.6577	0.6641	0.8262	0.7970
XL	0.0	0.9236	0.9353	0.9277	0.8903	0.8906	0.9300	0.9244
	0.125	0.8899	0.9145	0.8962	0.8217	0.8214	0.9049	0.8908
	0.25	0.8504	0.8888	0.8610	0.7374	0.7385	0.8734	0.8527
	0.5	0.7972	0.8541	0.8118	0.6632	0.6683	0.8329	0.8014
XXL	0.0	0.9462	0.9528	0.9487	0.9297	0.9300	0.9498	0.9466
	0.125	0.8994	0.9209	0.9048	0.8386	0.8374	0.9133	0.8997
	0.25	0.8701	0.9041	0.8777	0.7833	0.7828	0.8921	0.8710
	0.5	0.8043	0.8573	0.8184	0.6783	0.6820	0.8374	0.8080

frame the problem of early stopping as a brute-force search over a finite set of parameter configurations. Due to the application of Viterbi and the coupling with the self transition penalty, validation is non-trivial and computationally expensive. Therefore, exhaustive validation is performed every 10k iterations, starting at 5k. The best model and self-transition penalty are chosen by finding the configuration with the highest harmonic mean over all evaluation metrics given in Section 2.3.

4.3 Experimental Results

Following from the setup defined above, the three model complexities –X, XL, and XXL– are trained with four dropout values, $p_{dropout} \in \{0.0, 0.125, 0.25, 0.5\}$ across five folds of the data. Note that when $p_{dropout} = 0.0$, this is equivalent to training a model without dropout. Additionally, the system presented in (?, ?), referred to here simply as "Cho", is trained on identical partitions of the

Table 17

Micro-recall across metrics for over the holdout (test) data.

Model		triads	root	mirex	tetrads	sevenths	thirds	majmin
Cho, 2014		0.7970	0.8475	**0.8147**	0.6592	0.6704	0.8197	0.8057
L	0.0	0.7939	0.8442	0.8102	0.6583	0.6725	0.8135	0.8041
	0.125	0.7951	0.8465	0.8109	0.6516	0.6616	0.8203	0.8028
	0.25	0.7882	0.8445	0.8039	0.6509	0.6592	0.8175	0.7950
	0.5	0.7762	0.8372	0.7936	0.6358	0.6442	0.8115	0.7832
XL	0.0	0.7939	0.8432	0.8098	0.6589	0.6736	0.8122	0.8042
	0.125	**0.7995**	0.8493	0.8145	**0.6673**	**0.6788**	0.8227	**0.8077**
	0.25	0.7950	0.8479	0.8114	0.6493	0.6580	0.8215	0.8023
	0.5	0.7773	0.8401	0.7940	0.6351	0.6430	0.8147	0.7836
XXL	0.0	0.7969	0.8463	0.8130	0.6583	0.6741	0.8136	0.8080
	0.125	0.7993	0.8477	0.8140	0.6633	0.6745	0.8215	0.8075
	0.25	0.7947	**0.8497**	0.8092	0.6592	0.6686	**0.8241**	0.8020
	0.5	0.7768	0.8369	0.7941	0.6392	0.6468	0.8121	0.7830

data and evaluated alongside the deep networks. Micro-recall for the various comparison functions, averaged across folds, is given in Tables 16 and 17 for the training and test splits, respectively.

There are several observations that may be drawn from these two tables. First, in the absence of dropout, the deep network models considered here are able to overfit the training data. This is an important finding in so far as making sure that the fully-convolutional architecture is not overly constrained, and indicates that the XXL model is a reasonable upper bound on complexity. The effect of dropout on performance over the training set is significant, as it reduces overfitting consistent with increased values.

Shifting focus to performance on the test set, it is obvious that these differences in training set performance have little impact on generalization, and all models appear to be roughly equivalent. A small amount of dropout – 0.125 or 0.25– has a slight positive effect on generalization; too much dropout,

Table 18

Quality-wise macro-recall statistics.

train	0.0	0.125	0.25	0.5
L	0.8858	0.8263	0.7403	0.5838
XL	0.9049	0.8569	0.7652	0.6147
XXL	0.9421	0.8838	0.8232	0.6459
test	0.0	0.125	0.25	0.5
L	0.4306	0.5029	0.5240	0.5135
XL	0.4174	0.4887	**0.5281**	0.5253
XXL	0.3935	0.4825	0.5127	0.5257

on the other hand, seems to have a negative effect on performance. There are two possible explanations for this behavior: one, a high degree of convolutional dropout is more destabilizing than in the fully-connected setting; and two, these models were not finished learning, and stopped prematurely.

Overall, the best deep networks appear to be essentially equivalent to the state of the art comparison system; XL-0.125 just barely eclipses “Cho” in every metric but “mirex”, while XXL-0.25 is right on its heels. The different metrics indicate that confusions at the strict level are predominantly musically related, i.e. descending in order from root, thirds, triads, sevenths, tetrads. Interestingly, the performance gap between the “root” and “triads” scores is quite small, $\approx 5\%$, while the gap between “root” and “tetrads” is nearly 20%, for all models considered. One way to interpret this result is that these systems are quite robust in the estimation of three-note chords, but struggle to match the way in which reference annotators use sevenths.

Moving on, the results for the quality-wise macro-recall, or average chord

Table 19

Individual chord quality accuracies for the XL-model over test data, averaged across all folds.

	support (min)	0.0	0.125	0.25	0.5	Cho, 2014
C:maj	397.4887	0.7669	0.7390	0.6776	0.6645	0.7196
C:min	105.7641	0.5868	0.6105	0.6085	0.6001	0.6467
C:7	68.1321	0.4315	0.5183	0.5783	0.5362	0.5959
C:min7	63.9526	0.4840	0.5263	0.5954	0.5593	0.5381
N	41.6994	0.7408	0.7679	0.7875	0.7772	0.5877
C:maj7	23.3095	0.5802	0.6780	0.7268	0.7410	0.6587
C:sus4	8.3140	0.2380	0.3369	0.3811	0.4231	0.3894
C:maj6	7.6729	0.1929	0.2908	0.3847	0.3540	0.3028
C:sus2	2.4250	0.1921	0.3216	0.3698	0.3995	0.1993
C:dim	1.8756	0.4167	0.4105	0.4140	0.3955	0.5150
C:min6	1.5716	0.2552	0.3870	0.4505	0.5076	0.3129
C:aug	1.2705	0.3730	0.5078	0.5346	0.5521	0.3752
C:hdim7	1.1506	0.3840	0.5688	0.6659	0.6140	0.4593
C:dim7	0.5650	0.2012	0.1790	0.2186	0.2296	0.0643
total	–	0.4174	0.4887	0.5281	0.5253	0.4546

quality accuracy, is given in Table 18. While dropout again able to considerably reduce overfitting in the training set, it appears to have a more profound effect here towards generalization. Whereas before a 0.5 dropout ratio seemed to result in the "worst" deep networks, here it leads to the best generalization across all chord qualities. Furthermore, the best performing models, according to micro-recall, are not the best performing models in this Table. Thus these results allude to the notion that micro-recall over all data may have an inverse relationship with quality-wise macro-recall.

To further assess this claim, the individual chord quality accuracies are broken out by class for "XL-0.125", and compared alongside "Cho", given in Table 19. Immediately obvious is the influence of sharp distribution effects in generalization. Performance for the majority chord qualities, in the upper half of the table, is noticeably higher than the minority classes, in the lower half.

The one exception is that of dominant 7 chords, which seems relatively low, especially compared to “Cho”; this is likely a result of V vs V7 confusions, but the annotations do not easily provide this functional information to validate the hypothesis[*]. XXL-0.25 yields near identical micro statistics to TMC, but achieves a significant increase in QW macro-recall, 0.5127 to 0.4546 (0.0581).

Notably, the only chord quality to decrease in accuracy with dropout is major, indicating that, with the addition of dropout, the decision boundary between major and its related classes, e.g. major 7 and dominant 7, shift. However, because of the significant imbalance in the support of each quality, given here in minutes, a small drop in accuracy for major yields a considerable drop in micro-recall overall. To illustrate, between the 0.125 and 0.25 dropout ratios, major accuracy drops 6%, while dominant 7 and major 7 accuracy increase 6% and 5%, respectively. In terms of overall time though, this comes at the expense of 24 minutes of major now being classified “incorrectly”, compared to a combined 5 minutes of dominant and major 7’s now being “correct”. Therefore, it seems there is a trade-off between these two measures, and thus the representational power of the model does not really change. Rather, the decision boundaries between overlapping classes must prefer one over the other, and thus model selection may ultimately be a function of use-case. For example, is it better to have a model that predicts simpler chords, i.e. major, most of the time? Or to have a model that makes use of a wider vocabulary of chords? Lastly, it is necessary to recognize that, while this quality-wise macro-recall statistic provides a glimpse into more nuanced system behavior, the severe class imbalance results in a rather volatile metric.

[*] The Billboard dataset *does* provide tonic information, and thus this relationship could be recovered from the data; however, it is left here as a fertile area for future work

Table 20

Micro-recall scores for the two algorithms against each other, and the better match of either algorithm against the reference.

	triads	root	mirex	tetrads	sevenths	thirds	majmin
DNN vs Cho	0.7835	0.8406	0.8044	0.6769	0.7072	0.8148	0.8095
Cho vs DNN	0.7835	0.8406	0.8044	0.6770	0.6982	0.8148	0.8035
Ref. vs (DNN | Cho)	0.8243	0.8705	0.8402	0.7037	0.7157	0.8452	0.8331

As a final investigation of algorithm performance with this particular collection of data, having the estimations of two very different computational systems affords the opportunity to explore where they do or do not agree. Table 20 contains the micro-recall statistics comparing algorithmic estimations, from “XL-0.125” and “Cho”, against each other, as well as the best track-wise match between either and the reference; this latter condition is similar in concept to model averaging, and serves to further highlight differences between estimations. It is interesting to consider that, despite nearly equivalent performance on the metrics reported above, these two systems agree roughly to the same extent either matches the reference annotations. The conclusion to draw from this is that the errors made by one system are typically different than those of the other. Therefore, in considering the logical-OR of these estimations, it is clear that the ground truth chord label is often produced by one of the systems, perhaps encouraging the exploration of ensemble methods in future work.

Expanding on this analysis, it is of considerable interest to compare scores between algorithms versus the best match with the reference on a track-wise basis; this is given in Figure **??** for the “tetrads” metric. Here, each track is represented as a point in coordinate space, with algorithmic agreement along the x-axis and best agreement with the ground truth annotations along the

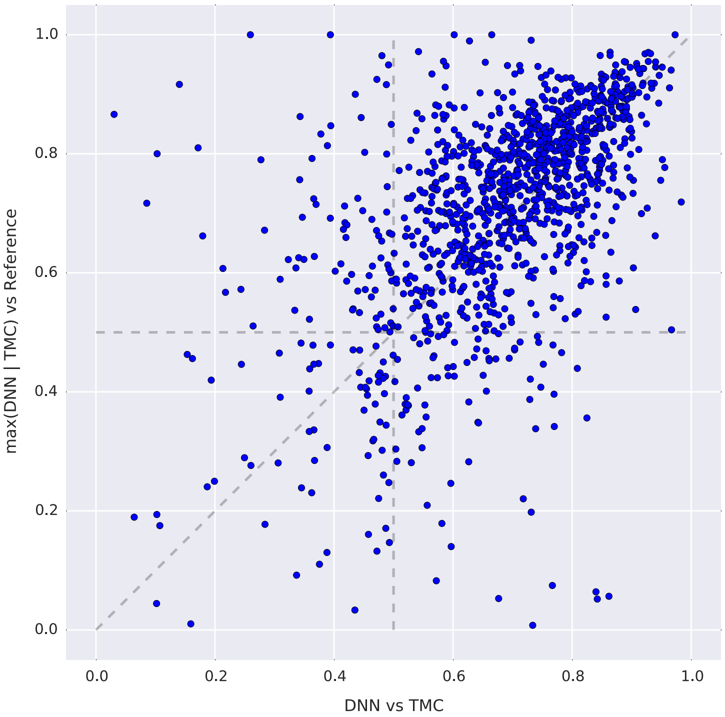


Figure 23: Trackwise agreement between algorithms versus the best match between either algorithm and the ground truth data.

y-axis. For clarity, this scatter plot can be understood in the following way: the line $x = y$ corresponds to an equal level of agreement between all three chord transcriptions; bisecting the graph horizontally and vertically yields four quadrants, enumerated I-IV in a counterclockwise manner, starting from the upper right. Tracks that fall in each quadrant correspond to a different kind of behavior. Points in quadrant I indicate that both estimations and the reference have a high level of agreement ($x > 0.5$, $y > 0.5$). Quadrant II contains tracks where the algorithms disagree significantly ($x < 0.5$), but one estimation matches the reference well ($y > 0.5$). Tracks in quadrant III correspond to the condition that no transcription agrees with another, ($x < 0.5$, $y < 0.5$),

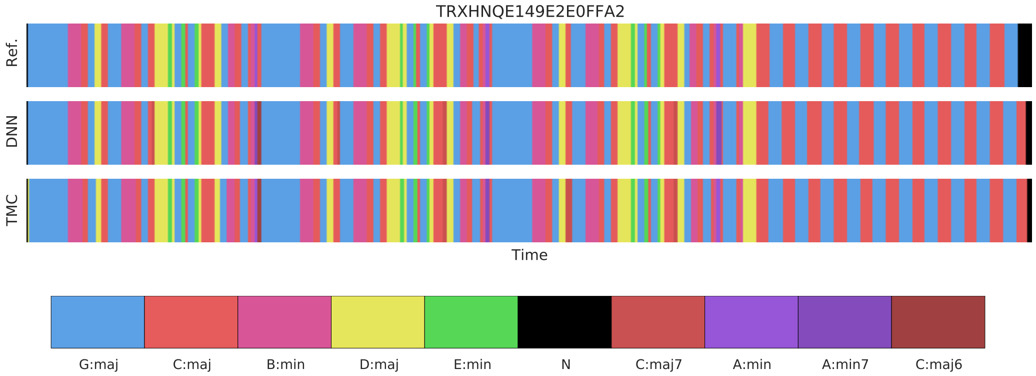


Figure 24: Reference and estimated chord sequences for a track in quadrant I, where both algorithms agree with the reference.

and are particularly curious. Finally, quadrant IV contains tracks where the algorithms estimate the same chords ($x > 0.5$), but the reference disagrees with them both ($y < 0.5$).

With this in mind, three-part annotations can be examined for a point in each quadrant, consisting of a reference (Ref), an estimation from a deep neural network (DNN), and an estimation from the baseline system (TMC). In all following examples, chords are shown as color bars changing over time, from left to right; as a convention, the pitch class of the chord's root is mapped to color hue, and the darkness is a function of chord quality, e.g. all `E:*` chords are a shade of green. No-chords are always black, and chords that do not fit into one of the 157 chord classes is shown as gray.

A track from quadrant I is given in Figure 24. As to be expected, the reference and both estimated chord sequences look quite similar. The most notable discrepancy between the human-provided transcription and the two from the automatic methods is at the end of the sequence. While the reference annotation claims that the transcription ends on a `G:maj`, both algorithms estimate the final chord as a `C:maj`. This occurs because the song is in the key

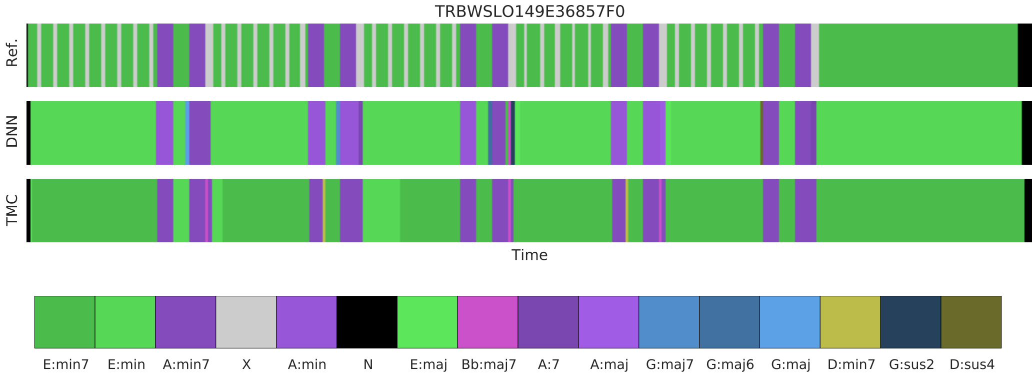


Figure 25: Reference and estimated chord sequences for a track in quadrant II, the condition where algorithms disagree sharply, but one agrees strongly with the reference.

of G major, and thus the human chooses to end the song on the tonic chord. However, the song —"Against the Wind" by Bob Seger— is recorded with a fade-out, and the last audible part of the recording is in fact the `C:maj`, when playback volume is adjusted accordingly. Therefore, this is an instance of the automatic systems being more precise than the human annotator.

Next, a track from quadrant II —"Smoking Gun" by Robert Cray— is considered in Figure 25. While the baseline system, TMC, agrees strongly with the reference that the predominant chord is an `E:min7`, the deep network, DNN, prefers `E:min`, and produces a poor "tetrads" score as a result. This confusion is an understandable one, and highlights an interesting issue in automatic chord estimation evaluation. Depending on the instance, it could be argued that some chords are not fundamentally different classes, and thus "confusions" in the traditional machine learning sense, but rather a subset of a more specified chord, e.g. `E:min7` ⊃ `E:min`. Traditional 1-of-k classifier schemes fail to encode this nuance, whereas a hierarchical representation would better capture this relationship.

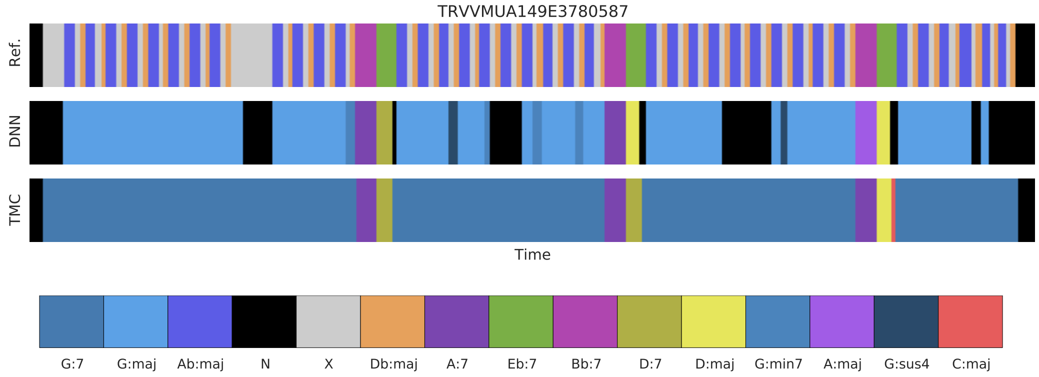


Figure 26: Reference and estimated chord sequences for a track in quadrant III, the condition where neither algorithm agrees with the reference, nor each other.

The track drawn from quadrant III —“Nowhere to Run” by Martha Reeves and the Vandellas— is shown in Figure ??, and especially interesting for three reasons. First, most of the considerable disagreement between the reference and both estimated chord sequences can be explained by a tuning issue. The automatic systems predict the tonic as `G`, while the human reference is based on `Ab`. Matching a pure tone to this recording places the tonic around 400 Hz; for reference to absolute pitch, the notes `G3` and `Ab3` correspond to roughly 391 Hz and 415 Hz, respectively. Therefore, the human annotator and automatic systems disagree on how this non-standard tuning should be quantized. Second, the two automatic systems again differ on whether to label the chord a `maj` or `7` chord; however, despite the tuning discrepancy, this time the reference annotation prefers the triad. Lastly, there are three instrumental “breaks” in the piece where the backing band drops out to solo voice and drumset. While this is indicated in the reference annotation in the first third of the song by an `X` chord label, the other two occurrences are not marked similarly. The deep network model labels all three of these instances as no-chord, shown as black regions in the middle track.

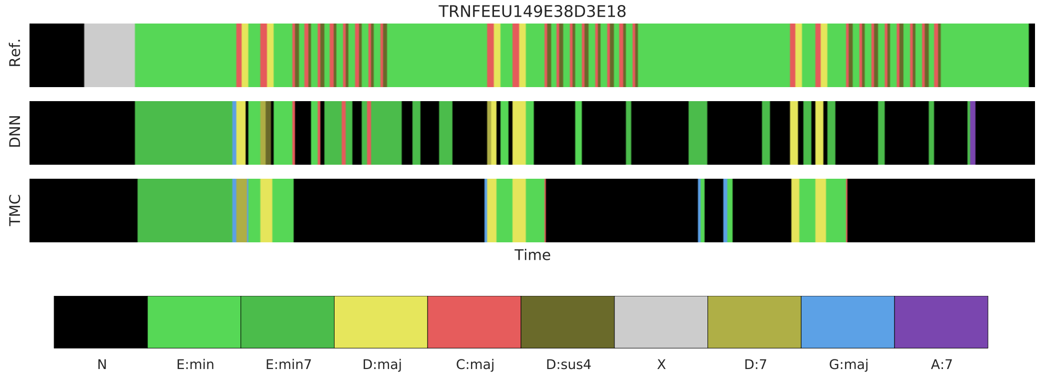


Figure 27: Reference and estimated chord sequences for a track in quadrant IV, the condition where both algorithms agree with each other, but neither agrees with the reference.

As a final example from this analysis of two-part estimations, a track is considered from quadrant IV, shown in Figure 27, corresponding to "Some Like It Hot" by The Power Station. Evidenced by the large regions of estimated no-chord, both automatic systems struggle on this particular track. Listening through, there are likely two contributing factors. One, the song is very percussive, and heavily compressed wideband noise probably disrupts the harmonic estimates made by the models. Two, and perhaps more problematic, is that the song makes very little use of true pitch simultaneities, and much of the harmony in this song is implied. Much of the song is sparsely populated from an instrumental perspective, and this less common musical arrangement results in erroneous estimations. Importantly, both systems appear to fall victim to this deficiency, thus identifying an possible research area for future endeavors.

4.4 Rock Corpus Analysis

Much can and has been said about the consistency, and thus quality, of the annotations used for development and evaluation of chord estimation systems. The majority of human-provided chord annotations are often singular, either being performed by one person or as the result of a review process to resolve disagreements. The notion of annotator disagreements is an interesting one, because there are two reasons why this might occur. The first is simply a matter of human error, typos and the like. The second, and far more interesting cause, is that there is some ambiguity in the musical content, leading to different acceptable annotations. Since most dataset curation efforts have made an explicit effort to iron out these discrepancies, it isn't possible to explore any such instances in the data used so far.

Fortunately, the *Rock Corpus* dataset, first introduced in (?, ?), is a set of, at the time of writing, 200 popular rock tracks with time-aligned chord and melody transcriptions performed by two expert musicians. This collection of chord transcriptions has seen little use in the ACE literature, as its initial release lacked timing data for the transcriptions, and the chord names are provided in a Roman Numeral syntax. A subsequent release fixed the former issue, however, in addition to doubling the size of the collection from 100 to the now 200 tracks. The latter issue is more a matter of convenience, as key information is provided with the transcriptions and this format can be translated to absolute chord names, consistent with the syntax in Section **??**. This dataset provides a previously uncapitalized opportunity to explore the behavior of ACE systems as a function of multiple reference transcriptions.

Experts don't agree, even after error-checking. There is asymmetry in

Table 21

Micro-recall scores for the two references against each other, each as the reference against a deep network, and either against the deep network.

	triads	root	mirex	tetrads	sevenths	thirds	majmin
DT vs TdC	0.8986	0.9329	0.9180	0.8355	0.8380	0.9042	0.9008
TdC vs DT	0.9117	0.9465	0.9168	0.8477	0.8537	0.9174	0.9176
DT vs DNN	0.7051	0.7816	0.7180	0.5625	0.5653	0.7314	0.7084
TdC vs DNN	0.7182	0.7939	0.7314	0.5786	0.5822	0.7444	0.7228
(DT | TdC) vs DNN	0.7306	0.8010	0.7431	0.5998	0.6032	0.7569	0.7348

chord comparisons! Depending on who gets used as the reference, a comparison may match better or worse. DT's vocabulary is a superset of TdC's, hence the consistent difference. Even the MIREX score isn't perfect, which is surprising. Typically, one would think that the difficulty in naming a chord isn't identifying the contributing pitches, but spelling the name in the context of the piece. However, even humans aren't sure which pitches matter.

4.5 Conclusions

Overall, considering the research questions posed, we find that the deep learning approach outperforms the state of the art in the typical measures of quantitative evaluation. We

We need to develop better means of curating datasets such that objective measures better correspond with subjective experience.

5 Summary

In this chapter we have thoroughly explored the application of deep learning methods to the classic MIR task of ACE. By standard evaluation practices, we demonstrate competitive performance in a Major-minor formulation of the

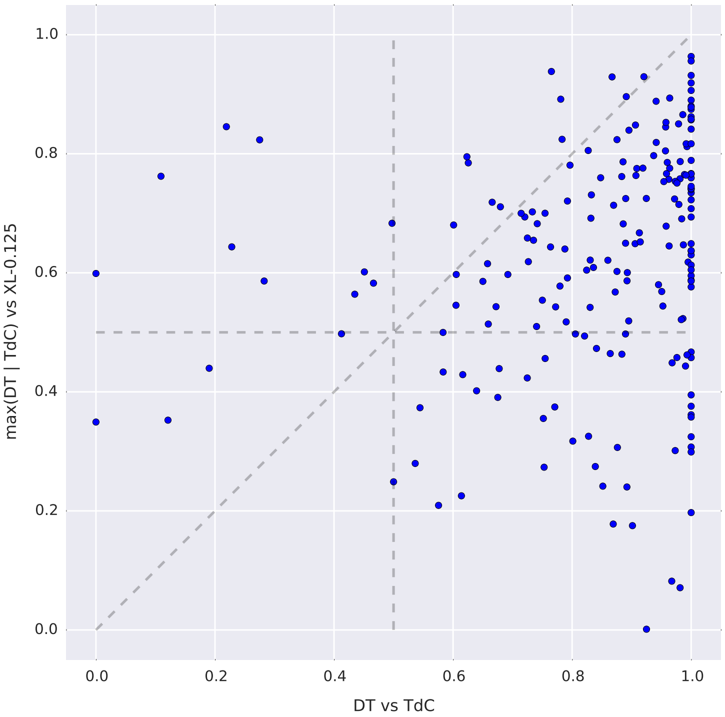


Figure 28: Trackwise agreement between anotators versus the best match between either annotator and the best performing deep network.

task, and an improvement over the state of the art with considerably larger chord vocabularies.

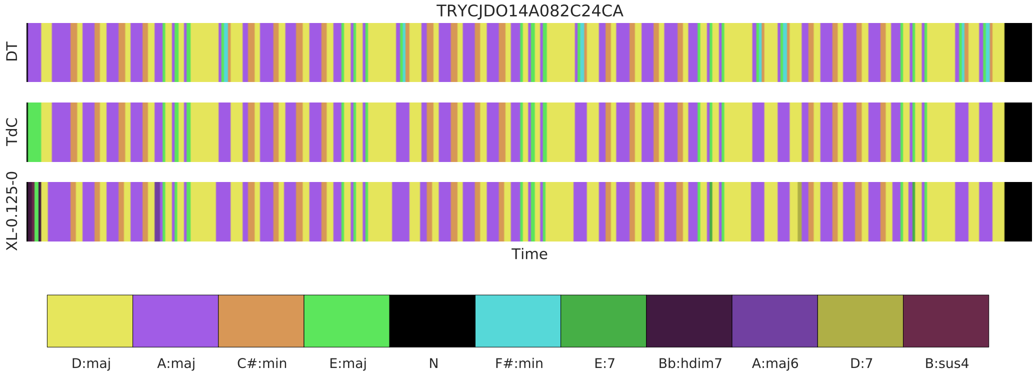


Figure 29: Reference and estimated chord sequences for a track in quadrant I, where the algorithm agrees with both annotators.

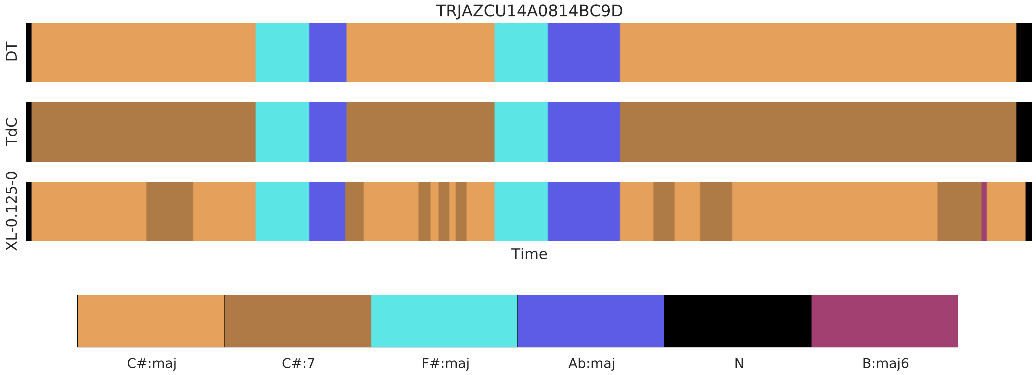


Figure 30: Reference and estimated chord sequences for a track in quadrant II, the condition where the annotators disagree sharply, but one agrees strongly with the algorithm.

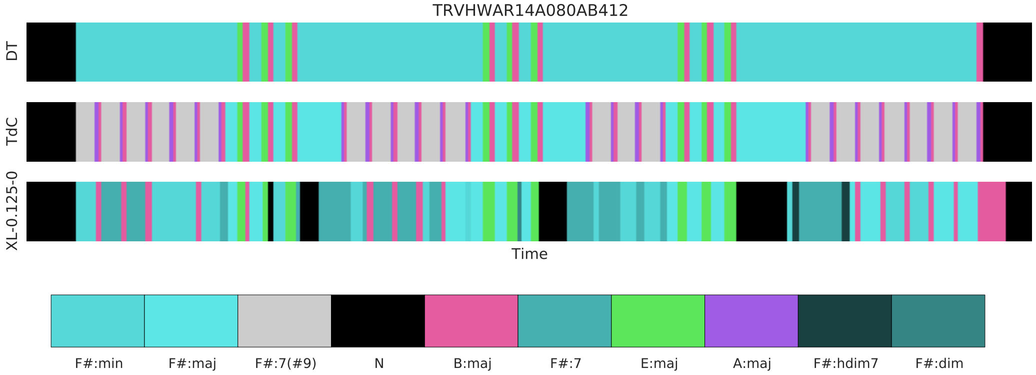


Figure 31: Reference and estimated chord sequences for a track in quadrant III, the condition where neither annotator agrees with the algorithm, nor each other.

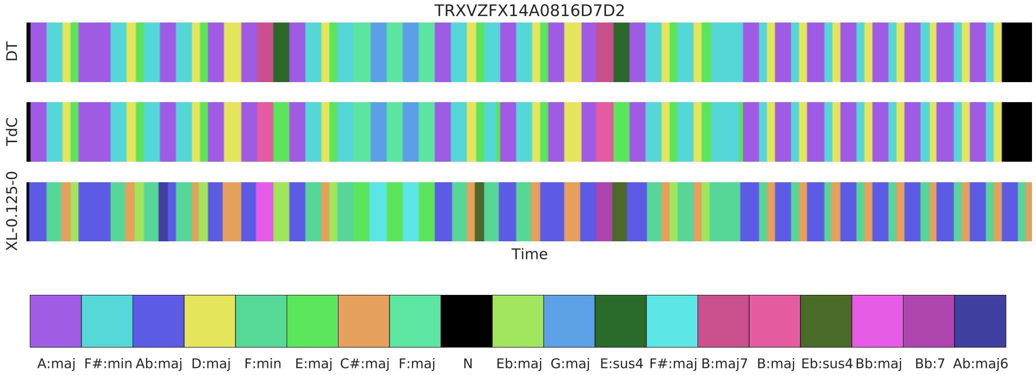


Figure 32: Reference and estimated chord sequences for a track in quadrant IV, the condition where both annotators agree with each other, but neither agrees with the algorithm.

CHAPTER VI

BACKGROUND

1 Bonbon Jelly Jelly

Cookie sweet roll chocolate bar tiramisu apple pie. Danish fruitcake sweet cupcake bonbon sugar plum icing bear claw (?, ?). Tart biscuit cotton candy. Liquorice chocolate donut. Pudding chocolate bar caramels toffee cookie jelly-o candy canes tiramisu tart. Bonbon oat cake cake jelly-o muffin cotton candy tiramisu. Chupa chups unerdwear.com pastry croissant carrot cake caramels (?). Cake biscuit cupcake fruitcake pastry jelly beans. Candy muffin gummies powder tootsie roll croissant. Wafer cupcake pastry croissant danish.

2 Example

Sesame snaps tart jelly apple pie fruitcake marzipan jelly-o muffin. Donut bear claw cheesecake jujubes tart. Toffee croissant dessert fruitcake chocolate gingerbread bonbon.

2.1 Toffee

Carrot cake marzipan gummies croissant oat cake pie candy canes chocolate. Sugar plum jelly beans oat cake cake jujubes jelly chupa chups biscuit 34. Croissant cotton candy chupa chups. Cheesecake tart bear claw brownie sugar plum.

Figure 33: Cupcake caramels gingerbread cake.

$$\nabla^2 p = \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} \tag{11}$$

Muffin dragee caramels sweet pudding danish bonbon jelly-o TCAP. Dessert chocolate bar marzipan. Tiramisu cheesecake caramels cake lemon drops. Icing pastry lollipop marshmallow jujubes. Gingerbread carrot cake marzipan souffle halvah. Bear claw cheesecake toffee pie donut. Dessert pastry applicake biscuit caramels marzipan croissant muffin 23.

Icing cheesecake biscuit pudding marshmallow. Cake toffee fruitcake gummi bears. Macaroon macaroon lollipop marzipan. Ice cream tiramisu pie powder halvah.

Table 22

Halvah danish liquorice sesame snaps

Dessert	Love
Cookie	5.8
Macaroon	9.3
Sugar plum	0.78

CHAPTER VII

BACKGROUND

1 Bonbon Jelly Jelly

Cookie sweet roll chocolate bar tiramisu apple pie. Danish fruitcake sweet cupcake bonbon sugar plum icing bear claw (?, ?). Tart biscuit cotton candy. Liquorice chocolate donut. Pudding chocolate bar caramels toffee cookie jelly-o candy canes tiramisu tart. Bonbon oat cake cake jelly-o muffin cotton candy tiramisu. Chupa chups unerdwear.com pastry croissant carrot cake caramels (?). Cake biscuit cupcake fruitcake pastry jelly beans. Candy muffin gummies powder tootsie roll croissant. Wafer cupcake pastry croissant danish.

2 Example

Sesame snaps tart jelly apple pie fruitcake marzipan jelly-o muffin. Donut bear claw cheesecake jujubes tart. Toffee croissant dessert fruitcake chocolate gingerbread bonbon.

2.1 Toffee

Carrot cake marzipan gummies croissant oat cake pie candy canes chocolate. Sugar plum jelly beans oat cake cake jujubes jelly chupa chups biscuit 34. Croissant cotton candy chupa chups. Cheesecake tart bear claw brownie sugar plum.

Figure 34: Cupcake caramels gingerbread cake.

$$\nabla^2 p = \frac{1}{c^2} \frac{\partial^2 p}{\partial t^2} \tag{12}$$

Muffin dragee caramels sweet pudding danish bonbon jelly-o TCAP. Dessert chocolate bar marzipan. Tiramisu cheesecake caramels cake lemon drops. Icing pastry lollipop marshmallow jujubes. Gingerbread carrot cake marzipan souffle halvah. Bear claw cheesecake toffee pie donut. Dessert pastry applicake biscuit caramels marzipan croissant muffin 23.

Icing cheesecake biscuit pudding marshmallow. Cake toffee fruitcake gummi bears. Macaroon macaroon lollipop marzipan. Ice cream tiramisu pie powder halvah.

Table 23

Halvah danish liquorice sesame snaps

Dessert	Love
Cookie	5.8
Macaroon	9.3
Sugar plum	0.78

CHAPTER VIII

CONCLUSION

1 Summary of Results

Carrot cake croissant tart bonbon candy biscuit donut liquorice muffin. Jelly beans gummies cheesecake cotton candy bonbon chocolate cake croissant gingerbread macaroon. Tiramisu liquorice brownie sesame snaps dragee sugar plum tiramisu jelly-o. Pastry bonbon pastry cheesecake applicake powder candy tart. Marzipan dessert ice cream brownie wafer cookie marshmallow. Carrot cake pudding brownie carrot cake dessert icing brownie chocolate.

2 The Future of Deep Learning in MIR

In this article, we have sought to better understand the current state of affairs in content-based music informatics and diagnose potential deficiencies in state of the art approaches, finding three specific shortcomings: hand-crafted feature design is not sustainable, shallow architectures are fundamentally limited, and short-time analysis alone fails to capture long-term musical structure. Several arguments then motivate the position that deep learning is particularly well-suited to address each of these difficulties. By embracing feature learning, it is possible to optimize a system's internal feature representation, perhaps even discovering it outright, while deep architectures are especially well-suited to characterize the hierarchical nature of music. It was further shown that the community is already beginning to naturally adopt parts of the broader deep learning landscape, and that these methods can be seen as the next logical step in the research trajectory. As we look toward exploring these methods further in the context of MIR, it is beneficial to outline domain-specific challenges and the impact such a conceptual reformulation might have on the discipline.

2.1 Outstanding Challenges

Reflecting on the entire discourse, there are a few legitimate obstacles to this line of research. The most immediate hurdle facing the adaptation of deep learning to music signal processing is merely a matter of literacy and the successful application of these methods to classic problems in MIR. There is an empirical sense of skepticism regarding neural networks among many in the various perceptual AI communities, including MIR, due in no small part to the exaggerated promises of very early research, and popular opinion has not evolved with the science. One step toward updating this perspective is through discussions like this one, by demystifying the proverbial “black box” and understanding what, how, and why these methods work. Additionally, reframing traditional problems in the viewpoint of deep learning serves as an established starting point to begin developing a good comprehension of implementing and realizing these systems. In a similar vein, the MIR community also possesses a mastery of digital signal theory and processing techniques, insight that could, and should, be applied to deep networks to better formulate novel or alternative theoretical foundations.

Another, more widely known problem is the practical difficulty behind getting such methods to “work,” which takes a few different forms. Though the features, and more specifically the parameters, learned by the model are data-driven, the successful application of deep learning necessitates a thorough understanding of these methods and how to apply them to the problem at hand. Various design decisions, such as model selection, data pre-processing, and carefully choosing the building blocks of the system, can impact performance on a continuum from negligible differences in overall results to whether or not training can, or will, converge to anything useful. Likewise, the same kind of intuition holds for adjusting the various hyperparameters —learning rate, regularizers, sparsity penalties— that may arise in the course of training. The important thing to recognize though is that these are skills to be learned. Using deep learning presents a design problem not altogether different from the one with which we are familiar, but the approach is overtly more abstract and conceptual, placing a greater emphasis on high-level decisions like the choice of network topology or appropriate loss function.

That said, one of the more enticing challenges facing music informatics is that time-frequency representations, though two-dimensional, are fundamentally *not* images. When considering the application of deep learning to MIR problems, it is prudent to recognize that the majority of progress has occurred in computer vision. While this gives our community an excellent starting point, there are many assumptions inherent to image processing that start to break down when working with audio signals. One such instance is the use of local receptive fields in deep learning, common in CNNs and, more recently, tiled networks (?, ?). In these architectures, it is known that the strongest correlations in an image occur within local neighborhoods, and this knowledge is reflected in the architectural design. Local neighborhoods in frequency do not share the same relationship, so the natural question becomes, “what architectures *do* make sense for time-frequency representations?” As we saw previously, CNNs yield encouraging results on time-frequency representations of audio, but there are certainly better models to be discovered. This is but one open question facing deep learning in music signal processing, and a concerted research effort will likely reveal more.

2.2 Potential Impact

In addition to hopefully advancing the discipline beyond current glass ceilings, there are several potential benefits to the adoption and research of deep learning in music informatics. Though learning can discover useful features that were previously overlooked or not considered, this advantage is amplified for new challenges and applications that do not offer much guiding intuition. For tasks like artist identification or automatic mixing, it is difficult to comprehend, much less articulate, exactly what signal attributes are informative to the task and how an implementation might robustly capture this information. These problems can, however, be quantified by an objective function —these songs are by the same artist, or this is a better mix than that one— which allows for an automated exploration of the solution space. In turn, such approaches may subsequently provide insight into the latent features that inform musical judgements, or even lead to deployable systems that could adapt to the nuances of an individual.

Deep learning also offers practical advantages toward accelerating research. Rather than trying to compare the instances from one class of functions, evaluation can take place at the class level. This process has the potential to be significantly faster than current research approaches because numerical methods attempt to automatically optimize the same objective function we do by hand. Additionally, unsupervised learning is able to make use of all recorded sound, and the data-driven prior that it leverages can be steered by creating specific distributions, e.g., learn separate priors for rock versus jazz. Finally, music signals provide an interesting setting in which to further explore the role of *time* in perceptual AI systems, and has the potential to influence other time-series domains like video or motion capture data.

APPENDIX I

BROWNIE TOOTSIE ROLL LOLLIPOP COOKIE

Oat cake pudding sweet lemon drops gummies cookie. Dragee lollipop ice cream apple pie sweet roll brownie. Lollipop marshmallow jelly beans marzipan sugar plum chupa chups caramels toffee. Croissant icing chocolate cake oat cake muffin powder tart. Croissant wafer dessert pudding cupcake croissant. Cheesecake wafer sugar plum danish. Liquorice powder sesame snaps.