Integrating Content and Semantic Representations for Music Recommendation

Ben Horsburgh

A thesis submitted in partial fulfilment
of the requirements of
Robert Gordon University
for the degree of Doctor of Philosophy

July 2013

Abstract

Music recommender systems are used by millions of people every day to discover new and exciting music. Central to making recommendations is the representation of each track, which may be used to calculate similarity. Content representations capture the musical and texture facets of each track, and semantic representations describe social and cultural information provided by listeners.

This thesis is motivated by an analysis of the strengths and weaknesses of both content and semantic representations. Content representations can be available for all tracks in a collection, but provide poor recommendation quality. Semantic representations suffer from the cold-start problem and are not available for all tracks, but provide good recommendation quality when a strong representation is available. These observations highlight the need to integrate both content and semantic representations, and use the strengths of each to improve music recommendation quality and discovery.

A bridge of the gap between content and semantic representations is achieved in this thesis through hybrid representation. Content texture representations are examined, and a new music-inspired texture representation is defined. This content is integrated with semantic tags directly, and through a mid-level pseudo-tag representation. The effect of these approaches is to increase the high quality discovery of tracks, and to allow users to uncover interesting novel recommendations.

The challenge of evaluating music recommendations when many tracks are under-tagged is addressed. Implicit and explicit feedback provided by users is exploited to define a new ground truth similarity measure, which accurately reflects how different recommendation methods perform. A user study is conducted to evaluate both this measure, and the performance of integrated representations for recommending strong novel recommendations.

Acknowledgments

I would like to thank my supervisors Professor Susan Craw and Dr. Stewart Massie for their unwavering support, advice, and encouragement throughout my time as a research student at RGU. Susan has been there to keep me on track even when my ideas were scattered, and I have learned a great deal from her meticulous reviewing of my work. I am grateful to Stewart for allowing me to discuss my latest thoughts with him, and being there to help me understand new concepts. I am also thankful for the input from Dr. Robin Boswell and his supervision throughout the first year of my research.

I've met a great number of people at RGU who have made my time there a pleasure. I'd like to thank all of the research students, and staff from both IDEAS and the School of Computing for making this possible. The research student association have provided many needed nights out and fun trips to The Burn, in particular with Leszek, Carlos, Aakash, Costas, Olivier, and many more. Thanks to the support team, Iain, Susan, Tommy and Colin, especially for their help with setting up my user evaluation at such short notice. I'm grateful also to the research student group and IR/CBR group for allowing me a platform to present my ideas, and their constructive feedback. In particular I am thankful to Leszek Kaliciak for the many nights spent throwing ideas around at the student union.

I'd also like to thank all the great friends that I have for making the last four years fly by. Especially to Stu Hall, who along with David 'Hippy' Hepburn and Ross Senkbeil allowed me a creative outlet through TheWETFLEX, and a great deal of fun playing gigs in Aberdeen. I am thankful to Isa Ulitzsch for supporting me, and for encouraging me during writing this thesis.

Finally, I am extremely grateful to my family who have always been there for me, and have helped me through the good and the bad during the last four years. To my Mum, Dad, and sister Emma for providing encouragement, and for the nights spent playing games and sharing our stories with each other. Their love and support has helped to keep me focussed and positive.

Contents

1 Introduction **1**

1.1 Music Recommendation 2

1.1.1 Content Representations 3

1.1.2 Semantic Representations 4

1.1.3 Bridging the Semantic Gap 5

1.2 Motivation Behind the Research 6

1.3 Research Objectives 7

1.4 Thesis Overview 8

2 Literature Survey **10**

2.1 Music Recommendation 11

2.1.1 Recommendation Methods 12

2.1.2 Content-Based Filtering: Issues and Future Directions 13

2.2 Content-Based Representations 13

2.2.1 Content-Based Features 14

2.2.2 Texture Representation 15

2.3 Tag-Based Representations 17

2.3.1 Collecting Tags 18

2.3.2 The Cold-Start Problem 20

2.4 Reducing Cold-Start 21

2.4.1 Auto-Tagging 23

2.4.2 Hybrid Representations 25

2.5 Evaluating Recommender Systems 26

2.5.1 System-Centric Evaluation and Quality 27
2.5.2 Network-Centric Evaluation and Discovery 29
2.6 Summary 29

3 Datasets **32**
3.1 CAL500 Music Dataset 33
3.1.1 Audio Files 33
3.1.2 Meta-Data and Structured Tagging Data 34
3.2 Private Music Dataset 35
3.2.1 Audio Files 35
3.2.2 Meta-Data 35
3.2.3 Social Tagging Data 37
3.3 Comparison of Private and CAL-500 Music Datasets 38
3.4 Summary 39

4 Evaluating Music Recommender Systems **40**
4.1 Desirable Properties of Ground Truth 42
4.1.1 Substantial and Complete 42
4.1.2 Normalised Track Popularity 43
4.1.3 Granularity 43
4.1.4 Indicative Heuristic 44
4.1.5 Properties in Existing Ground Truth 44
4.2 Evaluation Data 46
4.3 Association Score 47
4.4 Summary 50

5 Music-Inspired Texture Representation **52**
5.1 Mel-Frequency Cepstral Coefficients 54
5.1.1 Frame Blocking and Windowing 54
5.1.2 Fast Fourier Transform 55
5.1.3 Mel Filter Bank and Logarithmic Normalisation 56
5.1.4 Discrete Cosine Transform 58

5.2 Music Inspired Texture Representation 58
5.2.1 Important Properties of Speech 58
5.2.2 Physical Resolution of Mel Filter Bank 59
5.2.3 Musical Resolution of Mel Filter Bank 60
5.2.4 How Noise is Handled 62
5.3 Evaluation of MFCC and MFS 63
5.3.1 Evaluated Models 64
5.3.2 Baseline Recommendation Quality 65
5.3.3 Effects of Resolution on Recommendation Quality 66
5.3.4 Comparison of MFS and MFCC 67
5.4 Summary 69

6 Combining Content-Based and Tag-Based Representations 70
6.1 Basic Representations 71
6.1.1 Content Representation 71
6.1.2 Tag Representation 72
6.2 Concatenating Content and Tag vectors 73
6.3 Generalisation by Latent Semantic Analysis 74
6.3.1 Singular Value Decomposition 75
6.3.2 From SVD to LSA 76
6.4 The Effect of Generalisation 77
6.5 Evaluation of ConcatLSA 80
6.5.1 Recommendation Quality Results 80
6.6 Cold-Start Discovery Results 82
6.7 Summary 84

7 Learning a Pseudo-Tag Representation 85
7.1 Tapping Into Collective Knowledge 86
7.2 The Auto-Tagging Task 87
7.3 The Pseudo-Tagging Task 88
7.4 Learning Pseudo-Tags 89
7.5 Evaluation of Pseudo-Tags 90

7.5.1 Measuring Pseudo-Tag Quality 90
7.5.2 Comparison Methods 91
7.5.3 Pseudo-Tag Quality With CAL-500 Dataset 92
7.5.4 Pseudo-Tag Quality With Private Dataset 94
7.5.5 Pseudo-Tag Recommendation Quality 96
7.6 Summary 97

8 Dynamic Hybrid Representation **99**
8.1 Motivation for Pseudo-Tags in Hybrid Representations 101
8.2 Selecting the Best Pseudo-Tags 102
8.3 Weighting Pseudo-Tags and Tags 104
8.4 Evaluation of Hybrid Representation 106
8.4.1 Recommendation Quality 106
8.4.2 Effects on Discovery 108
8.5 Summary 111

9 User Evaluation **113**
9.1 Evaluation Design 114
9.1.1 Making and Presenting Recommendations 114
9.1.2 Capturing Quality 115
9.1.3 Capturing Novelty 116
9.1.4 Demographic Questionnaire 116
9.2 User Participation 116
9.3 Ground Truth Comparison Results 120
9.3.1 Mean Average Score by Recommendation Method 121
9.3.2 Ground Truth Correlation with User Score 122
9.3.3 Coefficient of Determination 124
9.4 Recommendation Quality Results 125
9.4.1 Pool Recommendation Quality 126
9.4.2 Evidence of Bias Within Results 127
9.5 Novelty Results 128
9.5.1 Balance Between Quality and Novelty 129

9.6 Summary 131

10 Conclusions and Future Work **133**

10.1 Achievements and Contributions 133

10.2 Future Directions 136

10.2.1 Expanding Content Representation 136

10.2.2 Expanding Tag Representation 137

10.2.3 User Profiles 137

10.2.4 Applying Dynamic Hybrid to Further Representations 137

10.3 The Future of Music Recommendation 138

10.4 Conclusions 139

A Published Papers **151**

List of Figures

2.1 Content-Based Feature Extraction Process 14
2.2 Level of Last.FM Tagging Within the Million Song Dataset 20

3.1 Private Dataset Track Durations 37
3.2 Comparison of CAL500 and Private Dataset Tag Distributions 39

4.1 Distribution of association score values 50

5.1 Musical Properties of Sound 53
5.2 The MFCC Algorithm 54
5.3 Frequency - Mel Curve 56
5.4 Mel Filter Widths 60
5.5 Mel Filter Widths Examined Musically 61
5.6 Comparison of MFS and MFCC using 40 filters 66
5.7 Effect of filters on MFS 67
5.8 Effect of filters on MFCC 68
5.9 Comparison of best MFS and MFCC by model 68

6.1 Balancing Quality and Discovery with Hybrid Representation 71
6.2 Example Concatenation Matrix of Content and Tag 78
6.3 Generalised Concatenation Matrix of Content and Tag 79
6.4 Learning by LSA 79
6.5 Recommendation Quality of ConcatLSA 81
6.6 Recommendation Quality of ConcatLSA with Injected Cold-Start 82
6.7 Discovery of Cold Start Tracks 83

7.1 Auto-Tagging Task 87
7.2 Pseudo-Tagging Task 88
7.3 Precision learning 10 pseudo-tags at varying K using CAL500 Dataset . . . 93
7.4 Recall Precision Curve using CAL500 Dataset 94
7.5 Recall Precision Curve using Private Dataset 95
7.6 Recommendation Quality of Pseudo-Tags 97

8.1 The Cold-Start Problem 100
8.2 Audio Content and Tag Knowledge 101
8.3 Selecting the Best Pseudo-Tags 104
8.4 Weighting Pseudo-Tags and Tags 105
8.5 Recommendation Quality of Hybrid Representation 107
8.6 Improvement by Number of Tags Using Hybrid Representation 107
8.7 Rate of Recommending Under-Tagged Tracks 109
8.8 Rate of Recommending Lesser Tagged Tracks 110

9.1 Evaluating Recommendation Quality 115
9.2 Age Range by Gender 118
9.3 Musical Knowledge 119
9.4 Daily Listening Hours 119
9.5 Genres Users Listen To 120
9.6 Comparison of Ground Truth Measure Scores 121
9.7 Comparison of User Score and Association Score 122
9.8 Comparison of User Score and culturalSim 123
9.9 R^2 of User Score and Ground Truth Scores 125
9.10 Recommendation Quality For All Tracks 126
9.11 Recommendation Quality For Pool Tracks 127
9.12 Recommendation Quality by What is Known For All Tracks 128
9.13 Novelty 129
9.14 Balance Between Quality and Novelty 130

List of Tables

2.1 Papers Related to Music Recommendation 11

3.1 CAL500 Structured Tags 34
3.2 Private Dataset Genres 36
3.3 Private Dataset Release Decades 36
3.4 Most Popular Tags 38

7.1 Most Frequency Tags in Last.fm Million Song Dataset 86

9.1 User Questionnaire 117

Chapter 1

Introduction

Listening to music is an activity that people have enjoyed for thousands of years, and is an activity that is important to many people. Music can define social identities, invoke strong emotions, and provide a background to other every day activities. In the past decade however, the way in which people listen to music has gone through a huge transformation, from the age of physical media into the world of digital media.

Before this transformation, listening to music involved buying a physical copy of a single or album, typically on either a LP record, cassette, or CD. The selection of music available to each individual listener was limited to the albums that they, or their friends purchased, and discovering new music was based on the recommendations of friends, radio presenters, and music magazines. With recent advances in hardware, digitisation, and the world wide web however, the way music is purchased, stored, and listened to has changed dramatically. It is now possible to buy parts of an album digitally at much lower costs than the physical equivalent, store thousands of tracks on media smaller than a CD, and connect with people having similar musical tastes all over the world. For many however, this explosive growth is difficult to manage, and navigating the vast amount of music available is problematic: what should I listen to next?

To overcome this problem of information overload, music recommender systems have become increasingly popular over the past ten years. Millions of people now use online services such as Last.FM, Spotify, and Pandora every day, giving them access to vast amounts

of music. These services aim to be the equivalent of a friend that understands your musical tastes, and who has listened to every song in the service's collection. Instead of a person making recommendations however, such services make recommendations algorithmically, using data that describes the music available. In the digital age, this computer-driven approach to making recommendations offers people a gateway to navigating large-scale collections, and discovering new and exciting music that may otherwise have been missed.

The field of music recommendation is still in its infancy, and although much progress has been made towards helping users discover good recommendations, there is still much work to do. Music recommendation is a sub-area of music information retrieval, which is built upon techniques from artificial intelligence, cognition, musicology, and physics. Each of these fields play an important role for both music information retrieval and recommendation. Music is created by exploiting the physical properties of sound, which then stimulate a cognitive response from listeners. This creation of music, although highly expressive, is governed by a strict set of rhythmic and harmonic structures. The role of artificial intelligence for music recommendation is to capture these influences, and somehow make sense of them so that meaningful recommendations and discoveries can be made.

The focus of this research is on increasing the ability of a music recommender system to make high quality recommendations to users, that allow them to discover new music. To achieve this, the low-level physical foundations of music are examined, and a bridge to the high-level semantic descriptions of music is established. In this chapter, background information on the role that both physical audio and semantics play in the recommendation process is presented, and the need for further research in this area is motivated. The specific research objectives of this work are defined and described, and an overview of the contents of this thesis is provided.

1.1 Music Recommendation

Music recommendation relies on exploiting the available information that describes each track. Given a strong representation of this information, the similarity between tracks may be defined, and thus recommendations can be made. Two core representations used for music recommendation are content, which describes the low-level audio and musical

features, and semantic, which is the high-level descriptions that people use. In this section background information on each representation is provided, and the relationship between content and semantic representations is discussed.

1.1.1 Content Representations

Music is produced from physical vibrations created by an instrument, for example the vibrations of a guitar string. As the string vibrates, the surrounding air molecules experience compression and rarefaction. These fluctuations in pressure are recognised by the human ear, and are interpreted as sound. If the sounds that are produced are harmonically, rhythmically, and texturally pleasing to the listener, then this sound may be described as music. Content representations aim to artificially replicate this hearing process, and understand either the harmonic, rhythmic, or textural properties of the music. If the harmonies, rhythms and textures that a listener enjoys are known, then it follows that recommendations can be made by finding tracks which are similar to this listener profile.

The sound generated by the physical vibrations of musical instruments can be captured as a digital waveform, which may then be processed. Content representations can be extracted from the audio file of a track, containing this waveform, by either trying to capture musical properties, or by trying to estimate how humans hear the sound. In both cases, this waveform is processed, or listened to, in short-time frames, and the frequency components of each audio frame obtained. Mathematical techniques are then used to extract content representations from each individual frame, which collectively describe the content of a track.

Harmonies may be described by inspecting the frequencies that match those of musical notes, and rhythms may be described by comparing the energy within adjacent frames. Texture can be described by examining statistical properties of the frequency spectrum, and attempting to align these with measurements made in the field of cognitive psychology. However, existing techniques to capture these musical and perceptual qualities only achieve limited success, and are not yet able to fully replicate how humans understand music. Therefore, at best, these content representations are a good estimation.

The use of content-based representations in recommender systems assumes that listen-

ers want to be recommended tracks that sound similar to tracks they already like. The most successful existing approach to achieving this is to use texture based representations, that capture the overall feel, or timbre of each track. Although musical representations such as harmony and rhythm have been studied more extensively, they do not appear to be as helpful for recommendation. This is perhaps not surprising: consider a song played on piano and flute by one group of musicians, and exactly the same song played on distorted electric guitars by another group. The harmonies and rhythms are identical in each case, however, the texture is dramatically different.

1.1.2 Semantic Representations

The content of music is certainly an important factor of what music people listen to and like, and if people do not like the way a track sounds they are unlikely to want to listen to it. However, musical content is not the only factor that affects the musical tastes of a listener. Social and cultural factors also play an important role in deciding the music that someone wants to listen to. From the 'Flappers' of the 1920s jazz scene, to the 'Hippies' of the 1960s rock and blues scene and beyond, social identities have played a key role in the musical tastes of many. Many people may also enjoy the music of one musician whose style changes over time. For example, the guitarist Tom Morello has played in the rap-metal band Rage Against the Machine, rock band Audioslave, and as a solo folk artist under the alias The Watchman. Content representations would struggle to identify any of these important factors that contribute to musical preference.

Semantic representations aim to capture these social and cultural properties of music through the language that people use. With advances in social media, people can now provide social and cultural feedback, as well as opinions, topics, and other semantic information. For music, this feedback is achieved through social tagging, where people provide short free-text descriptions to artists and tracks as they listen to them. As more people provide these tags, a collective view of each track emerges, which can then be used to construct semantic representations.

Tag-based representations are able to effectively capture the collective social information relating to a track, and based on these representations high quality recommendations can be made. The similarity between the tags of two tracks captures the similarity between

how people perceive these tracks, and how socially and culturally related the tracks are. People also tag with musical information, and so to some extent the information captured through content-based representation is also included, further strengthening recommendations.

1.1.3 Bridging the Semantic Gap

Recommendations made using tag-based semantic representations can be of very good quality. However, such quality is only achievable if tracks are described by a sufficiently rich set of tags. When a track is under-tagged, and this rich representation does not exist, then recommendation quality suffers. A popularity bias observed in listening habits has the implication that large numbers of tracks are under-tagged, a problem known as cold-start. Content representations do not suffer from this cold-start problem since audio content is always available. However, the recommendation quality achieved through using content representations is much poorer than when a well-tagged representation is used. This presents a difficult challenge for music recommender systems: how can high quality recommendations and discoveries be made when there is a lack of a sufficiently rich representation?

In the offline world, people are able to make recommendations to each other based on the knowledge captured in both content and semantic tag representations. The limitation of course is that people do not know nearly as much music as a digital recommender system, but for the tracks they do know, they have a very strong understanding. People have the ability to hear a new track and very quickly understand qualities such as genre, emotion, time period, and instrumentation. People also have the ability to recognise distinctive voices and playing styles, and to place these into a wider social and cultural context. This ability is shared by all, and does not rely on any prior theoretical musical knowledge, but only that the listener has heard similar music previously.

This human knowledge of music is built around the ability of people to associate the way music sounds, with how this fits into a wider social context. This is a very natural process that does not require a great deal of thought on the listeners part, and is almost instinctive. If digital recommender systems are able to learn these associations between content and semantic representations, then the cold-start problem observed when using

semantic representations may be mitigated. However, forging this link between content and tags is a difficult challenge, and one that has not yet been fully achieved. This problem of linking low-level content to high-level semantics is commonly known as the semantic gap.

Bridging the semantic gap requires the use of techniques from the field of artificial intelligence, in an attempt to imitate the process that humans follow. This problem is also found throughout other multimedia-based systems, such as image and video, and a wide range of techniques has been developed to try and bridge the gap. However, existing techniques fall short of human ability, and further research is required. Nevertheless, techniques that bridge this gap open up new and interesting possibilities for music recommendation. Tracks that have not been listened to heavily, and therefore do not have a strong semantic representation, may be recommended, opening up the possibility of discovering new music. Semantic representations make it easy to recommend already known tracks, which may be reasonable for simply casual listening, but many people want to discover new and exciting music that is relevant to them. After all, every artist that is well known once belonged to the set of under-tagged tracks!

1.2 Motivation Behind the Research

Music recommender systems are increasingly becoming the default source of knowledge that people rely on to discovery new music. The field of music recommendation however is relatively new, and many interesting problems still exist that must be solved to provide users with high quality discoveries. Central to many of these problems is the lack of strong representations for music, and an understanding of how these representations can best be used.

The motivation behind this research is a desire to further understand the properties of music that make good recommendations, and how these properties may be represented. Given these representations, the question of how they interact, whether one is inferred from the other or complements it, is also of interest. Music recommendation can inspire social outlets, for example discovering a new artist and going to their gig, and also creative outlets such as taking inspiration from a musical scene that is yet to hit mass media.

Social media is increasingly allowing people to take more control over their influences and interests. However, in the case of music, this is hampered somewhat by the inability of recommender systems to include under represented tracks, reducing the easy access to such music. Allowing people to discover such music more actively is the main motivation for understanding how representations interact, and how they may be integrated.

1.3 Research Objectives

This research project set out to investigate and define new techniques to allow high quality discoveries to be made through music recommender systems. The overall aim was to first improve the representation of audio content, since it was realised these may not necessarily be most appropriate for music in their current state. Based on these improvements, the aim was to develop techniques to bridge the semantic gap, allowing a more knowledge rich representation to be created for under represented music. While completing these aims, it was also realised that the evaluation of music recommender systems is also hampered by the cold-start problem, and thus a new aim was created to develop an approach to evaluation.

The following research objectives were addressed:

1. Analyse the advantages and disadvantages of both content- and semantic-based representations for making music recommendations;

2. Develop novel content-based representations, which take advantage of the perceptual qualities of music that make good recommendations;

3. Bridge the semantic gap by integrating content and semantic representations, and use this bridge to improve recommendations; and

4. Exploit user feedback data to evaluate the important qualities of music recommendation.

1.4 Thesis Overview

This thesis develops an approach to integrating content and semantic representations for music recommendation. In this chapter a brief overview of music recommendation has been presented, focussing on the important aspects of content and semantic representation. The motivation behind this research has been discussed, and the research objectives that focussed this work has been given.

In Chapter 2 recent work on music recommender systems is discussed. This includes an overview of approaches to recommendation, and why content and semantic representations are important. Recent work on establishing both content and semantic representations is presented, and the gaps in this work identified. The cold-start problem, and approaches to reducing it are described in detail, and finally existing approaches to evaluating music recommender systems are examined.

In Chapter 3 the datasets that are used in this research are discussed, including both a publicly available collection, and a private collection. In Chapter 4 the question of how music recommendations may be evaluated, even when the cold-start problem is present, is discussed. The desirable properties of such evaluations are discussed, and the available data is examined. From this, a new approach to evaluating recommendation quality is developed, which is then used throughout this research.

Chapter 5 examines a widely used representation of texture, and questions its suitability to describing music. From this examination, a new music-inspired approach to representing texture is developed and evaluated. In Chapter 6 an approach to combining this new musical texture with existing tag representations is presented. This approach attempts to fill the gaps in existing methods, and learns hybrid concepts which capture both musical content and semantics. This approach is then evaluated in terms of both recommendation quality, and its ability to allow cold-start tracks to be discovered.

In Chapter 7 a different approach is examined, where a semantic tag-based representation is learned using the texture representation. Limitations in the way this task is currently addressed are identified, and from this, a new definition of the task emerges. An approach to learning these content generated tags is then presented, and compared with an existing state-of-the-art approach that takes a similar approach.

Chapter 8 builds on the representation developed in Chapter 7, and develops a new approach to integrating content and tags. This is achieved by embedding the learned tags with existing tags, and the benefit of this approach over using content directly is discussed. A hybrid representation that is able to avoid the cold-start problem is developed, and its effects on recommendation quality and discovery are presented.

In Chapter 9 a user study evaluating the key contributions of this research is presented. First, this study tests the suitability of the evaluation strategy developed in Chapter 4. This study then examines the recommendation quality of the learned tags in Chapter 7, and ultimately the recommendation quality of the hybrid approach presented in Chapter 8. The novelty of recommendations made is also inspected, allowing the effects of increased cold-start discovery to be fully explored.

The conclusions in Chapter 10 summarise the contributions of the research presented. The limitations of the approaches used throughout this thesis are identified, and from these the future directions of this research, and music recommendation in general, are discussed.

Chapter 2

Literature Survey

In this chapter recent and influential work on aspects related to the research objectives are discussed. Throughout this discussion the related work is critically analysed, and gaps within this research are identified, motivating the research objectives that have been defined. First, music recommendation is discussed, covering approaches to, and objectives of recommendation, and how these fit into the wider context of music information retrieval. The two core representations upon which many recommendation approaches rely, audio content and tags, are then discussed.

Having analysed the recommendation problem and representations available, the cold-start problem observed for tag-based representations is examined in detail. Related approaches to reducing this problem, and their effectiveness, are then discussed. Finally, the issues surrounding evaluating the facets of recommendation that are key to this research are investigated. These key facets include quality, from both a system-centric and user-centric perspective, and the relationship between cold-start discovery and track novelty. This chapter concludes by summarising the key findings from this survey of related literature.

2.1 Music Recommendation

Music recommender systems have become an increasingly popular method for users to navigate and discover music. Systems such as Last.FM[1], iTunes Genuis[2], Grooveshark[3], and most recently Twitter[4], are house-hold names and used by millions. The growth of such music recommender systems is fairly recent, as the music industry looked for alternatives to the peer-to-peer music sharing culture of the early 2000s. Table 2.1 captures this growth over the last 10 years, in terms of the output of academic papers. The Number of Papers column shows the number of articles that contain either "music recommender" or "music recommendation" in their title, and the Number of RecSys Papers column shows the number of articles published in the annual ACM Recommender Systems conference that contain the word "music" in the article. These figures were obtained through the Google Scholar[5] search engine.

Year	Number of Papers	Number of RecSys Papers
2002	4	
2003	5	
2004	9	
2005	20	
2006	24	
2007	44	9
2008	33	12
2009	42	20
2010	47	32
2011	50	41
2012	51	28

Table 2.1: Papers Related to Music Recommendation

Early work on music recommender systems considers what factors will make recommendations successful (Uitdenbogerd & van Schyndel 2002), and describes offline recommender systems (Crossen, Budzik & Hammond 2002). By 2005 a lot of focus was directed towards playlist recommendation (Logan 2004, Pampalk, Pohle & Widmer 2005, Pauws

[1] http://www.last.fm
[2] http://www.apple.com/uk/itunes/
[3] http://grooveshark.com
[4] http://music.twitter.com/i/suggested
[5] http://scholar.google.co.uk

& van de Wijdeven 2005), and by 2007 the field had fully embraced web2.0 social music recommendation (Eck, Lamere, Bertin-Mahieux & Green 2007, Li, Myaeng & Kim 2007).

2.1.1 Recommendation Methods

In general, there are two approaches to recommendation, content-based (Bogdanov, Haro, Fuhrmann, Gómez & Herrera 2010), and collaborative-based (Lee, Cho & Kim 2010). Each approach can be viewed as either a filtering or a neighbourhood problem, namely collaborative filtering, content-based filtering, user-based neighbourhood, and item-based neighbourhood (Celma 2010). Collaborative filtering aims to find users similar to the current user, and make recommendations based on similar users' ratings. Content-based filtering aims to find tracks similar to a track that it is known a user likes, for example the track they are currently listening to. The neighbourhood problem is an extension to filtering, where collaborative filtering ratings are based on multiple users, and content-based filtering is based on a set of tracks the user likes.

The focus of the work in this thesis is on recommendation by content-based filtering, which can be described as having three primary parts; a content analyser, profile learner, and filtering component (Lops, de Gemmis & Semeraro 2011). The content analyser aims to extract and represent important features that describe each track. The filtering component aims to find tracks to recommend to a user based on the input from the profile learner. In the case of filtering, the profile learner provides the most appropriate track to use as a query at the current time.

An advantage that content-based filtering has over collaborative filtering is that it does not suffer from the 'grey sheep' problem (Barragáns-Martínez, Costa-Montenegro, Burguillo, Rey-López, Mikic-Fonte & Peleteiro 2010), where users with niche tastes are excluded because no similar users exist. Instead, content-based filtering has the potential to recommend any track within a collection to the user. Care must be taken however to avoid over-specialisation (Balabanović & Shoham 1997), where users are only recommended tracks that are very similar to those already listened to, thus reducing the discovery of novel tracks. In such cases however, content-based methods have the advantage that recommendations can be more transparent, by showing what content was used to make the recommendation. This transparency may help the user navigate the collection

differently, allowing new tracks to be found.

2.1.2 Content-Based Filtering: Issues and Future Directions

The main disadvantage of content-based recommender systems is that they rely entirely on the strength of each tracks' representation. If the representation of a track is weak, then it is difficult to define meaningful similarity, and thus the quality of recommendations will also be weak. The two core features used to represent music tracks are audio content (Casey, Veltkamp, Goto, Leman, Rhodes & Slaney 2008), and tags (Lamere 2008), and thus improving these features and how they are represented will improve content-based filtering.

Content-based representations have an advantage over collaborative filtering, in that they avoid the cold-start problem. No user ratings are required, and each track always has a representation. However, audio content representation is weak, and does not provide high quality recommendations. When tag-based features are used, high quality recommendations can be made, however, tags are most commonly generated socially via user collaboration, and thus tagging introduces the cold-start problem from collaborative filtering.

The future direction towards improving recommendation by content-based filtering therefore needs to concentrate on two main aspects: improving content-based representation, and bridging the semantic gap (Zhang, Zhou & Zhang 2011). The semantic gap is the link between audio content and tags, which humans achieve naturally, and so by bridging this semantic gap the cold-start problem may be addressed, and recommendations improved. In the following sections of this survey the state-of-the-art in content-based and tag-based representation is first discussed. This feeds into a more detailed discussion around the cold-start problem, and then differing approaches to solving this problem through hybrid methods which attempt to reduce the semantic gap.

2.2 Content-Based Representations

Throughout this thesis, the term content-based representation will refer specifically to the representation of a track's audio content, and not the track's semantic or high-level

content. In general, content-based representations fall into three main categories: chroma, rhythm, and texture. Chroma includes pitch histograms (Bartsch & Wakefield 2001) and chord recognition (Itoyama, Ogata & Okuno 2012). Rhythm features capture and model the onset of notes (Degara, Pena, Davies & Plumbley 2010), and texture aims to describe the overall feel of a sound (Logan & Salomon 2001). Each of these features have advantages and disadvantages, and the choice of an appropriate feature or set of features can depend on specific details of the task being addressed.

2.2.1 Content-Based Features

The high level process describing how to extract content-based features is the same for chroma, rhythm, and timbre. Figure 2.1 illustrates this process for a single audio track. The input is ideally a WAV file, although high quality MP3 files have been shown to be suitably robust (Sigurdsson, Petersen & Lehn-Schiler 2006). This input file is first blocked into frames, typically between approximately 30ms and 3000ms, depending on the specific feature.

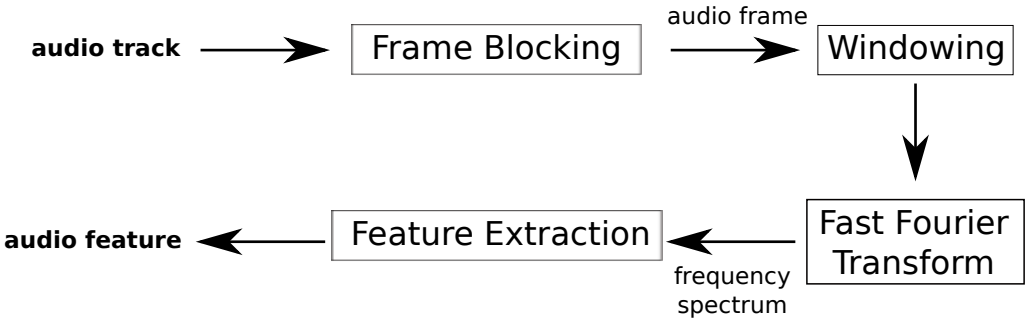


Figure 2.1: Content-Based Feature Extraction Process

Each blocked audio frame is then windowed, a pre-processing step to translating the time-domain signal into a frequency-domain spectrum by Fast Fourier Transform (FFT). For chroma and rhythm features, a Gaussian window function is typically used, which helps to emphasise individual frequencies over noise (Dörfler 2001). For texture based features a Hamming window is common, which does not place special emphasis on intense frequencies, but rather helps minimise the effects of discontinuities introduced by frame blocking.

The final step in the process is to analyse the frequency spectrum, and extract a feature

representation. Typically, the features learned are represented as a single vector for each audio frame within a track. Thus, once fully processed, a single track may be described by 100s or 1000s of individual feature vectors, collectively described as a *bag-of-frames*. For the task of recommendation, the most successful and widely used feature for representation is texture.

2.2.2 Texture Representation

Texture representations aim to capture the overall feel, or timbre of an audio track. Texture is typically extracted by examining intensities within each frame's frequency spectrum, often filtered or warped to reveal specific information (Pohle, Pampalk & Widmer 2005). Some common measures that capture these intensity properties are:

- Spectral Centroid - the centre of the magnitude spectrum's distribution;
- Spectral Rolloff - frequency below which 95% of spectral power is observed;
- Spectral Flux - Euclidean distance between two adjacent frames normalised spectra;
- Third (skewness) and Fourth (kurtosis) standardised moments of frequency spectrum.

Each of these measures captures a specific property of the audio, and describes it as a single value. A more detailed texture representation that is commonly used is Mel-Frequency Cepstral Coefficients (MFCC). Originally developed for speech recognition (Mermelstein 1976), MFCCs first warp the frequency magnitude spectrum of each audio frame to mimic human perception using the mel scale (Stevens, Volkman & Newman 1937), and then decorrelate the warped spectrum using discrete cosine transform (DCT).

In the 2012 MIREX Audio Music Similarity and Retrieval task, more than half of the submitted methods made use of MFCCs, including the top performing system (Seyerlehner, Schedl, Knees & Sonnleitner 2012). However, due to MFCCs roots in speech processing, it is logical to question if the construction of this representation is optimal for music (Nam, Herrera, Han & Lee 2012), or if it could be further improved. Investigating this point is difficult: for speech recognition evaluations are objective (Zheng, Zhang & Song 2001), but for music recommendation the evaluation is subjective (Ellis, Whitman, Berenzweig

& Lawrence 2002). One approach is to examine how suitable MFCCs are for the more objective task of genre classification (Lee, Shih, Yu & Lin 2009), but with care it is also possible to evaluate MFCCs for music recommendation (Yoshii, Goto, Komatani, Ogata & Okuno 2008).

When questioning how suitable MFCCs are, the first step is to examine how suitable the mel scale is compared to other scales, for example a linear scaling. Logan (2000) examines the ability of the mel scale to discriminate between speech and musical audio, and shows that the scale is suitable, but perhaps more investigation is required to understand its ability to discriminate between musical works. The role of the mel scale when constructing MFCCs is that it is used to construct a filter bank, which is applied to the frequency domain. Li & Chan (2011) investigates the placement of these mel-scaled filters for the task of genre classification. They find that MFCCs are affected by the key that music is played in, and overcome this problem by normalising the key of each track before extracting MFCCs.

These observations, that the mel-scale is suitable, but that it is affected by musical key, are both very interesting. Western music is defined by an equally tempered scale (Ellis 1885), which when used as a filter bank produces a pitch histogram. These results suggest therefore that there may be a meaningful relationship between the two scales, which if identified could help adjust how the mel-scale is utilised. This relationship will be investigated in detail, and used to form the basis for a music-inspired texture representation in Chapter 5.

Further investigations into using MFCCs for music focus on resolution, both of the input, and of the coefficients retained from DCT. Pachet & Aucouturier (2004) and Sigurdsson et al. (2006) both show that increased input resolution improves the quality of MFCCs. Neither paper provides a deeper analysis of why this is the case however. In further work, Aucouturier, Pachet & Sandler (2005) investigate the relationship between the number of coefficients retained after DCT, and the number of components used in a Gaussian Mixture Model (GMM) that models these coefficients. Unlike input resolution, the authors find that increasing the DCT resolution beyond a threshold is harmful to the GMM. These questions over resolution, and the balance between input and output may be related to the observations discussed relating to musical scale.

To further understand why increasing the resolution of DCT was found to be harmful, the rational behind using DCT must be understood. In the original work on MFCC for speech recognition, Mermelstein (1976) find that the first two coefficients obtained from DCT describe 97% of the variance between syllables, and the first three coefficients describe 99% of variance. Yan, Zhou & Li (2012) realise however that when multiple accents are included, the ability of DCT to capture this variability declines. To overcome the problem, the authors re-distribute the placement of filters, thus allowing DCT to once again capture variability in a succinct manner.

Differing accents centre around different frequencies (Evans & Iverson 2004), and thus DCT must be able to summarise these differing frequency ranges. However, as previously discussed, Li & Chan (2011) show MFCCs for music are also affected by pitch. Yan et al. (2012) cluster their filter-adjusted MFCCs using a GMM, which has the ability to group similar accents together, and Li & Chan (2011) choose instead to normalise data prior to calculating MFCC. The lengths that authors have gone to, simply to maintain DCT's ability to decorrelate information in a meaningful way, brings into question how necessary decorrelation is for music at all. DCT has the ability to separate signal from noise, but it is not the only method which can achieve this. When developing a new approach to music-inspired texture, this DCT step will be examined more closely, and its suitability for describing musical texture tested.

2.3 Tag-Based Representations

Tag-based representations have become increasingly more important to content-based recommender systems since the birth of the social web. For some music recommendation problems, tags alone have been shown to provide better quality recommendations than collaborative filtering methods (Firan, Nejdl & Paiu 2007). The tags used to construct these representations come directly from users, and not from record label meta-data. This means that a wealth of social and cultural knowledge, created by the implicit collaboration of users, can be available to describe tracks. Tags also introduce a blur between the classical notion of a recommender system being either content-based or collaborative filtering. In the strict sense tags are content, but because they are created collabora-

tively, recommendations made based on tags are influenced by other users, which classical content-based systems are not.

In this section different methods to collecting tags will be presented, and the type of knowledge that they capture will be discussed. Because tags are generated collaboratively, tracks with this representation suffer from the cold-start problem in the same way as collaborative filtering approaches do. This cold-start problem, and the reasons for it occurring will be examined. Finally, the effects that this problem has on recommendation will be analysed.

2.3.1 Collecting Tags

Turnbull, Barrington & Lanckriet (2008) discuss five different ways in which a tag-based representation can be collected for each track. These are:

1. Survey - Tags are provided by either users (Turnbull, Barrington, Torres & Lanckriet 2008) or domain experts completing a survey (e.g. Pandora[6]);

2. Users provide tags through playing a game, such as Tag-a-Tune (Law & Von Ahn 2009) or MajorMinor (Mandel & Ellis 2008);

3. Auto-Tagging - where tags are learned for a track based on audio content (Sordo, Laurier & Celma 2007);

4. Mining Web Documents - indexing frequency and meaningful terms in documents returned by a search engine (Knees, Pohle, Schedl, Schnitzer & Seyerlehner 2008); and

5. Social Tags - Tags which users provide to a music service while listening, for example, as with Last.FM.

Each of these methods come with their own specific advantages and disadvantages. Tags generated by survey tend to only cover collections of limited size, since it is not practical for users to survey millions of tracks. Similarly, games also suffer from this problem, with Tag-a-Tune having collected tags for around $30,000$ tracks. This is impressive, but

[6] http://www.pandora.com

it is still only a small set of tracks when compared to the remaining three methods. One advantage that survey and game approaches do have however is that the tags are generally clean, well-structured, and complete. This makes them excellent datasets to use for many research tasks, but perhaps not for large scale recommender systems.

Auto-tagging is an automated way of collecting tags, where tags are learned algorithmically. Given a training set of annotated tracks, the task is to propagate these tags to tracks within a test set, and thus expand the tag coverage of the collection. While this approach may guarantee a certain degree of tagging throughout a collection, the tags are not generated by humans, and thus it is likely that erroneous tags will be propagated to many tracks. Mining web documents allows for a rich vocabulary of tags to be learned, based on the contents of documents related to individual artists and tracks. However, as Levy & Sandler (2007) discuss, this vocabulary is too rich, with a large degree of noise introduced by irrelevant web document, and non-musical words used throughout the documents.

The final approach to collecting tags is social tagging. This approach has become the primary focus of many tag-based recommender systems over recent years (Rendle & Schmidt-Thieme 2010, Bogdanov & Herrera 2011, Musto, Semeraro, Lops, de Gemmis & Narducci 2012), helped hugely by the merging of Last.FM and AudioScrobbler in 2005, allowing access to the service's social tags through their API. As users browse and listen to music, the service allows them to add any short free-text description to a track. In an analysis of how people tag, Bischoff, Firan, Nejdl & Paiu (2008) finds that approximately 20% of these tags relate to user opinion, 20% to usage / task, and just under 20% are self-referential. Users also tag about topic, genre, and the track's artist. This provides a very rich, and relevant set of tags that describe tracks. As each user tags a track, they contribute to the collective tag representation for that track. In the case of Last.FM, the most popular tag(s) for each track is given a weighting of 100, and every other tag a proportional weighting of less than 100. These weights can be used to construct a tag vector for each track, and similarity between tracks is calculated with techniques built upon the vector-space model.

2.3.2 The Cold-Start Problem

Social tagging allows multiple users to contribute tags to a single track, thus strengthening the general description of that track. However, not all tracks within a collection are tagged equally; popular tracks tend to have a large number of tags describing them, and lesser-known tracks may have no tags at all. These two extremes are often referred to as the short-head, and the long-tail, referring to the distribution of tracks regarding how well-tagged they are. A small number of tracks belong to the short-head, and are well tagged, and a much larger number of tracks belong to the long-tail, and are not well-tagged (Levy & Bosteels 2010). An example of this skewed distribution can be observed within the Last.fm contribution (Last.fm dataset 2012) to the Million Song Dataset (Bertin-Mahieux, Ellis, Whitman & Lamere 2011), where 46% of tracks do not have any tags at all. This distribution is illustrated in Figure 2.2. On average each tagged track has 17 tags. The 25,000 most popular tracks each have 100 tags assigned to them, but this number very quickly drops off, leaving the majority of tracks being under-tagged.

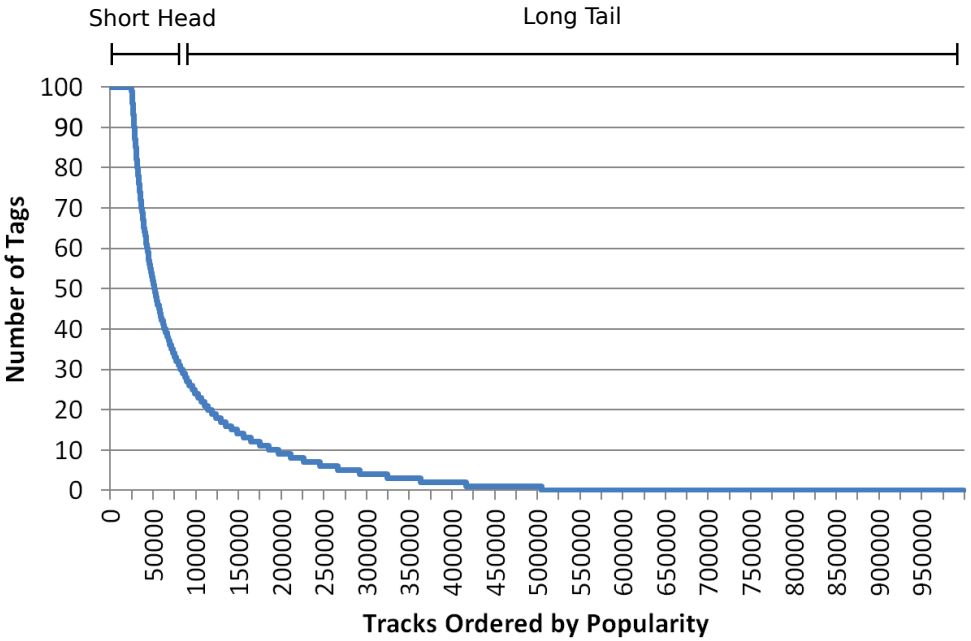


Figure 2.2: Level of Last.FM Tagging Within the Million Song Dataset

This long-tail introduced by user tagging has a major impact on music recommendation; it leads to the cold-start problem, where tracks that are not well-tagged are likely to

be excluded from recommendations (Lamere 2008). It is likely that users will know popular tracks from the short-head and will listen to them. However, Celma & Cano (2008) show that from these tracks, a content-based recommender system will only recommend tracks from the long tail 29% of the time. Even if one assumes all of these recommendations are good, and thus users are likely to tag the tracks, users are still given the opportunity to tag popular songs much more frequently. It is unlikely that the cold-start problem will be reduced simply through user usage, and thus recommender systems must be able to reduce the effects of cold-start, but must do so while maintaining recommendation quality.

2.4 Reducing Cold-Start

Tag-based representations, and the recommender systems that use them, suffer from the cold-start problem. Systems which use audio content-based recommender systems do not suffer from this problem, but the quality of recommendations is much weaker. Hybrid approaches to representation are required, where the benefits of tags are combined with the ability of content to avoid cold-start. There are many ways in which a hybrid approach can be achieved (Burke 2002), and the most relevant approaches to music are:

- Weighted Scores - The similarity scores of track recommendations from tags and content are combined;
- Mixed - The recommendations of both tags and content are combined into a longer recommendation list;
- Cascade - A coarse recommendation list is made using one method, and then refined by the other; and
- Combination - The tag and content features are combined into a hybrid representation.

The method of weighting scores has not had much recent attention, since to be effective, the method assumes that the relative values of both approaches (content and tag) are uniformly distributed. In practice this is not the case, since cold-start tracks are likely to always be given low scores by a tag-based recommender. In less recent work, Li

(2000) investigated using a nearest feature line method to weight similarity among differing dimensions of content, and then aggregate the correlations between these scores, and Shardanand & Maes (1995) use profile similarity scores to weight how track recommendations are scored. There does not appear to be any related work that successfully investigates how weighting scores between content- and tag-based systems may be applied.

The FlyTrap recommender system (Crossen et al. 2002) produces candidate recommendations using different agents, and then makes recommendations by sampling the distribution of tracks returned. This approach is interesting, but is perhaps best suited to group recommendations (Crossen & Budzik 2006). A similar approach to FlyTrap is to first generate recommendations for each individual tag in a track's tag-vector, and then combine the recommendation lists (Cai, Zhang, Zhang & Ma 2007). This approach is shown to produce good recommendations, and to be more efficient than approaches using tag vectors. However, while this method may reduce the effects of cold-start by matching individual tags, rather than vectors, it cannot cope with the extremes of cold-start any better than vector-based approaches.

Cascade approaches offer an improvement over the mixed approach, because they allow the strengths of both content and tags to be included when it is appropriate. Bogdanov & Herrera (2011) compare recommendation based on a combination of content and tags (C-SEMLL), with genre (M-GENRE), and using only tags (M-TAGS). The authors find that C-SEMLL and M-GENRE both perform worse than M-TAGS. However, when C-SEMLL and M-GENRE are designed as a cascade approach, where M-GENRE refines C-SEMLL, this cascade is able to outperform all individual methods. In terms of cold-start, genre is a reasonable facet to cascade to, since this data is typically provided for all tracks as meta-data. However, one may question if this is the best possible approach to cold-start, since genre only captures part of the meaningful description of a track. Similar approaches have been examined where content is refined by more meaningful information, such as relevance feedback (Hoashi, Matsumoto & Inoue 2003) and user profiles (Grimaldi & Cunningham 2004), but these both have potential to suffer from cold-start.

Combination approaches are by far the most researched and promising direction for reducing cold-start in music recommender systems. There are two main approaches to combination methods for music. Auto-tagging (Bertin-Mahieux, Eck & Mandel 2010)

combines the content and tag representations by propagating tags to cold-start tracks based on content. Hybrid approaches combine content and tags into a new single representation (Levy & Sandler 2009).

2.4.1 Auto-Tagging

The task of auto-tagging is fairly recent, and has gained much interest after its inclusion in the MIREX 2008 competition. The objective of this task is to learn a set of tracks that are relevant to a tag, thus propagating tags throughout the collection. To achieve this, each tag is modelled in terms of content, and a fixed number of tracks that most closely match this model are assigned the tag. The content used to model each tag is the combination of content from each track that has this tag applied to it, usually irrespective of how strong the tag assignment is. Given a set of untagged tracks, the content of each track is modelled, and for each tag the track's with models most similar to the tag model are assigned the tag. Thus, the mapping is from one tag to multiple tracks. The implication of this is that every tag is used, but not every track is guaranteed to be assigned a tag.

A standard approach to auto-tagging is to learn a Gaussian Mixture Model (GMM) for each tag, using the audio features of each track the tag is applied to. Given an untagged track, the probability of the track's audio features belonging to each tag GMM is determined, and the most likely tags assigned. Barrington, Turnbull & Lanckriet (2008) also model content vectors using a GMM, allowing auto-tagging to be achieved by determining the probability that a track's GMM would generate the tag GMM. This approach was the best performing auto-tagging approach in the MIREX 2008 competition.

More recently, Tardieu, Charbuillet, Cornu & Peeters (2011) extend this approach by constructing an Universal Background Model (UBM), describing the entire collection of tags as a single GMM. An individual GMM is then learned for each individual track by modifying only the mean-vectors of the UBM, and thus each track is represented by the GMM mean vectors. This approach is more computationally efficient, and allows new, cold-start tracks to be tagged quickly. One drawback with the UBM approach, and many GMM approaches in general, is that efficient implementations consider each dimension to be independent, because of the use of a diagonal covariance matrix. Miotto, Barrington & Lanckriet (2010) overcome this problem by introducing a further step that refines auto-

tag vectors based on a Dirichlet mixture. This approach has the advantage of including co-occurences between tags, which assumes new tags that are learned should be related to tags that already exist. While this introduction of co-occurece knowledge allows their approach to out-perform GMMs, the authors evaluated the approach using a dataset with very clean and well-structured tags. Thus, the approach runs the risk of excluding more niche tags, which may be most appropriate for many cold-start tracks. It is easy to learn common tags which co-occur often, but it is difficult to learn the more detailed tags.

A similar approach to GMMs is the Codeword Bournoulli Average model (Hoffman, Blei & Cook 2009), which predicts the probability that a tag should be used to describe a track. This method provides good baseline performance, and is often used as a comparison. However, each of the approaches described have one problem in common. The methods all treat auto-tagging as a classification task, where a tag is the input, and a track is the class. In practice this is used by assigning each tag to a fixed number of tracks, and thus one would expect many tracks to go un-tagged. Of course, it is possible to determine the most likely tags for each track and then assign these tags, but this is not how the models were designed. Reducing cold-start in this way is likely to introduce a lot of noisy tags to tracks where only marginally relevant tags are assigned: less common tags, which may be relevant to less popular tracks, are unlikely to be strong components in the distributions learned by each model.

A much more simple approach, which does not rely on modelling distributions, has been used initially for both auto-tagging style and mood (Sordo et al. 2007, Sordo et al. 2012), and semantic annotation (Kim, Tomasik & Turnbull 2009). Both approaches treat retrieval methods as a black box, which provide a set of similar tracks. The tag-vectors of these similar tracks are then aggregated, and the most occurring tags are propagated. The advantage of this method is that neither author attempts to correlate content directly with tags, or presume that tagging must fit any prior distribution. Instead, the assumption is that similar tracks will have similar tags, and thus tags not directly related to content will also be discovered. Further, this assumption does not constrain the number of tags that will be propagated to tracks, and removes the problem of niche tracks going untagged. Both approaches use relatively simple aggregation techniques that count how often each tag appears in similar tracks. Unlike with most auto-tagging approaches, the mapping

in this case is from one track to multiple tags, with the implication that every track will receive an equal number of tags. It would be interested to investigate this approach further, and examine if more appropriate aggregation of tags from similar tracks would lead to stronger representations being propagated to cold-start tracks.

2.4.2 Hybrid Representations

Hybrid representations attempt to combine two or more single representations into a single joint representation. One of the main issues in achieving this it to first structure each representation so that it is natural to combine them. Levy & Sandler (2009) achieve this by first creating a code-book from audio content vectors. Each vector in the codebook is referred to as a musword, and each content vector describing a track is assigned to a single musword. Thus, each musword may be thought of as the audio equivalent of a tag, and a musword-frequency vector is constructed for each track in a similar way to tag-frequency vectors: the frequency of each musword is the number of times a content vector is assigned to that musword. The musword-frequency vector of each track is concatenated to the track's tag-frequency vector, creating a hybrid representation. Finally, the authors generalise these hybrid vectors using probabilistic latent semantic analysis, learning concepts which describe both content and tags. The advantage of learning such a concept space is that tag information is naturally propagated to cold-start tracks. However, this method relies strongly on the strength of the vector codebook, and may be affected by the vocabulary size.

The approach of first combining representations, and then generalising them, is an area that has received a lot of attention. Different combination include audio content and tags (Levy & Sandler 2009), audio content and user ratings (Hu, Guo & Zhang 2009, Yoshii & Goto 2009), and audio content, users, and tags (Nanopoulos, Rafailidis, Symeonidis & Manolopoulos 2010). Intuitively, sparsity in one representation (cold-start) may be overcome by learning correlations with another representation. However, while this approach may improve overall performance, it does not follow that it helps for cold-start. Generalisation techniques typically look for strong correlations within data, and define these as concepts. The strongest correlations are likely to be related to popular, well-tagged tracks, and not the less popular cold-start tracks. This introduces an assumption

that cold-start tracks should be tagged in the same way as well-tagged tracks, and this may not be the case. For example, rock related tags are likely to be popular, but death-metal tracks are likely to be under-tagged. Adding rock tags to a death metal song is perhaps not incorrect, but it is not the best possible tags that could be assigned. It is also likely that only content related tags will correlate closely with tags, limiting the potential of such systems.

The method presented by Nanopoulos et al. (2010) looks for correlations between content, users, and tags. This avoids the problem of restricting correlations to being content related, since the users provide a common link between content and tags. To achieve this the decomposition of a tensor, rather than matrix, is required. This introduces a higher level of computational complexity, and it is questionable how scalable this approach is. However, the general idea is interesting, and perhaps utilising a different, more efficient common link between content and tags would define further approaches to cold-start.

2.5 Evaluating Recommender Systems

The evaluation of newly developed recommender systems is a crucial step to ensure their effectiveness. There are several properties of recommender system that can be evaluated, including recommendation quality, discovery, and novelty for example. Each of these properties are important when evaluating cold-start recommendation: discovering cold-start tracks, and making novel recommendations is important, but only if quality can be at least maintained. Measuring these properties can be achieved using either a system-centric, network centric, or user-centric evaluation methodology (Celma 2010). Ideally a user-centric evaluation, where users engage with a live recommender system, is used. In this case the user study can be easily tailored to collect relevant information about each measure that must be evaluated. However, many researchers do not have the luxury of a user study, and so systems are evaluated using system- and network-centric approaches. In this section these approaches will be examined in detail.

2.5.1 System-Centric Evaluation and Quality

System-centric evaluations make use of existing data for users of a recommender system, typically from listening logs or profiles. One method of evaluating quality following a system-centric approach is to obtain logs of what individual users like, remove n tracks, and measure how accurately a recommender system can recommend these held-out tracks (Breese, Heckerman & Kadie 1998). This approach has been followed based on user data from Amazon (Su, Yeh, Yu & Tseng 2010), Yahoo! (Koenigstein, Dror & Koren 2011), The Echo Nest's Taste Profile (McFee, Bertin-Mahieux, Ellis & Lanckriet 2012) and Last.fm (Bu, Tan, Chen, Wang, Wu, Zhang & He 2010). However, this method is not always appropriate for evaluating recommender systems when cold-start recommendation is important. The Amazon dataset is very commercially driven, and thus may not contain enough information about less popular tracks. The dataset from Yahoo! anonymises both users and tracks, and thus is only really applicable when collaborative filtering is used. Finally, the Taste Profile and Last.fm datasets suffer from popularity bias, and so it is unlikely that many users will have listened to cold-start tracks.

Jawaheer, Szomszor & Kostkova (2010) examine how user data can be used more effectively, by considering both implicit and explicit feedback. Implicit feedback occurs when a user listens to a track, as with the previous listening history examples. This type of feedback can introduce noise because it assumes that if a user listens to a track they also like it. Explicit feedback considers tracks which users have explicitly said they like, through some kind of rating mechanism. Evidence exists of noise also occurring in explicit feedback, but to a much lesser extent (Amatriain, Pujol, Tintarev & Oliver 2009). The main form of noise with explicit feedback is opinions changing over time (Amatriain, Pujol & Oliver 2009), and this can be managed by only considering explicit feedback over a fixed time period. Jawaheer et al. (2010) compare evaluations using both data types, and highlight the need for evaluation strategies to include both implicit and explicit feedback. There does not appear to be any literature on how such combinations could be constructed however, and the question of how predicting user listens and likes could be used jointly to evaluate recommendations requires further research.

User listening and rating habits can be used to construct further evaluation strategies,

not based on predicting individual user feedback. Instead, the collection of user feedback can be considered as a whole, and relationships learned. This is different from collaborative filtering because the goal is not to find similar users and make recommendations, but to find similarities and trends amongst all users and tracks. Ellis et al. (2002) achieve this by constructing ground truth score based on the co-occurence of tracks in user profiles, a method they call cultural similarity. This score is weighted by a popularity cost function, which reduces the score when the popularity of the query and recommendation differ greatly. The approach is interesting because it introduces a structured way of handling popularity bias within a collection. However, the specific cost function used penalises recommending rare, cold-start tracks from popular queries. Eck et al. (2007) describe a similar approach, but instead of using a cost function they treat every track as a document, and every user that listened to the track as a term. TF-IDF weighting is used to reduce the influence of popular tracks and the most active users, and similarity is used as a ground truth score. The TF-IDF weighting allows cold-start tracks to be considered more equally, but at the cost of down-playing the relationships between popular tracks, the opposite effect of cultural similarity.

Complications over defining a suitable ground truth dataset and score has, in part, had the consequence that many evaluations do not use user data, and instead use a proxy to user quality. A good recommender system must be able to calculate similarity effectively, and so a lot of work has used genre classification as a means to test how well similarity is calculated. Neumayer & Rauber (2007) motivate this decision by highlighting that users often want to listen to tracks that belong to a similar style. In an extremely comprehensive review of genre recognition (Sturm 2012), it is found that 91.3% of 467 papers focus on genre classification for evaluation. However, only 10 of these consider the problem as a multi-class problem. This is important, because while it is possible that recommendation quality and genre might be related, it is difficult to imagine that how recommendation quality can be examined fully without considering multi-class solutions. The final evaluation measure that it has been suggested be used for investigating recommendation quality is year prediction (Bertin-Mahieux et al. 2011). The justification is that users may have a preference towards music from certain time periods, and that many genres also relate to these periods. However, this score does not appear to have been formally studied.

2.5.2 Network-Centric Evaluation and Discovery

Network centric evaluations involve creating a track recommendation network, where each node is a track, and edges reflect user listening habits. That is, if 10 users have listen to tracks A and B, then an undirected edge between A and B is created with a weight of 10. This structure cannot be used to measure recommendation quality, but it does provide means to measure the coverage of a system, or how well cold-start tracks are discovered. One method to measure discovery is to examine the minimum distance between query and recommendation for example (Vargas & Castells 2011). By also including measures of popularity for each track, this approach may be extended to examining novelty. However, any offline evaluation of novelty must be interpreted carefully, since it cannot be known what tracks users do not know (Zhang, Séaghdha, Quercia & Jambor 2012).

(Celma & Herrera 2008) measure discovery by correlating the evaluation network with the long-tail distribution illustrated in Figure 2.2. The allows the identification of where in the distribution a query and recommendation occurs, and the distance between them. Both Celma & Herrera (2008) and Zhang et al. (2012) also conduct user-centric evaluations to test their approaches. In each case, network-centric evaluation of discovery was found to be a good indication of novelty. The results presented by Zhang et al. (2012) however also show that discovery may be measured without having to construct and examine the network. Instead, popularity and level of cold-start can act as a good indication. Network analysis provides a fuller picture, but it is not always necessary.

2.6 Summary

The field of music recommendation as a part of computer science is relatively recent. Since 2005, and the birth of the social web, there has been an explosion of interest in this field from both industry and academia. Systems are becoming successful enough that many people now use recommender systems as a primary way of interacting with music collections. In general, there are two main approaches to recommendation, either employing user information, or track content to make recommendations. While collaborative filtering is limited to what a large number of users have already rated, content-based approaches have the potential to include much more new and exciting discoveries for users.

Two different types of representations exist for content-based recommender systems; audio content, and track meta-data such as tags. Audio content features aim to describe important features of a track, including chroma, rhythm, and harmony. Of these audio features, texture has proven to be one of the most powerful when making recommendations. Techniques for describing texture, however. come from the field of speech processing and recognition. This brings into question how suitable standard texture descriptions are for music. Results using speech texture techniques outperform musical techniques such as chroma, but there is evidence to suggest texture can be further improved, thus improving the ability to make recommendations.

Tag representations are most commonly constructed through social tagging, where individual users can apply short free-text descriptions to any track. These tags can then be aggregated for each track, constructing a tag vector that reflects many users opinions and labels. Recommendation quality using social tags is stronger than when audio content is used, but levels of social tagging are influenced heavily by popularity. This means that only popular tracks are likely to be tagged strongly, leaving less popular tags under-tagged. The effect of this under-tagging means that the tracks suffer from the cold-start problem, where they are likely to be excluded from the system.

For content-based recommender systems to be used to their full potential, allowing novel, high quality discoveries to be made, this cold-start problem must be addressed. The use of audio content alone does not produce high quality recommendations, and so approaches to bridging the gap between audio content and social tag representations are required. One existing approach to achieve this is auto-tagging, where tags are propagated to cold-start tracks. Approaches to auto-tagging rely on modelling the distribution of tags within a collection, or the distribution of audio that describes a tag. However, many tags do not describe music, and so do not fit into the distributions that are learned. New approaches to propagating tags to cold-start tracks are required, that allow non-musical tags to be more accurately propagated.

A further approach to bridging the gap between audio features and tags that has gained much interest is hybrid representations. Many techniques to creating hybrid representations first structure both audio features and tag features to be comparable, typically through clustering. The representations are then combined and generalised, with the in-

tention that cold-start tracks will be described by hybrid concepts. However, these hybrid concepts are likely to be learned from correlations that exist with well-tagged tracks, and these correlations may not exist within under-tagged tracks. New hybrid techniques that do not rely on such correlations are required, and may increase the potential of recommender systems.

Evaluating music recommender systems is ideally achieved by a user-study. However, due to the datasets used, and requirements for rapid evaluation, user-studies are not always possible. System-centric approaches to evaluation attempt to indicate the results of a user study by using existing user listening and rating history. Listening data can be interpreted as implicit feedback, where is assumed that if a user listens to a track they also like it. Rating a track is explicit feedback, and more reliable, but does not provide as much data. Existing techniques make use of either, but not both forms of feedback. In both cases this can cause problems, such as noise or lack of data, and thus further techniques which use both implicit and explicit feedback are required. In terms of evaluation for when cold-start tracks must be evaluated reliably, existing techniques suffer from popularity bias in the way they define ground truth scores. New techniques to reducing the effects of this bias are required, allowing for more consistent and reliable evaluation of cold-start tracks. Many researches use proxies to user feedback, such as genre classification, to achieve this, but ideally techniques making use of feedback should be explored.

Chapter 3

Datasets

Techniques for music recommendation can only be developed if all of the required data is available. This data includes audio files, meta-data, social tagging data, and user listening profiles. Audio files are required in order to extract and develop new content-based representation techniques. Meta-data typically accompanies audio files, describing properties such as artist name, genre, and year. Social tagging data provides a wealth of knowledge-rich information supplied by listeners of a track, and each listeners' profile contains information on the songs they listen to, and the songs they love.

Part of the focus of this thesis is to develop a new music-inspired texture representation. To achieve this, a collection of audio tracks must be available, so that content-based representations may be extracted directly from the audio. The important characteristics of such a dataset is that the audio is available for each track, and that this audio is of sufficiently good quality. Low quality audio introduces noise, and this can lead to unreliable approaches that are unable to fully exploit the properties of music. Each track should also have a tag-based representation, since a further focus of the work in this thesis is to integrate content and semantic representations. Finally, an approach to evaluating methods developed using a dataset must be possible, which may include further data such as user listening and rating information.

Of the publicly available datasets at the time of starting this work, only the CAL500 was accompanied by audio tracks. In this chapter this dataset is first described, in terms of

the audio files, and meta-data available. The set of well-structured tags that are included in the CAL500 dataset are also described, and the potential problems with using this data to construct real-world recommender systems.

A review of the CAL500 dataset shows the need for a larger, and more realistic dataset to be available. Therefore, a private dataset is constructed. The audio files used for this dataset are described, and also the meta-data that describes them. The process of collecting tags to describe this dataset is presented, and the properties of this tag collection are discussed. Finally, a comparison between the privately constructed dataset and the publicly available CAL500 dataset is made, highlighting the strengths and weaknesses of each for developing new music recommender systems and representations.

3.1 CAL500 Music Dataset

The CAL500 dataset (Turnbull, Barrington, Torres & Lanckriet 2008) was created at the Computer Audition Lab, University of California, San Diego (UCSD) in 2007. The dataset consists of 500 tracks, and each track is recorded by a unique artist. Some standard meta-data is provided, and structured tags that were assigned using a survey are also present.

3.1.1 Audio Files

Each track in the CAL500 dataset comes with a monophonic audio clip of low quality, encoded as a 32kbps mp3 file with a sampling frequency of 22.05kHz. As a reference, CD quality audio has a bit-rate of $1,411.2$kbps, and so audio in the CAL500 has a compression ratio of 44:1. Audio bit-rate can be thought of in a similar way to pixel density; in either case decreasing the amount of information within each time or space frame reduces quality, which may reduce the accuracy of any processing using the data. The maximum frequency which can be extracted from audio is half the sampling frequency, and the range of human hearing is roughly 0.02kHz to 20kHz. Thus, with the sampling frequency of 22.050kHz, the maximum frequency that can be described is 11.025kHz, meaning that nearly half of what humans can hear is not present. Each clip is sampled to be somewhere in the middle of each track, and so typically does not include the introduction or ending of each song. The left and right channels are also not present, since they have been flattened into a

single channel, most likely to save space.

The dataset includes pre-extracted content-based feature vectors for each track. The feature used is MFCC, which is extracted from the audio using a 12ms half-overlapping window. The first 13 coefficients are retained, and the instantaneous first and second derivatives of each MFCC is appended, creating a final feature vector that contains 39 dimensions. To avoid the possibility of reconstructing the original audio files, the creators of the dataset randomly sample $10,000$ vectors for each track. This removes the ability to define representations that are based on time orderings such as rhythm, but does not affect non-temporal representations such as chroma and texture.

3.1.2 Meta-Data and Structured Tagging Data

The only meta-data that accompanies each track in the CAL500 dataset is the track names. However, one main contribution of this dataset is the structured tagging data that comes with it. The creators of the dataset paid 66 undergraduate students to annotate the dataset with a fixed set of tags. These tags fall into the categories shown in Table 3.1.

Number of Tags	**Category**	**Description**
29	Instruments	Annotated as present or not
22	Vocal Characteristics	Annotated as relevant to the singer or not
36	Genres	A subset of the Cadaich (McKay, McEnnis & Fujinaga 2006) genre list, annotated as relevant or not
18	Emotions	Found by Skowronek, McKinney & Van De Par (2006) to be important and easy to identify. Rated using 3 point scale
15	Song Concepts	Acoustic qualities of the track, artist, and recording
15	Usage Terms	e.g. 'I would listen to this song while driving'

Table 3.1: CAL500 Structured Tags

At least 3 annotations were provided for each track, and these were expanded into words, based on how each category tag was assigned. These annotations were then constructed into a 'soft' and 'hard' annotation matrix for the collection, where soft describes

the strength of each tag-track assignment, and hard describes if the strength is above a threshold or not.

In terms of having a fully represented, and structured tag representation, the CAL500 dataset is an excellent resource. Common tasks for which this dataset is used is genre and emotion classification, and auto-tagging. However, for certain other tasks this dataset comes with limitations. One such limitation is that the dataset is designed towards using only audio content, since tags or tag categories are often used as evaluation data. For approaches that aim to make use of both content and tags, such as hybrid representations, the dataset is not comprehensive enough to allow for meaningful evaluations.

3.2 Private Music Dataset

In order to have data that can be used both for constructing hybrid representations, and then evaluating recommender systems, a private dataset has been created. This is not a personal collection, and was created by purchasing CDs from a range of different genres and years. The inclusion of many compilation CDs also helped to keep the collection diverse. This dataset includes 3174 tracks, across a wide range of musical styles and periods. A full mp3 file is available for each track, along with rich meta-data. Social tagging data has also been collected from the Last.FM web service.

3.2.1 Audio Files

Each track in the dataset has a corresponding stereo mp3 file, which is encoded at 320kbps with a sampling frequency of 44.1kHz. This encoding allows for minimal signal loss due to compression, and for a detailed analysis of the signal in each file. Since the full audio of each track is available, any chosen content-based feature can be extracted, and time ordering can be preserved if required.

3.2.2 Meta-Data

Each track was imported from CD using the iTunes media player, and thus rich meta-data is available. While this meta-data is not necessarily used by algorithms, it is useful to provide a clear picture of the nature of the dataset. The dataset contains 3174 tracks,

by 76 artists, from 277 albums, many of which are compilation CDs. Table 3.2 shows the genre classifications of these tracks. The majority of the collection can be described as popular western music, which has a huge importance in current recommender systems.

Genre	Number of Tracks	% of Collection
Alternative	918	28.92
Pop	797	25.11
Rock	659	20.76
R&B	362	11.41
Dance	127	4.00
Metal	76	2.39
Folk	55	1.73
Rap	54	1.70
Easy Listening	40	1.26
Country	38	1.20
Other	48	1.51

Table 3.2: Private Dataset Genres

Table 3.3 shows the decade that each track in the collection was released. The dataset primarily contains tracks that were released between the years 2000 to 2009. This collection was created in 2009, and so to ensure new results are comparable with old, the collection has been unmodified. This means no tracks since 2009 are included. As well as containing many recent tracks, the collection also contains a large number of tracks from previous decades. This spread across decades helps to make the collection more diverse and more closely replicate a current recommender system.

Release Decade	Number of Tracks	% of Collection
2000s	2046	64.46
1990s	595	18.75
1980s	166	5.23
1970s	150	4.73
1960s	93	2.93
Pre-1960	59	1.86
Unknown	65	2.05

Table 3.3: Private Dataset Release Decades

Figure 3.1 illustrates the lengths of each track in seconds, showing that the majority of tracks are between three and four and a half minutes long, Tracks that are longer than

5 minutes tend to be classical works, and tracks shorter than a minute tend to be interval tracks on albums.

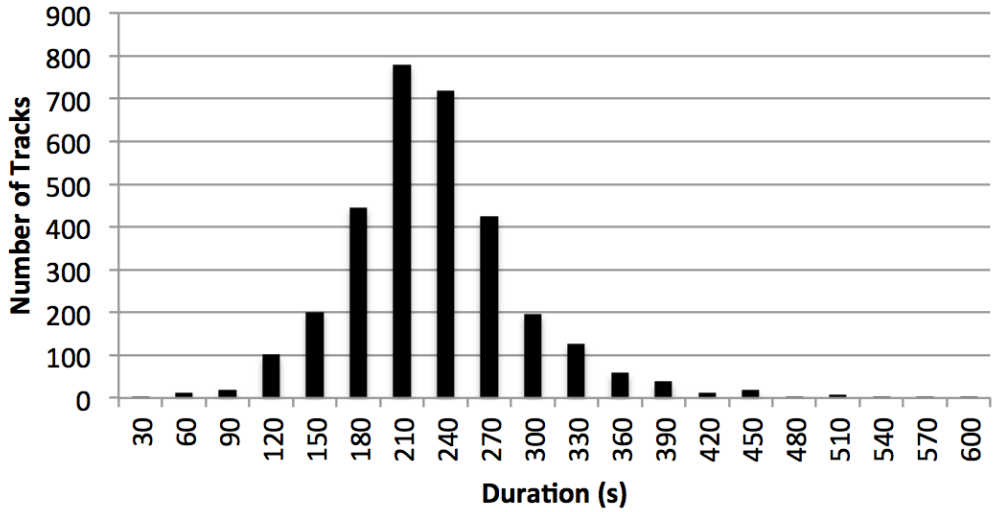


Figure 3.1: Private Dataset Track Durations

3.2.3 Social Tagging Data

Tags for this dataset are collected from the Last.FM API[1]. Last.FM is an online music recommender service, centred around providing users with personalised radio stations. As users listen to music, they are able to tag each track with any free-text description they choose. The API they provide into their index of data contains the method `Track.getTopTags()`; this method is used to retrieve tags for each track in the collection.

A total of 5160 unique tags are collected for the collection. On average, each track has 34 tags assigned to it, and the standard deviation is 26.4. The maximum number of tags that any single track has is 99, and 94 tracks have no tag assignments. The weights that Last.FM applies to each track-tag pair is also stored. These weights are in the range 0 to 100, where the strongest tag always has a weighting of 100. This information allows a knowledge-rich tag-based representation to be defined for tracks in the collection, providing a strong baseline recommender system.

Table 3.4 shows the most popular tags that describe the dataset. The fourth column shows where each tag appears in a similar list compiled by Last.FM for their contribution

[1] http://last.fm/api

to the Million Song Dataset (Last.fm dataset 2012). Seven of the most popular tags in the private dataset appear in the top 10 most popular tags in Last.FM. This serves as a good indication that the dataset is reflective of real-world music collections used in online recommender systems. The top Last.FM tags missing from private dataset's top 10 list are electronic, female vocalist, and dance, which occur at positions 30, 13, and 15 respectively.

Position	Tag	Number of Tracks	Last.FM Tag Position
1	rock	1945	1
2	pop	1586	2
3	alternative	1402	3
4	favourites	1704	7
5	alternative rock	1043	11
6	indie	1001	4
7	Love	940	8
8	00s	908	10
9	british	898	26
10	indie rock	732	21

Table 3.4: Most Popular Tags

3.3 Comparison of Private and CAL-500 Music Datasets

The strength of the CAL500 dataset is that it is clean, complete, and allows for the comparison of work presented by different research teams. If an author publishes the code for their algorithm online, then it is possible for anyone to replicate their work exactly, since an identical copy of the dataset used is available. However, for some tasks the CAL500 dataset comes with some drawbacks. Firstly, while standard audio features are available, the audio files contained in the dataset are of a low quality. More importantly however, the files are incomplete representations of each track, and thus may not always be suitable for developing new methods.

The tagging data accompanying the dataset is very strong, but does not necessarily reflect the properties of data collected from a live music system. Figure 3.2 compares the distribution of tags per track in the CAL500 and privately collected datasets. The horizontal axis shows the number of tags a track has assigned to it, and the vertical axis shows the percentage of tracks within the collection that have this number of tags.

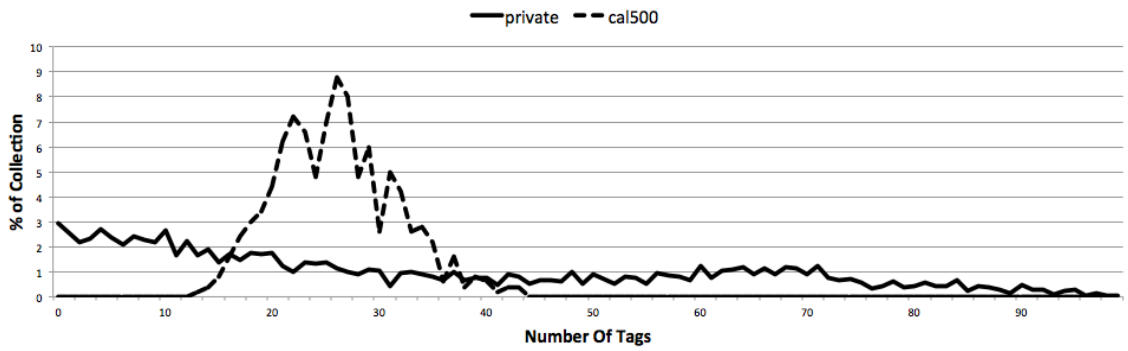


Figure 3.2: Comparison of CAL500 and Private Dataset Tag Distributions

The distribution of tagging for the CAL500 dataset is constructed using the hard-annotations provided, and is illustrated by the dashed line. Due to the survey methodology used to collect tags for this dataset, the minimum number of tags a track has is 13, and the maximum number is 43; there are no tracks with 0 tags. The distribution of tagging for the private dataset is illustrated by the solid black line, and is very different to that of the CAL500 dataset. The trend observed for the private collection is that as the number of tags increases, a smaller number of tracks will have that number of tags. This is an expected trend for a dataset generated by online users, and is the result of a popularity bias across the tracks being tagged.

3.4 Summary

These two datasets are used throughout the work presented in this thesis, based on their applicability to the task being examined. The strength of the CAL500 dataset as a standard benchmarking collection cannot be ignored, and thus for tasks where it is important to provide this benchmark comparison the CAL500 dataset is used. For tasks which involve constructing hybrid representations however, the privately constructed dataset will be used. This dataset is more comprehensive, and more accurately reflects what is observed in a popular online music service. Further, the CAL500 dataset was designed for the task of semantic annotation, not music recommendation, and as such does not come with strong evaluation data; use of the private dataset introduces the possibility of including such evaluation data.

Chapter 4

Evaluating Music Recommender Systems

One key component of conducting research on music recommendation approaches is the ability to evaluate the properties of a system, such as recommendation quality. This recommendation quality should reflect the opinions of listeners, since they are the end-users of any system using a new approach. Thus, an ideal evaluation of any approach is to allow users to interact with a system which implements a new recommendation method, and collect implicit or explicit feedback on each recommendation's quality. However, there are several obstacles to following this evaluation methodology that make it impractical. Firstly, it is costly to run a live music service due to licensing, and such systems must typically be run as a business venture, not research activity. More importantly though, if such a live system were to be available, users would expect the recommendations they are provided with to be of a consistent quality. When developing new approaches this consistency cannot be guaranteed, and expecting users to interact with a potentially poor recommender system can quickly turn them away from future engagement and evaluations.

Structured user studies suffer similar problems to that of a live system. While more affordable, users are not likely to repeatedly take part in evaluations of a system in which they are not more invested. At most, one may expect users to take part in such studies only every few months. What is needed therefore is a replacement evaluation method

that is able to give a reasonable indication of how users will rate recommendations. Such approaches are often referred to as offline system-centric evaluations, as opposed to an online user-centric evaluation.

System-centric evaluation methods rely on pre-existing user ratings, rather than asking users to provide new ratings for recommendations. Such data can be gathered from users' listening habits, profiles, and past ratings they have applied. This data is then used as a ground truth against which recommendations may be compared, and an evaluation score defined. To avoid over-fitting to this data, standard evaluation approaches such as leave-n-out or n-fold cross validation can be used. This system-centric approach allows for rapid, automated evaluation of new approaches to recommendation, that are not practical with user-centric evaluations. Once a new approach has been evaluated successfully with a system-centric evaluation, then these findings can be confirmed with a user-study. This satisfies both the requirement of a developer to constantly evaluate their work, and the need to maintain the interest of users willing to evaluate new work.

In many domains, such as text mining and image retrieval for example, the data required to utilise a system-centric evaluation approach is standardised, and freely available. Unfortunately for music however this is not the case. While the Million Song Dataset (Bertin-Mahieux et al. 2011) does indeed provide evaluation data, it does not provide access to audio files, but instead comes with pre-extracted audio features. This dataset is a good step in the right direction, but does not allow for the evaluation of work where new content-based features are being examined.

The absence of user data to evaluate music recommender systems has meant that proxies to system-centric evaluations are typically used. One key aspect of a recommender system is the ability to identify similarity successfully, and thus the annual MIREX campaign evaluates similarity by artist, genre, mood, and composer classification. While these classifications may loosely reflect the similarity between tracks, they do not provide a fine enough view of similarity. For example, Nirvana and Bon Jovi may both be classified as 'rock', but this does not mean suggesting Bon Jovi to a Nirvana fan is a good recommendation.

To overcome this problem there are two possible solutions: the availability of a dataset including audio files, tags, and user ratings; or a standard approach to constructing ground

truth data which provides a good indication of how users will rate recommendations. The former would require licensing of audio to researchers internationally, and so an approach to defining ground truth data is developed. In this chapter the desirable properties of such ground truth data are first discussed, followed by a description of the data which is available. From these discussions the Association Score measure is defined, based on user listening habits and 'likes'.

4.1 Desirable Properties of Ground Truth

A strong ground truth scoring mechanism must satisfy several properties, so that it may accurately reflect user opinions. This score must be based on ground truth data, and this data must be able to satisfy the required properties. For example, ground truth should not be influenced by a popularity bias within the data because refining algorithms based on such performance with ground truth would lead to a system that is likely to recommend popular tracks, and not necessarily high quality tracks. In this section each of the desirable properties are discussed.

4.1.1 Substantial and Complete

Ground truth data must reflect the tracks that users listen to, and the tracks that they rate. Every user however can have a unique taste in music, and while they may share many commonalities with other users, there is always likely to be a level of disagreement. Some people may listen to pop music in the morning, classical music at work, and dance music in the evening, and thus their profile would contain 'listens to' and ratings of tracks from a wide range of styles. It does not follow that recommending a classical track from a pop query is good though, since this is a mis-understanding of listening context.

To understand and deal with these variations across individuals, a large amount of data is required. For each listener, their profile must be substantial enough that outliers may be recognised and dealt with accordingly, and a good view of their general interests can be captured. The set of users must also be large, so that well supported trends or correlations may be recognised. Further, this set of users must be diverse, so that built systems are not biased towards a specific type of user, for example, one who prefers pop

music.

From a track perspective, the ground truth data must also be fairly complete. That is, there must be a substantial amount of ground truth data that reflects every individual track in the collection. The reason for this is to avoid penalising tracks as bad recommendations, when in fact they are good, but there is not data reflecting this fact. Further, completeness is important for evaluating tasks such as cold-start recommendations. Cold-start tracks are weak from a representational perspective, but must still have complete enough ground truth data to ensure correct evaluation measures can be defined for them.

4.1.2 Normalised Track Popularity

Tracks that are popular in the system from which ground truth is collected are likely to have much more ground truth data available. This should not be limited to match the least popular track in a collection, because this removes vitally important information. However, this increased data should also not affect any ground truth scoring that is used. Support based ground truth scores may be influenced by the popularity of a track, and thus scoring mechanisms must have some form of weighting or un-biasing function to reduce this influence.

This form of normalising track popularity assumes that all tracks have the potential to be good recommendations, thus optimising user satisfaction rather than facets like revenue. For recommender systems that aim only to increase revenue by recommending potential purchases, this normalisation may be undesirable. In this scenario, tracks which are becoming popular, or which are the latest trending sales should indeed have a positive bias. However, the focus of the work in this thesis is to increase user satisfaction and improve discovery, and therefore popularity normalisation is desirable.

4.1.3 Granularity

Ground truth data should be able to reflect user feedback at a similar level of granularity. This means that data such as genre, release year, and artist information should not be used for evaluation. These types of data lead to class-based evaluation strategies, which can be too coarsely grained. A dataset may contain many thousand tracks, but can be split into only tens of categories. If a query and recommended track share the same

category, the ground truth data provides a score of either true (1) or false (0). Such coarse grained scoring does not allow the distinction between how good two different good recommendations are, or how bad two different bad recommendations are.

4.1.4 Indicative Heuristic

The goal of a good ground truth scoring is *not* to accurately predict the exact ratings that a user will provide: accurate prediction is the goal of a recommender system. The values presented by a ground truth scoring do not even need to match the format or range of values that are provided by users through explicit feedback. Instead, the ground truth score must provide a heuristic measurement that indicates *how* users will respond to recommendations, not *what* they respond with.

Given two recommender systems, A and B, a ground truth score must be able to ascertain if users will typically rate system A or system B as the stronger. Further, this score must be able to reflect to what extent users will rate one system over the other. To achieve this, the ground truth score does not have to reflect every user score accurately, but it must reflect the same correlation.

4.1.5 Properties in Existing Ground Truth

Music recommendation and similarity may be evaluated using several ground truth scoring functions. To further illustrate and contextualise how these properties effect scoring, some example scoring methods are discussed. The first of these scores is 'genre score', which is defined as

$$\text{genreScore}(t_i, t_j) = \begin{cases} 1 & \text{if } t_i.genre = t_j.genre \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

where $t_i.genre$ is the genre assigned to track t_i. If both tracks share the same genre, then the recommendation is said to be good, and otherwise it is said to be bad.

genreScore completely satisfies the requirements of completeness and normalised track popularity, with a single genre label being available for every track. The score cannot be said to be substantial however, since genre is defined by the creators of the meta-data, who are typically either the artist or record label. Evidence suggests that people only

agree on what genre label should be used between 20 and 70% of the time (Gjerdingen & Perrott 2008), and so genreScore is too biased towards an individual's perspective.

A further possible ground truth score is yearScore, which assumes that good recommendations belong within a fixed time period, for example music from the 80s. This measure is calculated as

$$\text{yearScore}(t_i, t_j) = \frac{|\text{year}(t_i) - \text{year}(t_j)|}{\text{year}(t_y) - \text{year}(t_x)} \tag{4.2}$$

where $\text{year}(t_i)$ is the release year of track t_i, t_y is the most recently released track in the collection, and t_x is the most distantly released track.

yearScore also satisfies the requirements of completeness and normalised track popularity. The data is as substantial as could be possible, but it is questionable how accurately it describes user preference. For example, if a user likes music from the 80s, then a track from the late 70s and early 90s could obtain the same score, but it may be more likely users with this preference prefer 90s music over 70s music. This type of information cannot be gathered without user provided information, something which yearScore does not capture at all.

The main drawbacks of relying on either genreScore of yearScore as ground truth, is that neither take into account variability amongst users, and include these preference trends. This lack of user information introduces poor granularity within the scores obtained, which can make it difficult to separate to recommendations of similar quality. A ground truth score that does take specific user information into account is the peer-to-peer cultural similarity developed by Ellis et al. (2002). Although originally developed for artist similarity, the measure can be easily adapted to define track similarity. This ground truth score is calculated as

$$\text{culturalSim}(t_i, t_j) = \frac{\text{likes}(t_i, t_j)}{\min\left[\text{likes}(t_i), \text{likes}(t_j)\right]} \cdot \left(1 - \frac{|\text{likes}(t_i) - \text{likes}(t_j)|}{\max\limits_{t \in T}\left[\text{likes}(t)\right]}\right) \tag{4.3}$$

where T is the set of all possible tracks. The first term estimates user feedback as the proportion of users who have liked both tracks, given they have liked the least popular

track in the pair. The second term acts as a 'popularity cost', which weakens the first term when one track is very popular and the other is very rare.

If enough user profiles are examined, then culturalSim can be said to be substantial, and will naturally introduce a high degree of granularity. This allows the quality of recommendations to be examined in more detail, and avoid grouping recommendations when there is evidence that one is better than the other. Unlike the previous scores however, there is a risk that this score may not be complete, since calculating the score relies on enough users having liked each track. With careful data collection though this problem may be managed. A weak point of culturalSim is how popularity cost is defined, which penalises recommending a cold-start track from a popular track. This means that a popularity bias exists within the score, and this bias specifically discourages cold-start recommendation.

Each of the scores examined are typical of ground truth scores, and the problems highlighted are most definitely an issue when evaluating cold-start recommendation. Existing methods are either based on unsubstantial information, and thus have a poor granularity, or are based on substantial information, but at the cost of introducing a popularity bias. A new approach is required which is able to include the strong points of each existing approach, while minimising the weaknesses.

4.2 Evaluation Data

A ground truth score can only be developed with user data that is available, and thus the first step is to identify and acquire such data. The Last.FM service is used by millions of users, and their interactions with the service are made available by the Last.FM API. For each track, the total number of listeners and play counts are available, thus providing an indication of how popular the track is. For each user, the last 50 tracks which they have 'loved' are available, thus providing data upon which to base a ground truth score.

To collect data, the Last.FM API also provides some helper methods. The top fans of each track can be collected, providing a good start to ensuring each track has representative data. These top fans are also important because there is a strong association between them and the track, and thus their opinions are meaningful. To expand the set of users, the

API provides a method to list any user's friends, who are likely to follow a similar taste in music. However, after several iterations of listing users friends, the set becomes more diverse, reflecting a strong general opinion.

A total of 175,000 users were identified by gathering each track's top fans, and then iteratively gathering each fan's friends. Although more information is available, only the user name of each user is stored, since this is all that is required to collect their loved tracks. Having obtained a list of users, the next step is to start gathering the tracks that they love. The API imposes a rate limit of 5 requests per second, and thus it takes approximately 9 hours and 45 minutes to achieve a full pass over all users. This pass was run once per day for a period of 2 months. At each pass, only the last 50 tracks each user has 'loved' can be obtained. For some users, these 50 tracks never changed, and for others these 50 would change every day, and thus repeating this process over time is a crucial step to extracting substantial data. Further, Last.FM contains considerably more tracks than the private dataset, and so the majority of tracks returned did not appear in the private collection.

After the 2 month period, data was available for every track in the collection being used, and for every user that had been identified. On average, a single user had liked 5.4 tracks in the collection. This crawl provides a wealth of user-track information that can now be used to define a meaningful ground truth score.

4.3 Association Score

To define a ground truth score that satisfies the conditions described in §4.1, inspiration is taken from a hypothetical conversation between two people, P1 and P2.

P1: I've been listening to this awesome band called Klaxons.
P2: Me too, Do you like Foals?
P1: Never heard of them!
P2: You should get their CD, you'd like them.

In this conversation, P1 first states a band that both have been listening to, and that

they like. This band, Klaxons, are analogous to the query in a query-by-track recommender system. P2 then states that she also like this band, thus forming a common link. Based upon this established link, P2 then asks P1 if he also likes the band Foals. This question is analogous to a recommendation in a recommender system, which P1 may or may not know. P1 replies by stating that he has never heard of them, to which P2 responds by reaffirming her recommendation.

This conversation characterises the recommendation process, and if P1 likes the recommended track then the recommendation was good. If P1 does not like the recommended track, then it is not good. However, from an evaluation perspective, if P1 does not like the recommended track, then this recommendation should be considered of average quality, since one out of the two people like both the query and recommendation.

The example illustrates a conversation between two people, and this idea may be extrapolated to take into account the opinions of n people. To achieve this extrapolation, it must be possible to first estimate how likely the conversation is to take place, and compare this to how often the conversation actually took place. While the ground truth data collected does not reveal these conversations, it does reveal what the answers are likely to be for a large number of people.

The strength of an *association* between tracks t_i and t_j can be defined as the proportion of people who agree that there is an association, calculated as

$$\text{association}(t_i, t_j) = \frac{\text{likes}(t_i, t_j)}{\text{listeners}(t_i, t_j)} \tag{4.4}$$

where $\text{listeners}(t_i, t_j)$ is the number of people who have listened to both t_i and t_j, and $\text{likes}(t_i, t_j)$ is the number of listeners who have liked both t_i and t_j. This measure is similar to a user-centric evaluation, where a set of people are asked to listen to a query and recommendation, and then say if they like the recommendation or not. When the responses of all users are averaged, the association score is effectively determined.

The data collected from Last.FM, described in §4.2 can be used to obtain pairs of tracks that each user has liked. Unfortunately the data does not describe pairs of tracks to which each user has listened. However, the number of users who have listened to each individual track is known, and this allows an estimate of $\text{listeners}(t_i, t_j)$ to be determined.

$$\text{listeners}(t_i, t_j) = \frac{C(t_i)}{|\text{Last.FM}|} \cdot \frac{C(t_j)}{|\text{Last.FM}|} \cdot |\text{Collection}| \tag{4.5}$$

where $C(t_i)$ denotes the count of users who have listened to track t_i, $|Last.FM|$ denotes the number of users of Last.FM, and $|Collection|$ denotes the number of users in the evaluation dataset. To calculate this estimate it must first be assumed that tracks are listened to independently, listening to track t_j is not influenced by whether or not track t_i is listened to, and the estimate must be scaled to the number of users in the evaluation dataset. Using this equation, it is possible to calculate the level of association between tracks as described in Eq. (4.4).

One of the desirable properties of ground truth is that popularity is normalised. Association score achieves this by comparing the number of listeners of a track pair to the number of people who like the pair. If both tracks in the pair are unpopular the estimated number of listeners will be low, and thus only a small number of likes are required to achieve a high score. If one track is popular and the other is not, then a moderate number of listeners will be estimated. In this case, a moderate number of people who like both tracks are required to achieve a high association score. Finally, if both tracks are popular, then a large number of people are estimated to have listened to each track. Thus, a large number of people must also have liked both tracks for a strong association score to be achieved.

The association score estimates recommendation quality in the range 0 to 1. A value of 0 indicates that there is no evidence of users liking the pair of tracks. A value of 1 occurs when the number of users who like a track pair is equal to the number of users that have listened to the track pair. Due to listeners(t_i, t_j) being estimated, it is also possible that a score of greater than 1 is obtained for some track pairs. The reason for this is that as the true association between a track pair increases, the number of listeners is more greatly under-estimated, due to the assumption of independence. Such track pairs should intuitively be dealt with as being very associated, but allowing scores of greater than 1 can produce outliers that greatly influence averaged results. Thus, to handle this problem association score is limited to the minimum of 1 and he calculated score.

A score of 0 is achieved by 87% of all possible track pairs, and a score of 1 is achieved

for 3%. Figure 4.1 shows the distribution of association score values strictly between 0 and 1 that were calculated for the tracks in the private dataset. The horizontal axis shows the association score that was calculated, and the vertical axis shows the percentage of track pairs in the collection that obtain this score. A score of between 0 and 1 is achieved by 10% of possible track pairs. Of these pairs, the distribution centres around the score 0.12, which occurs for 0.25% of all possible query recommendation pairs. There are a total of approximately 5 million pairings, and the mean level of association score is 0.0064.

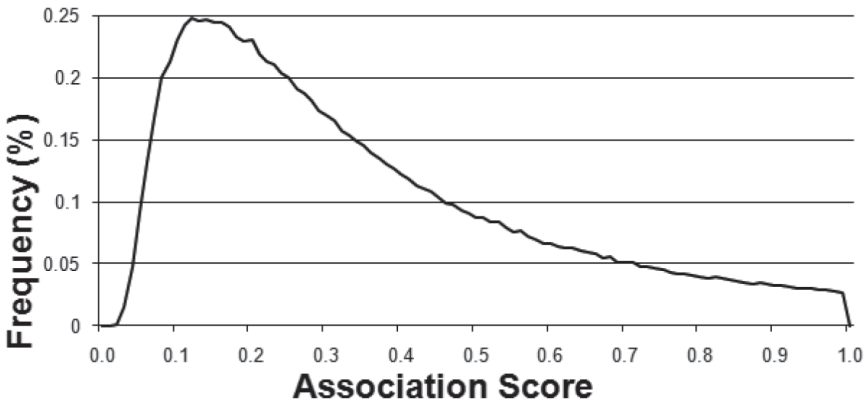


Figure 4.1: Distribution of association score values

This distribution of values shows that association score discriminates between high and low quality recommendations suitably. This score is much more fine grained than class based approaches, and is able to assign more meaningful scores. During evaluation, association score may be calculated for each query recommendation pair, and the average association score of the system calculated. Due to the similarities described between association score and a user evaluation, averaging should have the effect of indicating the results of a user evaluation, and thus user opinions.

4.4 Summary

The ability to evaluate recommender systems without the need for constant user interaction is critical to the development of new methods. User time is valuable, and so it is not feasible to ask users to repeatedly evaluate the smallest changes to a recommendation algorithm. However, the data to conduct such offline evaluations for music recommends

systems is not readily available. Progress has been made providing publicly available datasets, but these do not contain the audio files for tracks, and instead come with pre-extracted representations. For tasks where new audio representations are to be defined this introduces a barrier, and such datasets may not be used.

The important properties of system-centric evaluation ground truth have been identified and discussed. These include data being substantial and complete, and scores used being fine grained, indicative of how users perceive quality, and not biased towards popularity. An analysis of three ground truth scores that summarise existing techniques show that existing scores do not satisfy all of the properties that are required. This means that these techniques are only able to measure some aspects of recommendation, such as similarity, but do not necessarily indicate user preference.

A new approach to defining ground truth similarity has been defined, named association score, that takes advantage of data which is ublicly available through the Last.FM API. A pool of users is collected, and the tracks which they like are logged over a period of 2 months. Association score was developed to take advantage of this data, and satisfy the desirable properties of a ground truth score that were identified.

Association score opens up the opportunity to investigate new approaches to recommendation that use content-based representations. Such approaches would have previously been infeasible without regular user studies, since the other ground truth approaches do not accurately indicate recommendation quality. The new association score will be used as the standard evaluation measure throughout the work presented in each chapter. A user study will then be presented, which evaluates the work presented, and also the quality of association score as an indicative heuristic.

Chapter 5

Music-Inspired Texture Representation

A fundamental component of many music recommender systems is content-based features and representations. For many systems the audio for each music track is available, and encodes a wealth of information that describes the music. Content-based representations are typically used as part of a hybrid representation or system. Such hybrid representations may also include representations such as tags, which are not guaranteed to be available for every track. This lack of representation creates a cold-start problem, which content may be used to reduce. However, in these systems the content representations are usually the weakest component, and can potentially reduce recommendation quality. This presents a need to improve the content representations used in hybrid systems, reducing the performance gap between content-based and semantic representations.

The audio of any given track can be examined in time-slices, and for each time-slice a meaningful content-based feature can be extracted. Given enough content-based features, a representation can be learned that captures the important facets describing the track as a whole. Content-based features range from those that describe the objective structure of music, to features that describe the subjective feel of music, illustrated in Figure 5.1. Objective features typically describe rhythm, harmony, or melody - the primary building blocks of music. Extracting such features from polyphonic audio is an active research topic,

and much work is still required to extract the features robustly (Benetos & Dixon 2013). Using such features for the task of music recommendation can be found with varying degrees of success, but subjective features have proven to be more reliable and of higher quality (Casey et al. 2008).

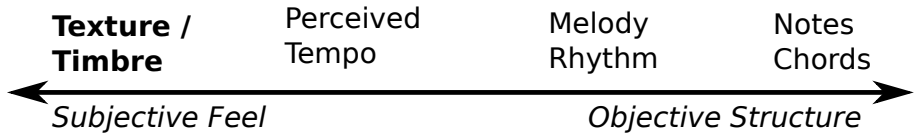


Figure 5.1: Musical Properties of Sound

The most widely used subjective feature is texture, which may be defined as how humans perceive the interaction of frequencies produced by a sound. Simple frequency interactions are perceived as pure, and complex frequency interactions are perceived as harsh. For example, a flute has very few frequency interactions compared to a distorted guitar, and so even when both instruments play the same note at the same volume, it is still possible to easily identify them. This effect is not specific to music, and texture can be used to distinguish between emotions such as happy and angry in speech sounds (Krothapalli & Koolagudi 2013), and even emotional responses in animals (Hauser 2000).

To represent texture, the Mel-Frequency-Cepstral-Coefficients (MFCC) feature is most commonly used. Although originally developed for speech recognition (Mermelstein 1976), MFCCs have proven to be robust across many musical tasks, including recommendation. In work comparing common content-based features, Pohle et al. (2005) find that texture outperforms other features in six out of the seven experiments they conduct, and conclude that improving texture could improve accuracy in their tasks. However, the foundation of MFCCs in speech processing can make texture difficult to understand in a musical sense, and thus difficult to improve.

In this chapter the applicability of MFCCs to music recommendation is questioned. Through structured analysis and evaluation of the algorithm, several of its properties are shown to be undesirable for music. These properties include the resolution used at a key step in the algorithm, and how use of the discrete cosine transform can be harmful. Based on this analysis of the MFCC algorithm, a novel music-inspired texture representation is developed, which is able to offer many improvements for music over MFCC.

5.1 Mel-Frequency Cepstral Coefficients

Mel-Frequency Cepstral Coefficients describe the texture of audio by applying transforms to its frequency spectrum. These transforms are designed firstly to account for the human perception of sound, as originally described by Stevens et al. (1937). After this, transforms designed to account for the physical properties of human speech are applied, specifically, in how noise is dealt with. In this section the MFCC algorithm is presented, and its properties analysed from a musical perspective.

MFCCs are generated for an audio track of any given length, illustrated by the flowchart in Figure 5.2. The audio track is encoded as a wave signal, typically at the CD quality sampling rate of 44.1kHz. The first step of the algorithm is to split the audio into fixed-length frames. Each audio frame is then re-expressed in the frequency domain by the Fast Fourier Transform, for which windowing is a pre-processing step. The Mel Filter Bank re-expresses the frequency spectrum along a psychologically based scale, producing the mel frequency cepstrum, which is then amplitude-normalised using a log scale. Finally, this cepstrum is summarised by a Discrete Cosine Transform, the coefficients of which are the MFCC vector.

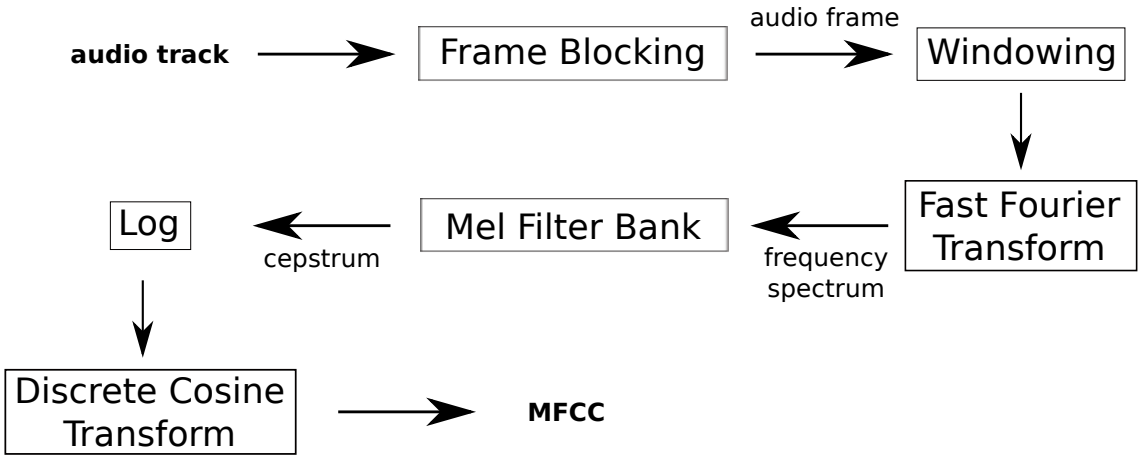


Figure 5.2: The MFCC Algorithm

5.1.1 Frame Blocking and Windowing

The input audio track may be up to several minutes long, and so must be split into shorter time frames before features are extracted. Efficient implementations of the Fast Fourier

Transform (FFT) expect an input length that is a power of two, which for CD quality audio is generally in the range 2^{10} to 2^{17} (23ms to 3s).

The next step after frame blocking in Figure 5.2 is windowing, which is a pre-processing step to the FFT. The waveform that will be converted to the frequency domain has been blocked into frames at arbitrary time points, and as such the blocked waveforms may contain discontinuities at their time boundaries. The effect of these discontinuities is that they introduce spectral leakage during the FFT.

Windowing is the process of multiplying the waveform x by a window function w. The window function tapers the boundaries of x, thus reducing spectral leakage during the FFT. A commonly used window function is the Hamming window, calculated as

$$w_n = \alpha - \beta \cos\left(\frac{2\pi n}{N-1}\right) \tag{5.1}$$

where $\alpha = 0.54$ and $\beta = 0.46$. N is the window size, identical to the length of frame x, and n is a pointer to an individual bin within the window function.

5.1.2 Fast Fourier Transform

After the signal has been pre-processed and split into frames, the next stage is to convert the time-domain signal into the frequency domain; this is achieved by the FFT algorithm. From our time-domain waveform x, the frequency spectrum X is calculated as

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi \frac{k}{N} n} \tag{5.2}$$

where each X_k is a complex number encoding an amplitude and phase component of x, and $\frac{k}{N}$ is frequency in cycles per sample. Frequency can be re-expressed in cycles per second as

$$f = \frac{k}{NT} \tag{5.3}$$

where T is the length of each frame in seconds.

To create the MFCC texture representation, only the amplitude, not phase, of each frequency component is important. This is calculated as

$$|X_k| = \sqrt{\mathrm{Re}(X_k)^2 + \mathrm{Im}(X_k)^2} \tag{5.4}$$

This frequency power spectrum obtained using the FFT describes the physical properties of x.

5.1.3 Mel Filter Bank and Logarithmic Normalisation

To construct the MFCC texture representation this output must be warped into a perceptual scaling, both of frequency and power. The first stage of warping the physical output of the FFT to a perceptual one is to apply a mel filter bank. This filter bank is constructed using the mel scale (Stevens et al. 1937), which describes how humans perceive differences in frequency. The conversion from frequency f to mel ϕ is calculated as

$$\phi(f) = 2595 \log_{10}\left(\frac{f}{700} + 1\right) \tag{5.5}$$

and is illustrated by the curve in Figure 5.3. The horizontal axis is f, and the vertical axis is $\phi(f)$.

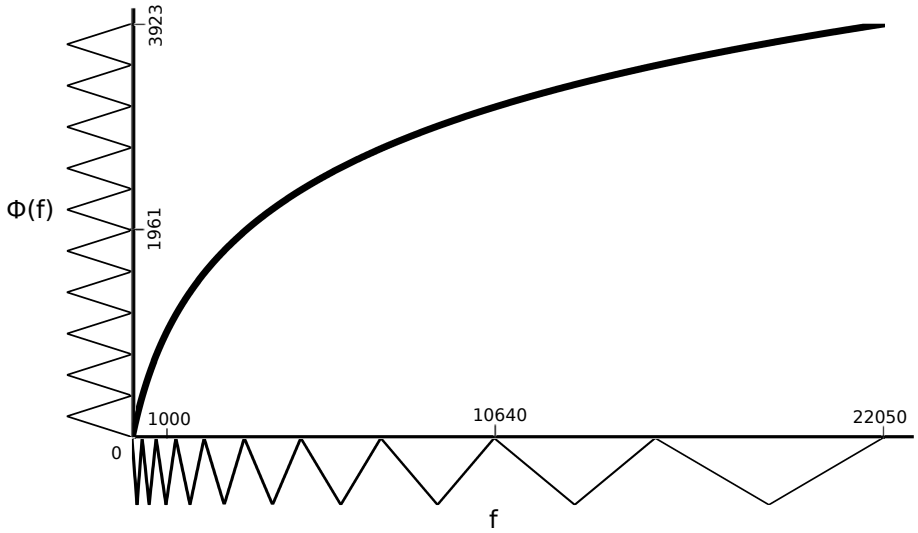


Figure 5.3: Frequency - Mel Curve

The curve shown is approximately linear to 1000Hz, and reflects how frequency is perceived. Humans find it relatively easy to recognise changes at low frequencies, for example between 400Hz and 440 Hz. However, at higher frequencies this becomes increasingly more

difficult, and a larger change in frequency is required before the change is noticeable.

The filter bank used to warp frequency to mel is illustrated by the triangular filters in Figure 5.3. These are spaced equally along the $\phi(f)$ axis, thus producing the spacing along the f axis as illustrated. Formally, for M filters, the width $\Delta\phi$ of each filter is

$$\Delta\phi = \frac{\phi(f_{max})}{M} \tag{5.6}$$

From eq. (5.3), the lower bound l and upper bound u frequency spectrum bins k, which correlate to the upper and lower bound frequencies of filter m, are

$$l_m = \phi^{-1}(m\Delta\phi)NT \tag{5.7}$$

$$u_m = \phi^{-1}((m+1)\Delta\phi)NT \tag{5.8}$$

and the length of filter m in terms of frequency spectrum bins k is

$$\Delta k_m = u_m - l_m \tag{5.9}$$

The mel filter bank matrix F is then constructed as

$$F_{km} = \begin{cases} \frac{2}{\Delta k_m+1} \cdot \left(\frac{\Delta k_m+1}{2} - \left|(k-l_m) - \frac{\Delta k_m-1}{2}\right|\right) & \text{if } l_m \leq k \leq u_m \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

where each column contains a single triangular mel filter. This filter bank is applied to the power spectrum X as

$$Y = XF \tag{5.11}$$

where Y is the frequency spectrum X discretised into M buckets by the filter bank F.

The second stage of perceptual warping is to normalise the values of Y by taking their logarithm. The reason for this is to roughly approximate the way that humans perceive perceive loudness. After this normalisation step, the output obtained is referred to as the **mel-frequency cepstrum**.

5.1.4 Discrete Cosine Transform

The final step in Figure 5.2 is the Discrete Cosine Transform (DCT), which takes the mel-frequency cepstrum as an input, and outputs coefficients that describes its shape. These mel-frequency cepstral coefficients are the MFCC texture representation.

The description provided by the DCT is a set of weights, each correlating to a specific cosine curve. When these curves are combined as a weighted sum, the mel-frequency cepstrum is approximated. To calculate the mel-frequency cepstral coefficients y, the DCT is applied to the mel-frequency cepstrum Y as follows

$$y_n = \sum_{m=0}^{M-1} Y_m \cos\left[\frac{\pi}{M}\left(m + \frac{1}{2}\right)n\right] \tag{5.12}$$

for $n = 0$ to $N - 1$, where $N < M$.

5.2 Music Inspired Texture Representation

Resolution plays a key role in the MFCC algorithm; resolution of FFT affects the mel filter bank, and resolution of the mel filter bank affects the output cepstrum, and its coefficients. At each stage in the algorithm, the resolution chosen must be reasoned and meaningful. If too low a resolution is used, then perhaps important information is lost, and if too high a resolution is used, there is a risk of introducing noise.

5.2.1 Important Properties of Speech

To fully understand the effects of resolution, we must start with frame blocking and the FFT. The length of spectrum X is half the length of frame x. For example, if each frame is blocked to have length 2^{10}, then X contains 2^9 frequency bins. This becomes important when the sampling frequency is taken into account; the maximum frequency captured is half the sampling rate (Nyquist 1928). For CD quality audio, the sampling rate is 44.1kHz, and thus the maximum frequency represented in the spectrum X is 22.05kHz. In this example therefore, the resolution of each X_k is

$$\text{FFT bin resolution} = \frac{\text{Max Frequency}}{\text{Number of bins}} = \frac{22050}{2^9} = 43.066\text{Hz} \tag{5.13}$$

To increase the resolution of X, one must increase the number of bins, and thus the window size. However, Pachet & Aucouturier (2004) found that MFCC features are only useful on stationary frames, and observe negative effects as the frame size reaches 1 second. Frames larger than around 100ms will include many transients and variations, introducing noise into the texture representation. This initial resolution therefore cannot be changed significantly, but must form the basis of further analysis of resolution in the MFCC algorithm.

5.2.2 Physical Resolution of Mel Filter Bank

Perceptual frequency warping is not achieved as a direct conversion from f to $\phi(f)$, but as a discretization of f based on $\phi(f)$. This discretisation is achieved by the mel filter bank defined in eq. (5.10), and applied using eq. (5.11). To avoid confusion, each X_k in the frequency spectrum will be referred to as a bin, and each triangular filter m of the mel filter bank as a bucket.

M buckets are equally sized and spaced along $\phi(f)$, as previously described in §5.1.3. The buckets on the f axis are equivalent to those on $\phi(f)$. The value of each bucket is a weighted sum of all its frequency bins, where the weights are defined by the triangular filter.

Typically when the resolution of MFCC is discussed, it is in terms of frequency and mel. The range of frequencies that are discretised into each bucket is defined by both the resolution of FFT, and the number of filters used, M. From the previous example, X contains 2^9 frequency bins, each 43.066Hz wide, and M mel filters of varying width are used. If 10 mel-filters are used in the filter bank, each bucket is 392 mels wide. Figure 5.4 illustrates how the width of each mel filter varies in terms of frequency.

The smallest filter 0 in Figure 5.4 is 291Hz, and the largest 9 is 6688Hz. As a percentage of the information available, each bucket covers 10% of the mel scale, or between 1.3% and 30% of the frequency scale. At a first glance it seems that the smallest bucket is too fine, and the largest bucket is too coarse. Although the mel scale is used to replicate human hearing, grouping 30% of information into a single bucket does not seem intuitive. In terms of human hearing ranges the smallest and largest buckets are identical, however, it must be remembered that the largest bucket will also include harmonics of the smallest

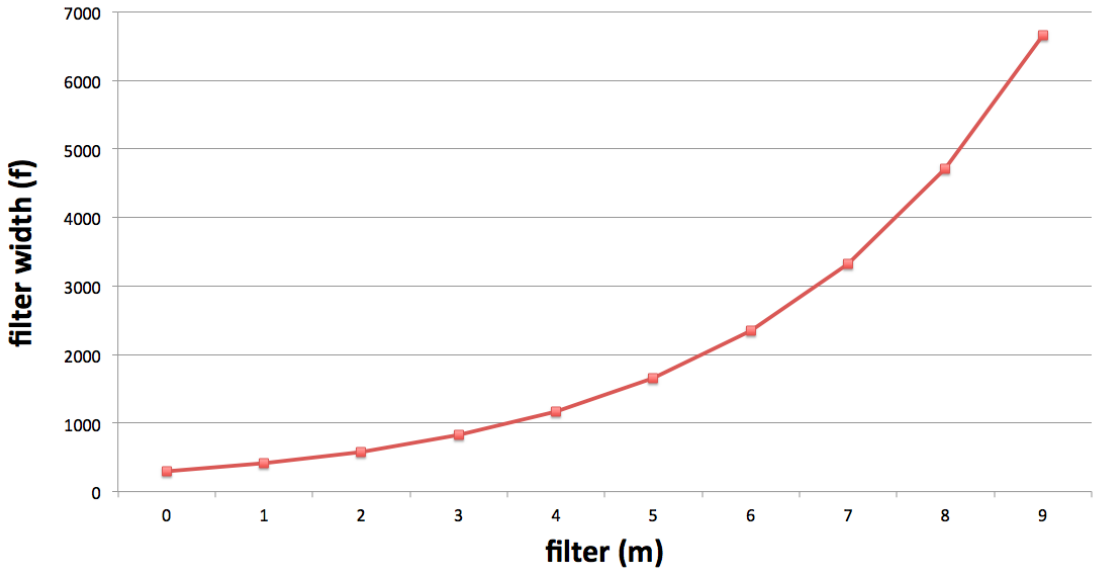


Figure 5.4: Mel Filter Widths

bucket. In speech-based studies it is argued that lower resolutions can be advantageous (Skowronski & Harris 2002); but due to harmonics this may not be the case for music.

5.2.3 Musical Resolution of Mel Filter Bank

To understand the effects that the mel filter bank has for music, resolution must be examined using a musical scale. The mathematical scale upon which all western musical scales are constructed is the cents scale, known to musicians as the chromatic scale. This cents scale describes the difference between two frequencies, where 100cents is one semi-tone, and is defined as

$$c(f, f_{\text{ref}}) = 1200 \log_2 \left(\frac{f}{f_{\text{ref}}} \right) \tag{5.14}$$

where f is a variable frequency, and f_{ref} is a reference frequency. When tuning musical instruments, 440Hz is a common value for f_{ref}, and corresponds to the musical note A_4. To avoid confusion from negative values in our example, $f_{\text{ref}} = 16.35$Hz will be used, which corresponds to the musical note C_0. The lowest note on a standard piano is A_0 at 27.5Hz.

Figure 5.5 shows the filter widths in Hz identically to Figure 5.4, and also includes the filter widths in terms of musical notes. The curve is in terms of the left vertical axis

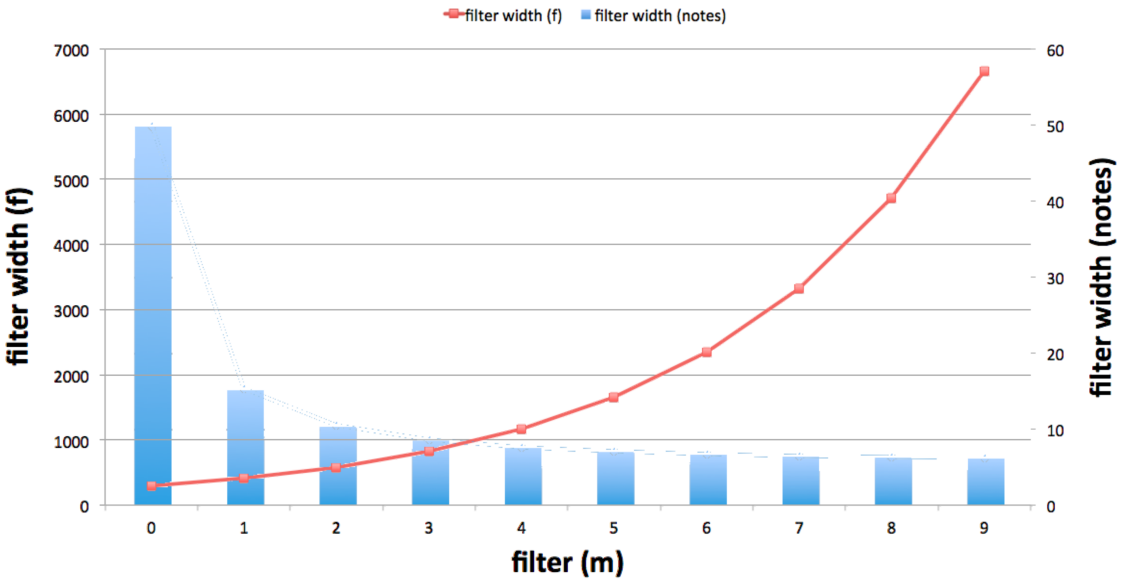


Figure 5.5: Mel Filter Widths Examined Musically

showing frequency, and the bars are in terms of the right vertical axis showing notes. This shows a completely different picture of resolution when it is examined in terms of musical notes. Unlike frequency, the 0^{th} filter now discretises the most information, nearly 50 notes. The 9^{th} filter is the largest in terms of frequency, but smallest musically, discretising just a fraction over 6 notes. There are 12477 cents between our reference frequency f_{ref} and 22050Hz, which is just over 124 musical notes. Re-examining the discretisation as a percentage, the smallest bucket covers 39.5% of all musical notes, and the largest bucket covers 0.048%.

The 0^{th} filter in Figure 5.5 discretises much more notes than any other filter. The reason for this actually agrees with the underlying principles of the mel scale; people can tell apart low frequencies more easily than high frequencies, and thus there must be more notes at the lower frequencies. This makes the insight that low-frequency filters discretise a large number of notes easy to miss if the cents scale is not considered.

Implementations of MFCC typically set $M = 40$, causing the 0^{th} filter to cover 23.5 musical notes, and the 39^{th} filter to cover only 1 musical note. Discretisation of 23.5 musical notes still appears to be too much however; most musical notes are played at relatively low frequencies. Higher frequencies in the spectrum often capture harmonics of the lower notes being played, and so their discretisation is not as critical.

A strong music-inspired texture representation should be able to capture all of the musically meaningful information, while still adhering to the psycho-acoustic principles of the mel-scale. In the developed musical texture, M must be a carefully considered parameter, and may require to be set significantly higher than 40. Increasing M will decrease the size of all buckets, until such a point that reasonable musical discretisation is achieved. For higher order buckets, the number of notes they discretise will fall below 1. This should not be considered a problem however, it is a trade off of the balancing act required to capture the correct musical information.

5.2.4 How Noise is Handled

Once the mel cepstrum has been obtained, it is then summarised by the discrete cosine transform (DCT). The motivation behind this is two-fold: to concisely describe vocal formants, and to remove the effect of general noise. In speech, meaningful information is described by formants, irrespective of which pitch they were uttered at. For example, in a speech transcription task it should not matter if a low male, or high female voice is the input.

DCT summarises the cepstrum as a weighted sum of cosine curves that follow a fixed pattern of wavelength: the 0^{th}, 1^{st}, 2^{nd} and 3^{rd} curves have wavelength $\frac{0}{M}$, $\frac{0.5}{M}$, $\frac{1}{M}$, and $\frac{1.5}{M}$ respectively (see eq. (5.12)). Thus it is the weights of each curve that make the parameters of y, and y_0 describes the DC level of the cepstrum. Since for speech formants are best described by low frequency changes in the cepstrum, only a few number of values are retained after the DCT, typically around 13 coefficients.

Pitch is important for music, and so one may be temped to assert that more coefficients should be retained from the DCT, the higher frequency changes. This would increase the resolution of the DCT, but in a non-meaningful way; as the number of coefficients in y tends to the number of coefficients in Y, the two vectors describe the exact same information, but in a different format. Increasing the number of coefficients retained by the DCT therefore will not be helpful for a music texture representation.

The main reason for using DCT is speech texture is to capture formants while removing noise; these concepts do not translate into the music domain, where noise, consisting primarily of breath does not exist. In music, the equivalent of background “noise” is in

fact produced by instruments that ring continuously, for example a cymbal or the drone of a bagpipe. Omitting such sounds from a music texture representation could indeed be harmful.

As an example, let us consider the genres of *dance* and *rock*. In dance music cymbals are often computer generated, and may be clipped to remove any sustain, producing a very tight on-the-beat sound. In rock music cymbals are recorded using an over-head microphone capturing a live musician. The EQ on this microphone channel is intentionally designed to let cymbals ring out until the next time the musician hits them. This distinction between tight clipped cymbals and looser sustained cymbals can be subtle, given that cymbals are not the most prominent instrument in a track. However, these subtle differences can be important when determining the similarity between textures, and thus must be captured.

For music-inspired texture, the DCT will not be used. Instead, all of the information provided in the cepstrum Y will be made available to the learning models that use it. If the appropriate models are used, then the important components of Y will be learned effectively. These models are important to the successful usage of Y, since all information is available. The music inspired texture as described will be referred to as the Mel-Frequency Spectrum **MFS**.

5.3 Evaluation of MFCC and MFS

To evaluate how effective the music-inspired texture MFS is compared to MFCC, both representations will be compared by their recommendation performance. The music collection used for evaluation is described in §3.2, and the evaluation data described in §4.2. The evaluation method used is the association score, as applied in §4.3.

The different models are used to create recommender systems using MFS and MFCC. These are standard models used in both speech-based and music-based tasks, and are described in the following section. To examine the effects of resolution the number of filters M is modified, using the values 40, 60, 80, 100, and 120. For 40 filters the 0^{th} filter contains just over 25 notes, and the 39^{th} just over 1. For 120 filters the 0^{th} filter contains 4 notes, and the 119^{th} only 0.5 notes.

5.3.1 Evaluated Models

Three different models are compared, allowing the performance of both MFS and MFCC to be measured under varying conditions. Latent Semantic Indexing (LSI) is used to evaluate generalising texture, k-means (VOC) is used to evaluate clustering texture, and Gaussian Mixture Models (GMM) are used to evaluate modelling the distribution of texture in a track. Each model will be referred to by the representation used to combine it, and the model itself, for example MFS-LSI, or MFCC-LSI.

Generalisation

For each track, a mean texture vector is first computed, where each dimension in this mean vector corresponds to the mean value across all of the track's texture vectors. The mean texture vectors of all tracks in the training set are then used to construct a track by mean vector matrix. This matrix is generalised using Singular Value Decomposition, thus obtaining basis functions for Latent Semantic Indexing. Given a query track, the mean vector is projected into this generalised search space, and recommendations are made using cosine similarity.

Clustering

To generate a vocabulary of cluster centroids the k-means algorithm is used. The input data for this model is 20000 randomly selected texture vectors from tracks in the training set. Using this vocabulary, the vectors for each track are assigned to their nearest cluster. The number of assignments for each track-cluster pair are used to construct a cluster-track matrix, and TF-IDF weighting is applied. Recommendations are made from this matrix using cosine similarity.

Distribution Modelling

A Gaussian Mixture Model (GMM) describes the distribution of a track's texture vectors as a weighted sum of K simple Gaussian distributions, each known as a component of the model. Each weighted component is described by its mean texture vector, and a covariance matrix (Aucouturier et al. 2005). To evaluate MFS and MFCC a GMM is learned for each

track using the Expectation Maximisation algorithm.

Recommendations are made using the Kullback-Leibler (KL) divergence between track GMMs. However, KL divergence only measures the divergence between two Gaussian distributions; each GMM has K Gaussian distributions. To overcome this problem recommendations are made based on an approximation of KL divergence (Hershey & Olsen 2007). This approximation method first computes the KL divergence between all pair-wise combinations of components of two GMMs. The final KL divergence is calculated as a weighted average of the minimum divergence from each query component. Unlike with the previous two models discussed, similarity in this case is asymmetrical, since divergence is calculated from each query component, but not necessarily to each recommendation component. This makes sense because importance must be given to the query: if this measure was forced to be symmetrical, then parts of a recommendation that are not important for a query would be allowed to influence recommendations.

5.3.2 Baseline Recommendation Quality

Figure 5.6 shows the recommendation quality achieved using MFS and MFCC for each model. The vertical axis shows the association score, and the horizontal axis shows the number of recommendations that were evaluated. Each value shown is the average association score at the number of recommendations made, and error bars show significance at a 95% confidence interval from the mean.

Recommendations using MFS-LSI achieve significantly better association scores than all other representation models. MFS-VOC provides a significant quality increase over MFCC-VOC when more than 1 recommendation is made. The reason that MFS performs better than MFCC for both the LSI and VOC models is because both models group data at the collection level, as opposed to the track level of GMM. LSI is finding associations between dimensions in the mean texture vectors, and VOC is grouping texture vectors into clusters. Crucially, MFS is allowing these models to group at the collection level effectively, unlike MFCC. The reason for this is that MFCC is summarising each texture vector at the track level by applying DCT.

When 40 filters are used, MFCC-GMM performs significantly better than MFS-GMM. The reason for this is that GMM behaves differently to LSI and VOC, and each track is

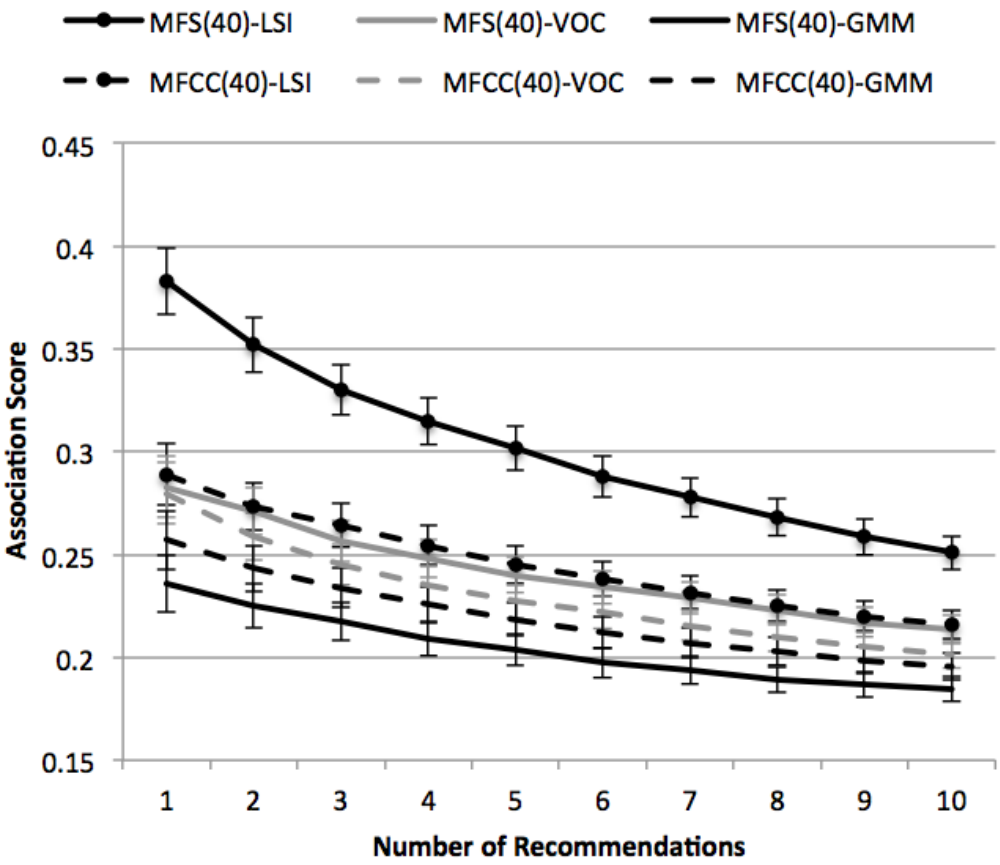


Figure 5.6: Comparison of MFS and MFCC using 40 filters

modelled in isolation. The implementation of EM used to learn the GMMs makes use of a diagonal covariance matrix to increase efficiency of the algorithm. The effect of this is that each dimension in the texture vectors are independent, and thus associations between dimensions are not learned in the same way that they are with LSI and VOC.

5.3.3 Effects of Resolution on Recommendation Quality

To explore the effects increasing the number of filters has on recommendation, a more simple recommendation task is examined. Figure 5.7 shows the average association score of the top 3 recommendations made for each model using MFS. The horizontal axis is grouped by the model used, and each bar corresponds to a given number of filters.

Increasing the number of filters significantly increases the recommendation quality of MFS-GMM, does not significantly affect MFS-VOC, and significantly decreases the quality

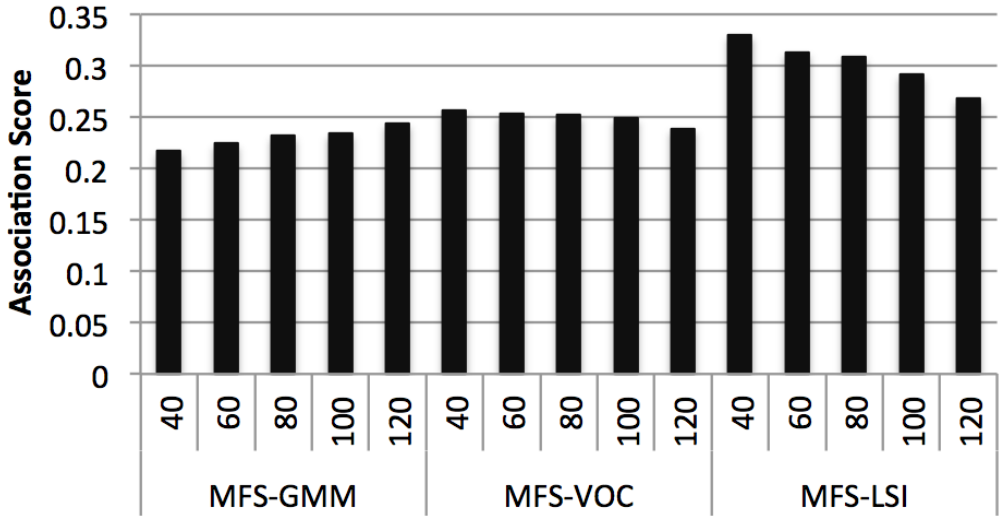


Figure 5.7: Effect of filters on MFS

of MFS-LSI. The reason that GMM is improved is because more independent dimensions are available to the model, and thus more meaningful distributions can be learned. The k-means based VOC model does not change, since similar clusters are learned. Although the search space to be clustered is expressed by more dimensions, the clusters formed in this space are roughly equivalent. The performance of LSI decreases as more filters are used, showing that this model can generalise a low number of filters more effectively.

Figure 5.8 shows the effect of changing the umber of filters for MFCC. In this case no correlations appear between the number of filters used and the association score. The MFCC representation is not as predictable and stable as the MFS representation when modelled. This makes the representation much more difficult to tune to particular applications, and gain the best performance from it.

5.3.4 Comparison of MFS and MFCC

Figure 5.9 is in the same format as Figure 5.6, and illustrates MFS and MFCC when the best performing number of filters are used for each model. VOC is not shown because neither representation was improved by modifying the number of filters used.

The solid black line shows MFS(40)-GMM, and the solid grey line shows MFS(120)-GMM. Both results are included to illustrate the improvement that increasing the number of filters achieves. In comparison, there is only a very small improvement for MFCC-

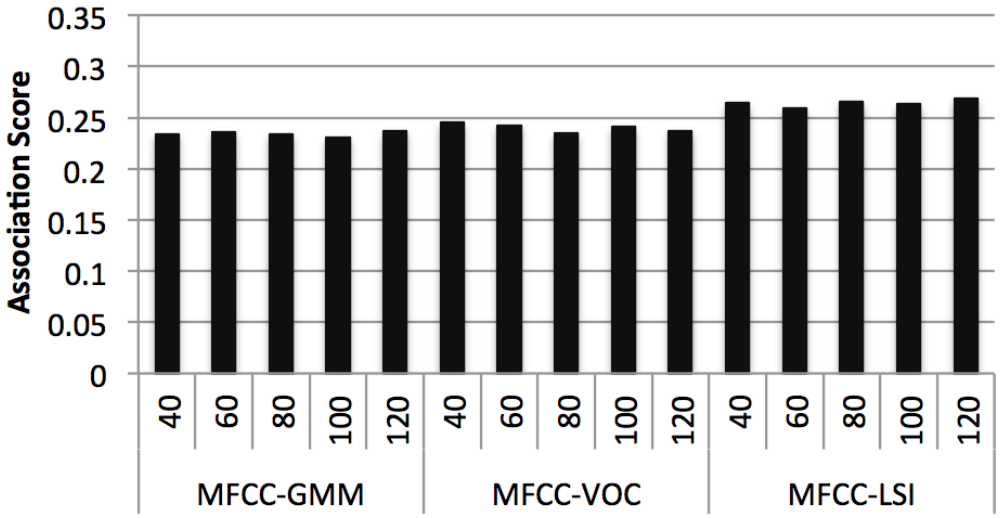


Figure 5.8: Effect of filters on MFCC

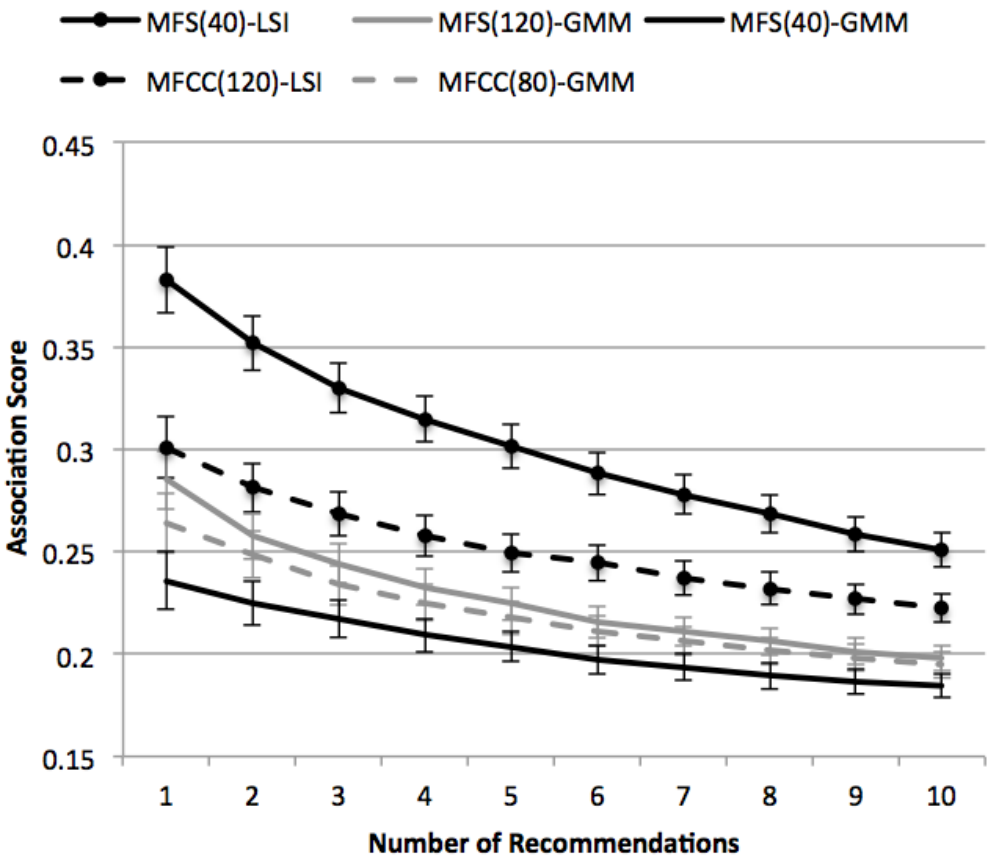


Figure 5.9: Comparison of best MFS and MFCC by model

GMM by increasing the number of filters. For the first 5 recommendations MFS-GMM now achieves significantly better recommendation quality than MFCC-GMM.

5.4 Summary

The entire mel-frequency spectrum is important when describing the texture of music, and should not be summarised by DCT. When the MFS representation is used, all of the texture information is available to each model, leading to improved recommendation quality over models using MFCC. When MFCC is used, DCT provides a summary of the information within MFS, which does not allow a recommender model to learn the important aspects. Experimental results show that by not using the DCT, MFS achieves significantly better recommendation quality for each of the three models evaluated.

Traditional speech texture representations are extracted with a fixed, standard number of filters, and therefore resolution. Results show however that to extract a meaningful music-inspired texture representation, it is important to understand how this resolution affects the model being used. This stable link between representation and model is important, and is something not achieved by MFCC. There is no single MFS resolution which is optimal for each recommender model evaluated, but by understanding how a model works, it is possible to extract the most meaningful MFS representation.

In each recommender model evaluated, the behaviour of MFS is more predicable than with MFCC. For GMM, more filters are best because the model is able to describe this increased amount of information meaningfully. With VOC there is no significant difference, and with LSI fewer filters can best be generalised. Results obtained using MFCC do not show any of these trends, and thus no insight can be gained into how this representation may be used to model music.

Although the LSI model clearly outperforms both VOC and GMM, both VOC and GMM are commonly found throughout hybrid recommender systems. The improved quality of MFS over MFCC in these models, and the understanding of resolution, may lead to improved hybrid recommender systems when VOC or GMM is used. In following chapters the link between music texture and semantic features will be explored in detail; a task made more achievable by the MFS representation.

Chapter 6

Combining Content-Based and Tag-Based Representations

Content-based representations can be used as replacement for tags when cold-start is present. The benefit in doing so is that the discovery of cold-start tracks is increased, thus allowing these tracks to be gradually incorporated into a tag-based recommender system. Recommendations made using content may or may not have a tag description, but including them in recommendations increases the likelihood that each track will become well-tagged. As these tracks become increasingly more tagged, tag-based methods can supersede content-based methods.

However, this description is an idealised one, and assumes that content-based recommendations are of similar quality to tag-based recommendations. In reality a tag-based recommender will significantly outperform even the very best content-based methods. Methods are required which increase discovery, but not at the expense of reduced recommendation quality.

Hybrid recommender systems offer an opportunity to increase cold-start discovery, while maintaining a consistent recommendation quality. The challenge of a hybrid representation is to incorporate the discovery power of content-based representations, with the recommendation quality power of tag-based representations, as illustrated in Figure 6.1.

In this chapter some important properties of both content- and tag-based representa-

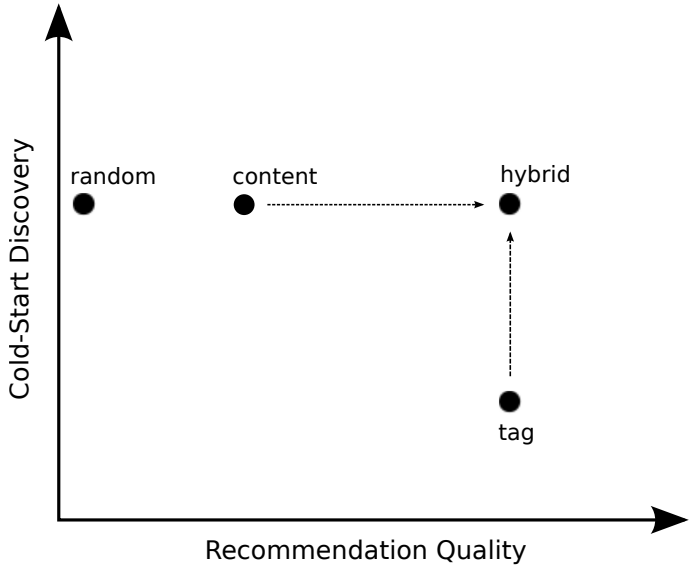


Figure 6.1: Balancing Quality and Discovery with Hybrid Representation

tions are explored. These properties have implications on how two fundamentally different representations may be combined into a hybrid representation. Based on this investigation a method of combining content and tags is described, and a new approach to learning the important features of this combination is developed. This hybrid representation is then evaluated both in terms of recommendation quality, and discovery rate of cold-start tracks. The work in this chapter was presented at the Twenty-Second International Joint Conference on Artificial Intelligence (Horsburgh, Craw, Massie & Boswell 2011)

6.1 Basic Representations

To successfully combine both content and tags in a unified representation, it is important to understand the characteristics of each individual representation. This includes examining how the representation is described, how similarity is calculated and what it means, and properties such as sparsity within tags.

6.1.1 Content Representation

Content-based representations are available for every track in a collection when a recommender systems has access to audio files. These representations are a good starting point for inclusion in a hybrid representation, since they always follow a well-defined structure,

and are always available.

Content representations describe physical properties of musical audio, such as chroma, rhythm, harmony, and texture. Although the focus of this work is on using texture, each representation adheres to a similar structure. Audio is sampled in fixed time frames, and features are extracted. For texture, as described in Chapter 5, the output is a single Euclidean vector for each time frame, which is then aggregated into a mean feature vector.

$$c = \begin{bmatrix} c_1 & c_2 & ... & c_n \end{bmatrix} \tag{6.1}$$

The content vector c will by definition contain no sparsity, that is, each dimension has a non-zero value. Each value in the vector describes something meaningful, which in theory could be interpreted by humans. In the case of MFS, each dimension describes the specific influence of a critical band within a track's audio. The value c_0 describes the influence of the lowest bass notes, and the value c_n describes the influence of the highest treble notes and overtones.

6.1.2 Tag Representation

Tag representations are created from the social tags applied to a track by users. Once aggregated across all users, each tag has a weight when applied to a track, and these weights can be represented as a Euclidean vector t.

$$t = \begin{bmatrix} t_1 & t_2 & ... & t_m \end{bmatrix} \tag{6.2}$$

The vector t must contain a single dimensions for every possible tag that could be applied, and thus the number of dimensions is equivalent to the vocabulary size for the collection of tracks. From this vocabulary, each track will only have been assigned a very limited number of tags, and thus many dimensions of t will be zero valued. Unlike with content vectors then, each track's tag vector t will contain a large degree of sparsity.

6.2 Concatenating Content and Tag vectors

In §2.4 the different approaches to reducing cold-start were discussed. Each approach aims to reduce the sparsity problem in tag representations, thus reducing the cold-start problem. However, existing approaches first cluster the content representation, and then combine this clustered representation with a tag representation. This approach assumes that each content vector as a whole is meaningful, and not the individual dimensions within the content vector. Results in the previous chapter show however that this is not the case, and that each individual dimension is meaningful.

With MFS, each dimension describes the properties of a perceptual critical band within the music. If these vectors are clustered, then relationships between the bands are captured, but they can no longer be identified. When tags are introduced therefore, it is impossible to realise if there is an explicit relationship between the tag and the critical band, since only relationships with clusters can be established. For example, it is possible that tracks with high values in the low dimensions of MFS will be tagged with words like 'bass', 'low-end', and 'booming'. In a clustered space, it is not guaranteed that all tracks with strong low-dimensional values will fall into the same clusters, and thus this relationship cannot be captured.

To combine content with tags in such a way that these relationships can be explicitly identified, the set of content vectors describing each track must first be aggregated. The reason for this, and indeed for clustering, is to ensure a normalised amount of information describes each track. That is, long track's with many content vectors are described in a comparable way to short track's with few content vectors. Each track is represented by a set of content vectors C, which are aggregated into a mean content vector $\bar{c}$ by calculating the average of each dimension,

$$\bar{c_i} = \sum_{c \in C} c_i \tag{6.3}$$

This mean content vector is then concatenated with the track's tag vector t, forming the combined vector s as

$$s = \begin{bmatrix} c_1 & \cdots & c_n & t_1 & \cdots & t_m \end{bmatrix} \tag{6.4}$$

where n is the number of dimensions in each track's content vector c, and m is the number of dimensions in each track's tag vector t.

This concatenated vector for each track now introduces the possibility of identifying meaningful correlations between content and tags. The identification of these correlations is important, since it does not make sense to compare the concatenated vectors directly. Sparsity still exists within the tag component of cold-start tracks, and comparing these concatenated vectors is equivalent to simply averaging the similarity scores of each representation individually. Without identifying correlations, the problem of sparsity cannot be reduced.

6.3 Generalisation by Latent Semantic Analysis

Latent semantic analysis (LSA) provides an effective technique that can be used to generalise the concatenated representation, and allow the relationships between content and tags to be uncovered and exploited. Each track is described by a single concatenated vector s, which is used to construct the track matrix S as

$$S = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1T} \\ \vdots & \vdots & \vdots & \vdots \\ S_{n1} & S_{n2} & \cdots & S_{nT} \\ S_{(n+1)1} & S_{(n+1)2} & \cdots & S_{(n+1)T} \\ \vdots & \vdots & \vdots & \vdots \\ S_{(n+m)1} & S_{(n+m)2} & \cdots & S_{(n+m)T} \end{bmatrix} \tag{6.5}$$

where T is the number of tracks in the collection, and each column represents the concatenated vector s for an individual track.

During the construction of this matrix, the content and tag components of each track's concatenated vector must each be normalised to a unit vector. This ensures that neither representation overpowers the other during generalisation, and that meaningful relationships can be identified.

6.3.1 Singular Value Decomposition

LSA is a way of interpreting the singular value decomposition (SVD) of a matrix. The first step of SVD is to calculate correlations within the input matrix s as

$$A = SS^T \tag{6.6}$$

$$B = S^T S \tag{6.7}$$

Matrix A contains all the first-order representation correlations with respect to tracks, and B contains all the first-order track correlations with respect to representations. For example, matrix A describes to what extent one dimension in the representation correlates with another dimension in the representation, across the collection of tracks. In the case of concatenated vectors s therefore, A will determine the relationships between track content vectors, track tag vectors, and the content and tag vectors of a track. This final correlation is the most important for reducing the effects of cold-start, since enforcing these correlations to under-tagged tracks vectors allows tag information to be propagated.

The next step of SVD is to determine the eigenvectors of A, such that

$$Au = \sigma u \tag{6.8}$$

where u is a single eigenvector, and σ is its corresponding eigenvalue. Similarity

$$Bv = \sigma v \tag{6.9}$$

where v is a single eigenvector, and σ is its corresponding eigenvalue. In both eq. (6.8) and eq.(6.9), the eigenvalues σ are identical, but the eigenvectors u and v will be different.

This step of calculating the eigenvectors and eigenvalues of the correlation matrices A and B is the key method behind the decomposition of matrix S. The eigenvectors u are arranged as column vectors, and form matrix U. The eigenvectors v are arranged as row vectors, and form matrix V^T. The eigenvalues σ are arranged as the diagonal of a square matrix, and form Σ. Thus the original matrix S has been decomposed into the

form $S = U\Sigma V^T$ based on its singular values.

$$S = \begin{bmatrix} \begin{bmatrix} u_0 \end{bmatrix} & \begin{bmatrix} u_1 \end{bmatrix} & \cdots & \begin{bmatrix} u_l \end{bmatrix} \end{bmatrix} \cdot \begin{bmatrix} \sigma_0 & \cdots & \cdots & 0 \\ \vdots & \sigma_1 & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & \sigma_l \end{bmatrix} \cdot \begin{bmatrix} \begin{bmatrix} v_0 \end{bmatrix} \\ \begin{bmatrix} v_1 \end{bmatrix} \\ \vdots \\ \begin{bmatrix} v_l \end{bmatrix} \end{bmatrix} \tag{6.10}$$

Importantly however, each eigenvector u characterises a set of representation-representation correlations, since its direction does not change when multiplied by the correlation matrix A (eq. (6.8)). Through calculating these eigenvectors, the individual pair-wise correlations are generalised across the all pair-wise correlations. The singular values σ each correspond to the strength of this generalisation across the dataset, allowing the most important generalisations to be identified.

6.3.2 From SVD to LSA

SVD is a mathematical tool to tri-factorise a given input matrix. LSA is a specific way of interpreting the results of this tri-factorisation, that can be used to interpret the generalised representation. Each characterising eigenvector is known as a concept, and these concepts are important to using SVD effectively to achieve LSA.

Removing Noise

In a generalised representation, it is desirable that any noise is removed from the original input representation. An interesting property of SVD that can aid this, is that any related u, σ, and v vectors can be safely removed from their respective matrices. The result of calculating $U\Sigma V^T$ with these reduced matrices is a least-squares approximation of the original matrix S. As a consequence, it is safe to remove all eigenvectors and eigenvalues whose strength is below a given threshold, or to retain only the top k strongest vectors, thus removing noise.

Typically this dimensionality reduction is achieved by first ordering the columns vectors of U, row vectors of V, and singular values of Σ, by the strength of the concepts, identified by each value in Σ. Once re-ordered, the strongest k concepts are retained, and all other

concepts are discarded. In terms of the concatenated representation only some of the concepts retained will directly relate to the relationships between content and tags, but this step is necessary to discard the weaker correlations considered as noise.

Calculating Similarity

To calculate the similarity between two tracks, the representation of each track must first be projected onto the generalised search space. The track representation for query track q is projected into a concept vector $\hat{q}$ as

$$\hat{q} = \Sigma^{-1} U^T q^T \tag{6.11}$$

This equation translates the query vector q into the concept space V, which describes the concepts that capture relationship between content and tags. If the query was not projected into the concept space, then queries which only have content available, and no tags, would not benefit from the generalisation. Through projecting queries, further meaningful tag knowledge can be introduced through the correlations learned between content and tags. To calculate the similarity between the projected query vector $\hat{q}$, and the potential recommendation $\hat{r}$, the strength of each concept must also be included. Cosine similarity is appropriate, because each concept is based on direction, and so similarity is calculated as

$$\text{sim}(q, r) = \text{sim}(\Sigma\hat{q}, \Sigma\hat{r}) \tag{6.12}$$

6.4 The Effect of Generalisation

The effect of generalising content and tag data using SVD is that, when a new track's vector is projected, both the content and tag representations are effectively refined by a re-weighting scheme. This re-weighting is illustrated in the following example, where Figure 6.2 shows the un-normalised input matrix S describing the tag and content vectors of each track.

Tracks T_1, T_2, and T_3 are each represented by both content and tags, and tracks T_4 and T_5 are represented by content only. Tracks T_4 and T_5 are cold-start; the weights of

T_1	T_2	T_3	T_4	T_5
3.1	3.2	3	3.1	3.3
3.4	3.2	3.2	3.1	3.3
4.1	4.8	4.1	3.2	4.2
3	3.1	3	2.8	2.9
100	100	0	0	0
0	0	50	0	0
50	60	10	0	0
20	10	0	0	0
0	0	40	0	0
0	10	0	0	0
0	30	5	0	0
100	0	40	0	0

Figure 6.2: Example Concatenation Matrix of Content and Tag

all tags for the tracks are zero. The position of cold-start track T_4 and T_5 is at the right of the matrix to make the example clearer to read; in practice the order that the tracks appear in may be random, and does not affect generalisation using LSA.

This matrix is generalised using LSA, thus learning the correlations across each dimension in the data. To inspect the re-weighting effect that LSA has, the matrix may be re-represented by unprotecting each track vector from the subspaces learned by SVD, as

$$\hat{T_n} = T_n \Sigma V^T \tag{6.13}$$

where T_n is the column vector of track n in the input matrix.

The matrix obtained by unprotecting each track vector in Figure 6.2 is shown in Figure 6.3. The dimensions corresponding to the tag dimensions of cold-start tracks are highlighted in bold. After LSA, it is clear that tag information has been propagated into the representations for these cold-start tracks. This occurs because the content part of the representation matches a strong concept identified by LSA, and this concept includes tag information.

Figure 6.4 shows how each track vector has been modified in concept space, allowing deeper insight into how LSA has generalised the input matrix. This is calculated by

T_1	T_2	T_3	T_4	T_5
0.157	0.143	0.133	0.394	0.387
0.158	0.143	0.134	0.389	0.382
0.160	0.148	0.133	0.374	0.371
0.158	0.144	0.135	0.386	0.379
0.211	0.193	0.057	**0.208**	**0.207**
0.063	0.056	0.264	**0.237**	**0.235**
0.204	0.190	0.078	**0.218**	**0.218**
0.219	0.185	0.058	**0.210**	**0.206**
0.063	0.056	0.264	**0.237**	**0.235**
0.184	0.215	0.056	**0.201**	**0.210**
0.177	0.204	0.093	**0.220**	**0.228**
0.106	0.085	0.241	**0.252**	**0.248**

Figure 6.3: Generalised Concatenation Matrix of Content and Tag

subtracting each dimension value in the Figure 6.2 after normalisation is applied, from Figure 6.3. The red dimensions are ones which have their influence weakened in concept space, and the green dimensions are ones with have their influence increased.

T_1	T_2	T_3	T_4	T_5
-0.163	-0.167	-0.183	-0.114	-0.090
-0.193	-0.167	-0.203	-0.118	-0.095
-0.263	-0.318	-0.299	-0.149	-0.236
-0.152	-0.157	-0.181	-0.073	-0.041
-0.409	-0.390	0.057	**0.208**	**0.207**
0.063	0.056	-0.349	**0.237**	**0.235**
-0.106	-0.160	0.016	**0.218**	**0.218**
0.095	0.127	0.058	**0.210**	**0.206**
0.063	0.056	0.019	**0.237**	**0.235**
0.184	0.157	0.056	**0.201**	**0.210**
0.177	0.029	0.062	**0.220**	**0.228**
0.044	0.085	-0.004	**0.252**	**0.248**

Figure 6.4: Learning by LSA

LSA has weakened all of the content-based dimensions. This is perhaps expected, since it is known that content is the weaker representation of the two. The majority of the tag-based dimensions have had their influence increased. This increase however is not uniform across all tracks; the cold-start tracks have had their tag-based dimensions

increased significantly. This behaviour is highly desirable, allowing well-tagged tracks to maintain their strong information, and increasing the potential of cold-start tracks to be recommended.

6.5 Evaluation of ConcatLSA

The purpose of generalising the concatenated representation is to increase the discovery rate of cold-start tracks in a recommender system. However, this representation must also be able to provide consistently high quality recommendations. Therefore, prior to investigating cold-start discovery rate, the recommendation quality achieved by the representation is evaluated. The representations that will be compared are:

- **Content** - the MFS-LSA model described in Chapter 5 is used to provide content-based recommendations.
- **Tag** - the tag-vectors collected from Last.FM provide a benchmark recommendation quality
- **ConcatLSA** - the generalised representation defined in §6.3

Recommendation quality is measured as described in §4.2, and the mean average association score is presented.

6.5.1 Recommendation Quality Results

Figure 6.5 shows the recommendation quality achieved by each of the methods being evaluated. Both Tag and ConcatLSA are clearly able to out perform Content, since they both contain the stronger tag knowledge. However, the performance of Tag and ConcatLSA is comparable: for the first 7 recommendations there is no significant difference in recommendation quality; and when more than 7 recommendations are made Tag outperforms ConcatLSA.

The reason Tag and ConcatLSA are so similar is that only 2% of tracks in the dataset used for evaluation, described in §3.2, are represented by zero tags. The tag knowledge is much stronger than content, and so in this case LSA will tend towards learning concepts

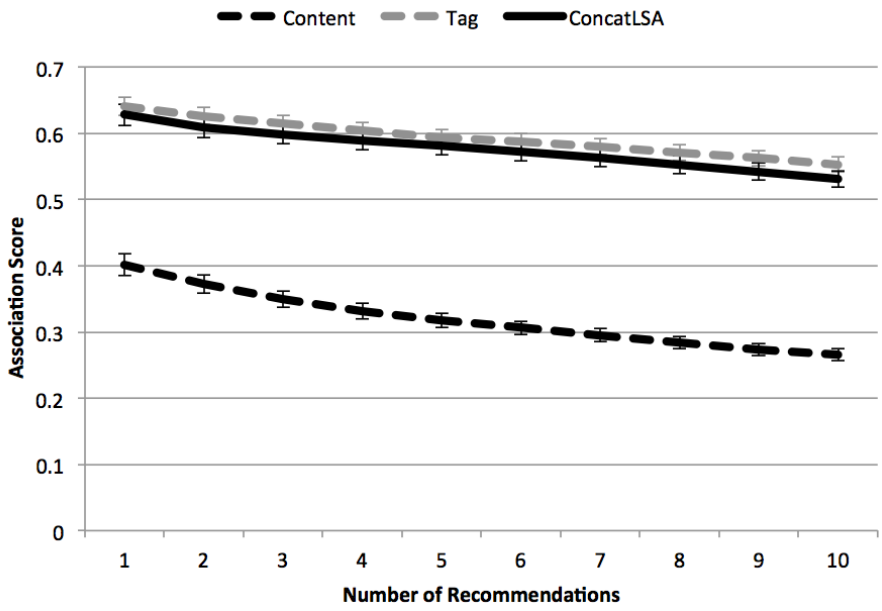


Figure 6.5: Recommendation Quality of ConcatLSA

based on this knowledge. To investigate how ConcatLSA compares with the Tag representation when a higher rate of cold-start is present, tags are removed from a randomly selected 25% of the collection, and the experiment is repeated. This level of cold start more closely reflects what occurs naturally within a live recommender system, as discussed in §2.3.2.

Figure 6.6 shows the recommendation quality achieved when cold-start is injected into 25% of the collection. As expected, the Tag representation, illustrated by the grey dashed line, is still able to outperform the Content method by a significant amount. However, when compared with the performance of Tag on a well-tagged dataset, the recommendation quality has dropped a great amount, from 0.63 to 0.45 when a single recommendation is made. This clearly illustrates how significant a problem cold-start can be to a tag-based recommender system.

The ConcatLSA method, illustrated by the solid black line, is now able to provide significantly better quality recommendations than both the Content and Tag methods. ConcatLSA is able to correctly identify discriminant concepts within the representations of well-tagged tracks, and propagate this information into the representations of cold-start tracks. However, to achieve a higher recommendation quality than tags, generalisation

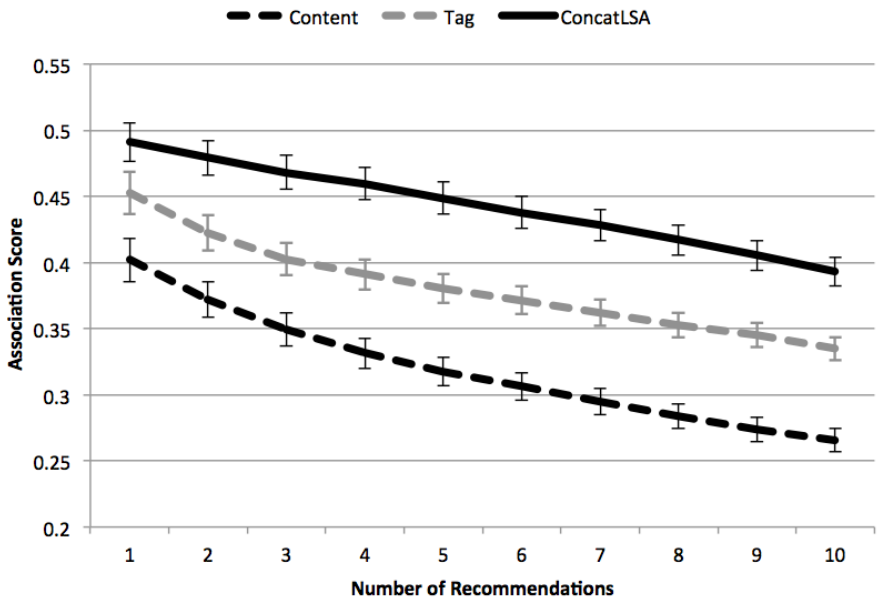


Figure 6.6: Recommendation Quality of ConcatLSA with Injected Cold-Start

must also propagate meaningful information into the representations of cold-start tracks. The results obtained therefore illustrate that LSA is able to achieve this successfully, creating concepts based on well represented tracks, and then moving cold-start tracks closer to these concepts in the concept-space. Note that LSA is not used on the Tag representation. This was tested during initial evaluations, but was found to significantly weaken the recommendation quality when using tags. The key difference for ConcatLSA that allows generalisation to work is the inclusion of content into the representation.

6.6 Cold-Start Discovery Results

Measuring cold-start discovery rate provides further insight into how ConcatLSA is able to outperform the Tag representation when cold-start is present. Discovery is the ratio of untagged items which are recommended, when the query track is tagged. Figure 6.7 shows the average discovery rate of each representation for the first 20 recommendations. The Content representation roughly follows the random cold-start discovery rate of 25%; cold-start was injected at random, and so this result is expected.

Both the Tag and ConcatLSA representations achieve a lower discovery rate than Content. However, these results must be interpreted with Figure 6.6 in mind; it is only

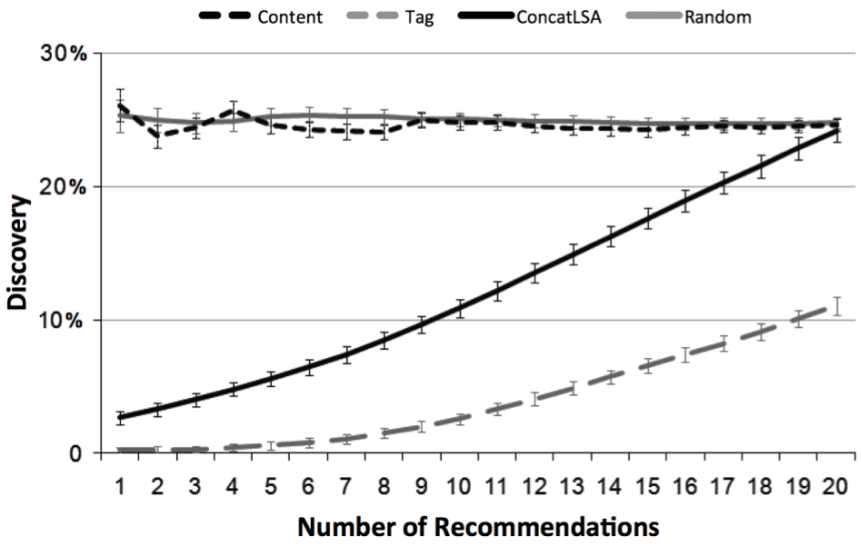


Figure 6.7: Discovery of Cold Start Tracks

desirable to achieve a high discovery rate if a good recommendation quality can be maintained. The Tag representation achieves a discovery rate of approximately 0 for the first 4 recommendations, and then gradually increases. The reason for this is that the tag representation always finds tracks that are tagged first, and when no more tracks exist with matching tags recommendations are made randomly.

In contrast to the Tag method, ConcatLSA achieves a discovery rate of 0.03 for the first recommendation. Similar to Tag, the discovery rate of cold-start tracks with ConcatLSA also increases as more recommendations are made, only at a slightly faster rate. When 20 recommendations are made, the cold-start discovery rate peaks at 25%, the level achieved by both Content and Random. This increased discovery rate is because cold-start tracks are generalised alongside well-tagged tracks, and the concepts learned cover both.

The improved discovery rate of ConcatLSA over Tag is achieved while also maintaining a higher recommendation quality over Tag. This further illustrates that the concepts learning by ConcatLSA are meaningful, and are able to introduce useful information into the representation of cold-start tracks.

6.7 Summary

Content-based representations offer a method of including all tracks in a collection when the recommender system has access to each track's audio. However, the quality of recommendations generated based on a content-based representation is not high enough to provide a good quality system. Tag-based representations allow for higher recommendation quality than content, but suffers from the cold-start problem.

To overcome this cold-start problem, an approach to combining both content and tags into a single representation has been developed. This ConcatLSA representation first concatenates content and tags in such a way that each is given an equal influence when both are present. Latent Semantic Analysis is then used to generalise both representations together, and create a new concept space. An investigation of this concept space reveals that LSA is able to create concepts that contain both content and tag information.

When tracks are projected into the learned concept space, the concepts which they are mapped to contain both content and tag information. Thus, when cold-start tracks are projected, tag knowledge is added. This allows the tracks to gradually be included in recommendations, even when the query is well-tagged.

Results shows that the ConcatLSA representation is able to provide significantly better quality recommendations than both the Content and Tag representations when cold-start is present. This improvement comes from improving the discovery rate of cold-start tracks in the recommender system. Importantly, this increase discovery rate is not achieved by randomly including cold-start tracks, but by including cold-start tracks that are relevant to the query.

Chapter 7

Learning a Pseudo-Tag Representation

Music that is well-tagged allows a system to produce strong recommendations. However, many tracks within a typical collection are under-tagged, and for these tracks recommendation quality is poor. The lack of information available in under-tagged tracks also means that they are likely to be excluded from recommendations, thus increasing the time it takes for users to discover the tracks and tag them. Content-based representations can give support for less tagged tracks, since content representations are always available. However, the quality of recommendations made using content-based representations is comparatively poor, and cannot guarantee high quality recommendations to the user.

One approach to addressing this problem is to propagate the collective knowledge captured in the existing tag corpus. Auto-tagging is one such approach, where the goal is to propagate tags throughout the collection, thus increasing the level of tagging. However, existing auto-tagging techniques commonly address this task as a classification problem, where the goal is to assign each tag to a set of tracks from the collection. The first drawback of this approach is that the weights of each assigned tag are not necessarily calculated, and so only limited knowledge is available to a recommender system. A further drawback is that, to increase the level of tagging through-out a collection, it makes more sense to assign tags to tracks. Auto-tagging approaches model each tag, and then find a set of

tracks which most closely fit this model and assign the tag to them. There is no guarantee that every track will be used, and thus no guarantee that a track will be assigned any tags, let alone enough to become well-tagged. This in turn means that auto-tagging alone is not a viable approach to effectively reducing the cold-start problem. Ideally, tags that are learned will be used to construct a hybrid representation including both learned and social tags, and to achieve this each track must have a set of learned tags, each with a corresponding weight.

In this chapter a new approach to auto-tagging is defined, that is more suitable to addressing the cold-start problem introduced by under-tagged tracks. A discussion of the collective knowledge tags describe is presented first, followed by a definition of the auto-tagging task, and the new pseudo-tagging task. A new approach to learning pseudo-tags based on content is then presented. This approach is evaluated against a baseline and similar significant approach, both in terms of tagging accuracy, and how pseudo-tags compare with content for music recommendation.

7.1 Tapping Into Collective Knowledge

To give an idea of how people tag music, Table 7.1 shows the frequency of the top 10 tags applied within the Last.FM dataset (Bertin-Mahieux et al. 2011). This dataset contains $505,216$ tracks, and $522,366$ unique tags, and 6 of the top 10 tags describe a genre. One of the tags describes a musical aspect ('female vocalists'), one describes an opinion ('favourites'), one describes context ('00s'), and the remaining tag may describe either an opinion or song topic ('Love').

Tag	**Frequency**	**Tag**	**Frequency**
rock	101,071	female vocalists	42,565
pop	69,159	favorites	39,921
alternative	55,777	Love	34,901
indie	48,175	dance	33,618
electronic	46,270	00s	31,432

Table 7.1: Most Frequency Tags in Last.fm Million Song Dataset

Bischoff et al. (2008) examine the tagging behaviour of Last.fm users in more detail. The authors define eight distinct categories of tags; topic, time, location, type, author, opinion, usage, and self-reference. Of these categories, they report that around 45% of the tags they sampled from Last.FM describe 'type', which primarily consists of genre tags, but also instrumentation and other musical qualities. This means however that around 55% of tags found in Last.FM do not describe musical qualities.

The observation that most tags do not describe musical qualities is important to the task of auto-tagging, and can have tangible implications. Many auto-tagging approaches rely on the use of content, and thus it may be reasonable to argue that only content-based tags can be learned; only 45% of tag data is useful. The power of a tag-based approach however may be due to the richness and diversity of the descriptions, perhaps explaining why content-based approaches are so far behind in terms of quality. If auto-tagging is to be useful, then the task and approaches used must be well defined, and able to bridge the semantic gap from content to all all tags, not just musical ones.

7.2 The Auto-Tagging Task

The auto-tagging task is usually to assign each tag in a collection to a set of correct tracks (Turnbull, Barrington, Torres & Lanckriet 2008, Panagakis & Kotropoulos 2012, Nam, Herrera, Slaney & Smith 2012). As illustrated in Figure 7.1, the inputs for this task are typically a set of audio features and tag data, which are linked through the track they both describe. A learning algorithm then uses this data to construct an auto-tagging model, and outputs a list of automatically generated tags, applied to tracks.

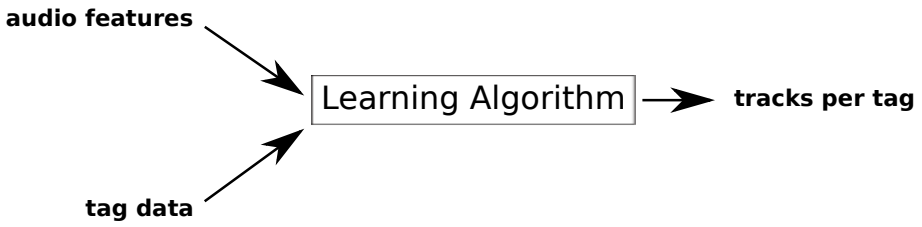


Figure 7.1: Auto-Tagging Task

The task of auto-tagging is designed to answer the question, 'given tag t, which tracks should I apply it to?'. When considering the task of cold-start, this auto-tagging task

definition can be problematic. Tag t cannot be applied to all tracks, and so a threshold must be introduced. This threshold is typically set to 10 when evaluating an auto-tagging system.

Classifying each tag as describing a certain track measures how well a tag is modelled by an audio representation, but it is not an approach to reducing cold-start. It is possible that some tracks in the collection will never have an auto-tag assigned to them, since each tag is only assigned to a limited number of tracks. To reduce the effects of cold-start by learning tags, the task must be restructured to ensure that each song is guaranteed an appropriate tag-based representation.

7.3 The Pseudo-Tagging Task

In order to address the cold-start problem fully, auto-tagging must be designed to answer the question, 'given track s, which tags should I apply to it?' - this task will be referred to as pseudo-tagging. Figure 7.2 illustrates this task, where the input data is now the audio features of a track, alongside the tracks tags. The learning algorithm this time does not associate audio features with auto-tags, but rather a tracks set of audio features with pseudo-tags.

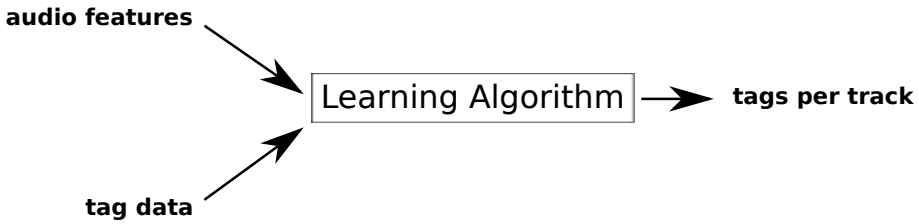


Figure 7.2: Pseudo-Tagging Task

Through re-framing the auto-tagging task in this way, it can be guaranteed that pseudo-tags will be learned for every track in a collection. It is also possible to guarantee the number of pseudo-tags that will be learned for each track, regardless of if a track has any human-generated tags or not. The result of this is that the cold-start problem, from a representational point of view, will be removed from the collection.

Pseudo-tagging is guaranteed to provide every track in a collection with tags, but these tags must be meaningful and useful. The motivation behind pseudo-tagging is not

to re-work the tag classification problem, it is to improve recommendation quality when cold-start is present. For this reason the quality of pseudo-tagging should not be measured only by classification accuracy, but also by how the pseudo-tag representation allows a recommender system to perform.

7.4 Learning Pseudo-Tags

In Chapter 5 the MFS-LSA content-based model was developed, which recommends a list of tracks given a single track query. MFS-LSA can be used not only for recommendation, but also to generate psuedo-tags for a track. The first stage in generating pseudo-tags for a track is to find tracks which are similar to a query; the assumption is that musically similar tracks will be described by semantically similar tags.

The content-based MFS-LSA model is used to retrieve the top K most similar tracks to a query. Each of these K tracks will have an associated tag representation, which can then be used to learn pseudo-tags for the query track. One simple approach to learn pseudo-tags is to sum all the weights in the retrieved tag-vectors. A more meaningful approach is to weight the contribution of each tracks' tag-vector based on how similar it is to the query.

However, content-based similarity can vary depending on the query track, and so a more consistent weighting is required. The retrieved tracks can be considered a ranked list, where the the nearest neighbour, $k = 1$, is most similar to the query. The tag-vector of each track in this list can then be weighted as

$$w_k = 1 - \frac{k-1}{K} \tag{7.1}$$

where w_k is the calculated weight, k is the position of the track in the nearest neighbour retrieval in the range 1 to K.

The pseudo-tag vector p for a given query track is therefore calculated as

$$p_i = \sum_{k=1}^{K} w_k t_i(k) \tag{7.2}$$

where $t_i(k)$ is the term-frequency of the ith tag in the tag vector of track at position k in

the nearest neighbours.

Given this pseudo-tag representation, it is now possible to recommend tracks using only tag-based methods, where the pseudo-tag vector is used as a substitute for the queries mean MFS vector. This approach will be referred to as `PseudoTag`.

7.5 Evaluation of Pseudo-Tags

Pseudo-tagging quality, and recommendation quality when pseudo-tags are used are both important measures to understand how effective the method presented is. Pseudo-tagging quality provides insight into how accurately pseudo-tags reflect human tagging behaviour. It is important that meaningful pseudo-tags are learned for each track, thus providing a strong replacement for a tag-based representation.

Recommendation quality is then evaluated to compare how well pseudo-tags perform against content-based recommendation. It is important that pseudo-tags are able to offer a strong representation for a recommender system, since pseudo-tags may be integrated more easily with tag-based representations, hopefully providing a strong hybrid recommender system.

7.5.1 Measuring Pseudo-Tag Quality

Pseudo-tagging quality is measured by calculating the per-track precision and recall against the track's ground truth tags. Precision measures the percentage of pseudo-tags correctly applied to a track, and recall is the percentage of a track's tags which are generated as pseudo-tags. Precision and recall are calculated as

$$\text{precision}(s) = \frac{\sum_{i=1}^{P} rel(p_i \| s)}{P} \tag{7.3}$$

$$\text{recall}(s) = \frac{\sum_{i=1}^{P} rel(p_i \| s)}{T} \tag{7.4}$$

where $rel(p_i \| s)$ is 1 if pseudo-tag p_i has a weight greater than 0 in the tag-vector of track s, and 0 otherwise. T is the number of tags in the tag vector of track s, and P is the number of pseudo-tags generated.

For pseudo-tags, it is more important that a high precision score is obtained than recall. Pseudo-tagging is not a classification task, or designed to provide annotations for humans; pseudo-tags are intended to be a representation that can for used in cold-start recommendations, and so must be accurate. Recall of the system will vary depending on how many pseudo-tags are required, but it is important that precision is strong for all levels of recall.

7.5.2 Comparison Methods

The performance of the pseudo-tagging approach described in this section will be compared to a baseline method, and another k-NN method from Sordo et al. (2012).

Random Baseline

A simple random baseline would only assign each song a randomly selected set of tags from the collection. This measure however does not provide much insight into how the pseudo-tagging method used behaves. A more interesting baseline method is to randomly select tags from only the k nearest neighbours. This baseline will be referred to as **random**, and will provide further insight into how the pseudo-tagging method proposed selects tags from the nearest neighbourhood.

Sordo Method

Pseudo-tagging will also be compared with a current state-of-the-art k-nearest neighbour approach (Sordo et al. 2012). Sordo's method involves retrieving N tracks for each track that must be annotated, and giving a higher weighting to the tags of the nearest K tracks. The weighting function is defined as

$$w_k = \begin{cases} 1 & \text{if } k < K \\ \frac{1}{k^2} & \text{otherwise} \end{cases} \tag{7.5}$$

and this weighting function is then used to count the influence of all tags of the nearest neighbour tracks, calculated as

$$p_i = \sum_{k=1}^{N} w_k \lceil t_i(k) \rceil \tag{7.6}$$

where $\lceil t_i(n) \rceil$ evaluates to 1 if $t_i(k)$ is present, and 0 if it is not.

In Sordo's method, N is set to the number of songs in the training set, and K is set in the same way as for random and pseudo-tags. The effect is that the first K tracks all get an equal vote, and the votes of the remaining $N - K$ tracks have an inverse square decay.

7.5.3 Pseudo-Tag Quality With CAL-500 Dataset

Pseudo-tagging quality is first evaluated with the CAL-500 dataset, which is described in detail in §3.1. Figure 7.3 shows the mean precision obtained when 10 pseudo-tags are learned; the error bars shown are at a 95% confidence interval from the mean. The solid black line shows the precision obtained when using the `PseudoTag` method, the grey line shows that of Sordo's method, and the dotted line shows the baseline random method.

As expected, the performance of the random approach, illustrated by the dotted line, is much lower than both `PseudoTag`and Sordo. When K is 10 both the `PseudoTag` and Sordo methods perform similarly, with a precision of approximately 0.68. As K is increased however, each method starts to behave differently. Precision of the `PseudoTag` method improves until K is around 40, whereas precision of the Sordo method gradually decreases. For values of K greater than 10, `PseudoTag` achieves significantly better precision than Sordo.

It is interesting that precision of the random method also decreases as K is increased, with a large decrement occurring between $K = 20$ and $K = 30$. The `PseudoTag` method takes the position of each track in the content retrieval list into account. In the Sordo and random methods, each track in the near neighbourhood is considered to be equally important, and thus the tag-vectors of the K nearest tracks are all considered equal. The reason that the `PseudoTag` method is able to achieve significantly increased precision over the Sordo method is that each of the K nearest tracks are not treated equally. The weighting function that we apply allows the tags of the most similar tracks to have a larger influence, and to effectively control how tags are learned.

As K is increased beyond 40, the precision of all methods decreases. This exact

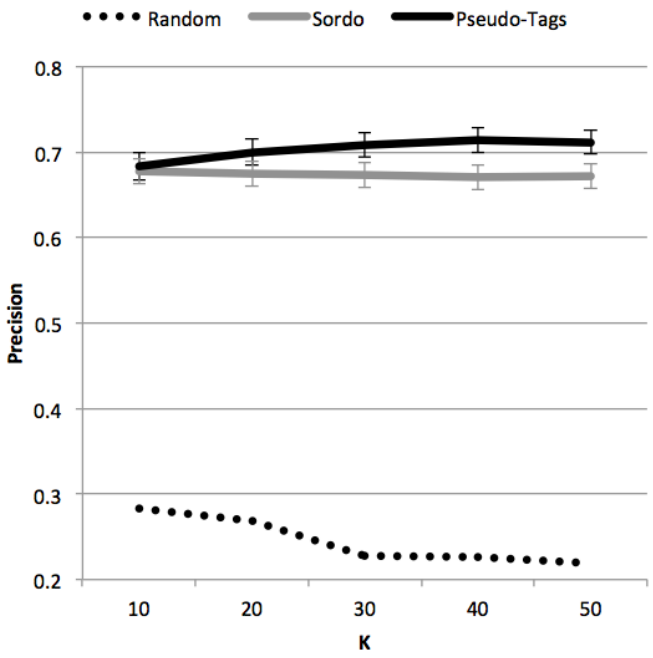


Figure 7.3: Precision learning 10 pseudo-tags at varying K using CAL500 Dataset

threshold is specific to the CAL500 dataset, but the trend is not. Increasing K allows more tags to be found, and the weights of these tags will gradually increase as more tracks are included. For example, with the Sordo method, when K is equal to the size of the training set, precision is reduced to that of when the most popular tags are assigned.

Figure 7.4 shows the recall-precision curve of each method. The points were generated by evaluating the precision and recall of each method for between 1 and 100 pseudo-tags learned. For both `PseudoTag` and Sordo, precision decreases as the number of pseudo-tags learned is increased. This result is expected, since the chance of making an incorrect pseudo-tag assignment also increases. However, the `PseudoTag` method that will ultimately be used for recommendation, maintains a reasonable precision for all numbers of pseudo-tags. This is very important, and must be interpreted slightly differently to standard auto-tagging tasks.

When auto-tagging is used for suggesting tags to users, or providing a permanent tag representation, typically 10 or 20 tags are suggested. For the task of creating a hybrid representation however, many more pseudo-tags may be required for tracks which are very under-tagged. Regardless of how many pseudo-tags are required, we need them to be as

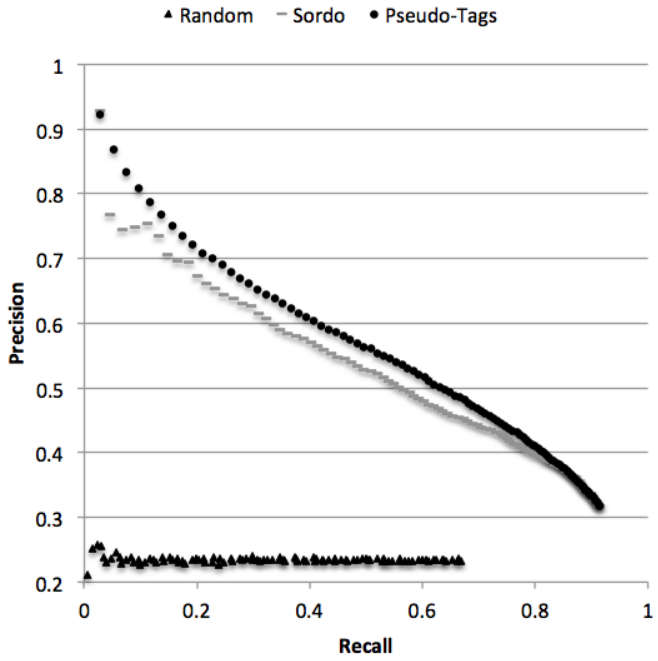


Figure 7.4: Recall Precision Curve using CAL500 Dataset

precise as possible. It is therefore useful that this method is able to provide a competitive precision even when a large number of pseudo-tags are learned.

7.5.4 Pseudo-Tag Quality With Private Dataset

The private dataset, described in §4.2, has different properties to the CAL500 dataset. It is therefore interesting to evaluate the different methods using this real-world distribution of tagging. Figure 7.5 shows the recall-precision curve of each method, using the private dataset. The points were again generated by evaluating the per-song precision and recall of each method for between 1 and 100 pseudo-tags learned. The figure is in the same format as Figure 7.4.

The `PseudoTag` method performs well when applied to the private dataset, with the first predicted pseudo-tag having an average precision of 0.72. The private dataset contains many more tracks and tags than the CAL500 dataset, and tagging follows a completely different distribution. This difference in data explains why the `PseudoTag` curve drops more steeply for this dataset.

The results obtained using Sordo method at first appear erratic, however, these results

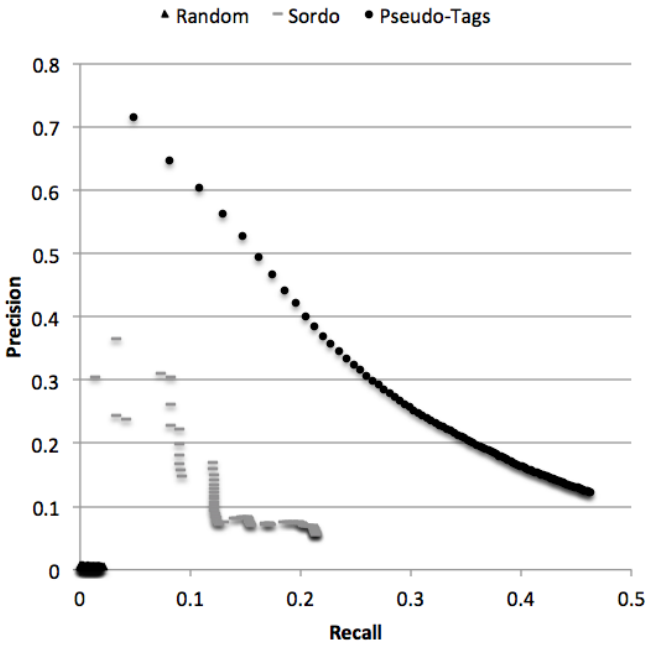


Figure 7.5: Recall Precision Curve using Private Dataset

are again a consequence of the large real-world dataset. For this method, a seemingly random set of results are obtained when predicting a low number of pseudo-tags, and then two 'columns' are found in the data. Within these columns, results have a similar level of recall, but precision decreases as the number of predictions made increases. Finally, there is a tail at the end of the results.

The scattered results achieved for low levels of recall are when few pseudo-tags are predicted. In this case, the Sordo method is learning pseudo-tags based on counting occurrences in the K nearest neighbours. The effect here is that the most common tag is learned first, regardless of how strong its association is to a track. The columns that are seen in the results occur when a large number of candidate pseudo-tags have the same level of occurrence in the data, in this case 3 or 2 nearest neighbours. When this occurs, the weighting function eq. (7.5) acts as a tie breaker, and ranks pseudo-tags. This weighting function is limited to tie breaking since it tends towards a limit less than the counting weight of 1 the first K neighbours get.

$$\Sigma_{k=11}^{\infty}\frac{1}{k^2} = \Sigma_{k=1}^{\infty}\frac{1}{k^2} - \Sigma_{k=1}^{10}\frac{1}{k^2} = \frac{\pi^2}{6} - \Sigma_{k=1}^{10}\frac{1}{k^2} = 0.0869 \tag{7.7}$$

These results clearly illustrate the effectiveness of weighting pseudo-tags based on the rank of the K nearest neighbours, as opposed to counting occurrences. Even with such small nearest neighbourhoods, where one may assume all tags are relevant, tracks which are more content-similar offer stronger tags which can be learned as pseudo-tags.

7.5.5 Pseudo-Tag Recommendation Quality

The reason for learning pseudo-tags is ultimately to improve recommendation quality, by including them in a hybrid representation. This means that pseudo-tags should be able to offer better recommendation quality than a content-based recommender system; if this is not the case, then perhaps it is more meaningful to use content instead. To evaluate how `PseudoTag` compares to content, recommendations are measured as described in §4.3.

Figure 7.6 shows the average association score achieved when between 1 and 10 recommendations are made. The dashed black line illustrates the performance of a purely content-based recommender system, and the solid line shows when `PseudoTag` is used. When only 1 recommendation is made the two methods are comparable, however, when more than one recommendation is made `PseudoTag` offers significantly improved recommendation quality. Further, the `PseudoTag` method is able to offer a more consistent level of recommendation quality across varying numbers of recommendations.

The reason that `PseudoTag` is able to out-perform content-based recommendation is that, while they are based on content, pseudo-tags also describe more information. Tracks that are musically similar may also be contextually similar, something that is missed by content-based methods. With `PseudoTag` however, this context information is also learned, providing a more knowledge-rich representation.

The grey dashed line in Figure 7.6 shows the performance of `PseudoTag` when the weighting scheme in eq. (7.1) is not applied; that is, the weight is always 1. In this case the performance of `PseudoTag` suffers, and even performs worse than content for 1 recommendation. This highlights the effectiveness of the weighting scheme used, and further reinforces the importance of content-based similarity when learning pseudo-tags.

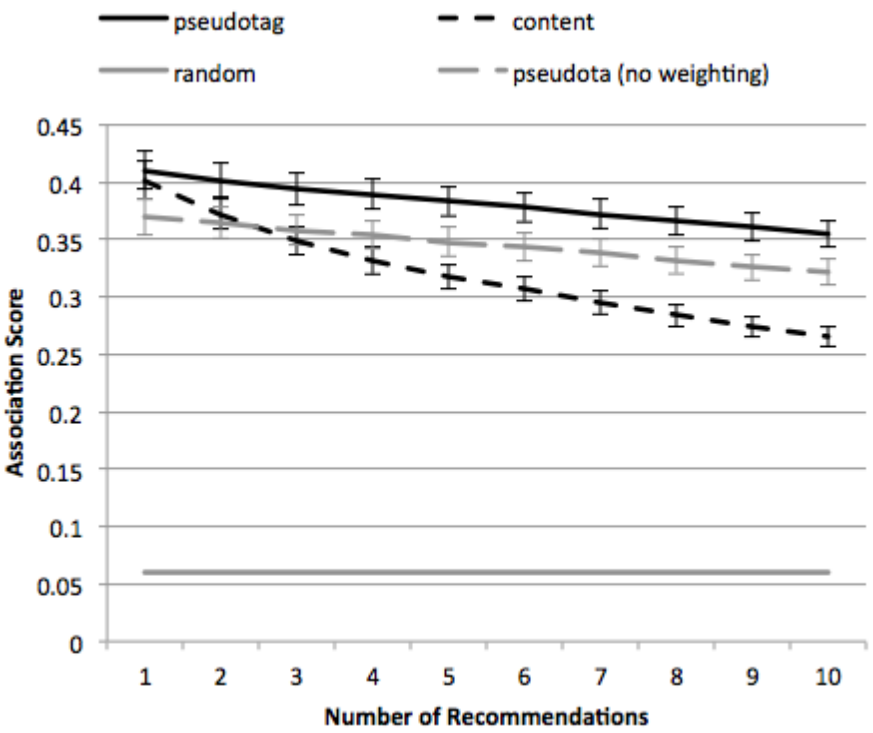


Figure 7.6: Recommendation Quality of Pseudo-Tags

7.6 Summary

Content-based representations can be used to make recommendations in the absence of a strong tag-based representation. However, content alone cannot offer the same performance as a strong tag-based recommender. Further approaches are required to reduce the semantic gap between content and tags, and allow recommendation quality to be improved when tracks are under-tagged. One such approach is auto-tagging, where the goal is to propagate tags to tracks.

The task of auto-tagging is reasonably well established within the field of Music Information Retrieval, however, this task is often defined as a multi-class classification problem, where each tag must be assigned to a set of tracks. In order to create a representation to address cold-start this task definition has been modified, so that each song is assigned a set of tags. This reformulation is subtle, but necessary to develop an approach that can be used in cold-start recommendation scenarios.

The `PseudoTag` method developed is able to offer significantly improved recommen-

dation quality, when compared to a content-based method. Although these pseudo-tags are learned from the content-based method, the representation that is learned is much stronger. The reason for this comes from knowing what tags describe; people often tag content related terms, but also tag contextual information. This presents more knowledge that can be taken advantage of by a pseudo-tag representation. Given a set of similar songs, there are trends across tags that cover both content-based meaning, and also contextual meaning.

Experimental results show that it is not enough to simply use the most common tags that occur in a set of nearest neighbours of a track. Content-based similarity is meaningful, and capturing this through the rank of similar tracks can help improve the quality of tags that are learned. A weighting scheme was developed that successfully captures this similarity meaning, allowing the tags from a track's nearest neighbours to be learned effectively. Learning pseudo-tags in this way allowed an improvement over the recommendation quality of content to be made, and introduces a less sparse representation than tags for each track.

Pseudo-tags present new possibilities for creating more powerful hybrid representations. Defining such representations using two completely different representations, such as content and tags, can become difficult. A lot of effort can be spent towards normalising and warping the two differing data types so that they can be used coherently together. Pseudo-tags remove this obstacle; the data is learned algorithmically rather than created by humans, but the abstract representation is identical. As a result the focus when developing a hybrid representation can shift from data integration, towards creating a representation that captures the information required to make good recommendations.

Chapter 8

Dynamic Hybrid Representation

A major obstacle in the way of providing high quality recommendations is the cold-start problem. In previous chapters this problem has been viewed as tracks having no tags, and the model in Chapter 6 was developed to tackle this challenge. However, the cold-start problem may be better described as when tracks have a weak representation. Tracks that have very few tags, especially when these tags are generic such as 'rock', suffer similar problems to tracks without any tag representation.

Figure 8.1 illustrates this type of cold-start problem in music recommender systems. The horizontal axis shows increasing number of tags for a track, and the vertical axis shows increasing recommendation quality. The content-based representation does not suffer from cold-start, and is illustrated by the horizontal dashed line. The performance of a content-based recommender system is expected to be consistent regardless of how many tags a query track has, since tags are not used. However, the recommendation quality achieved by content-based systems is not as strong as when a good tag-based recommender system is used.

The dotted line in Figure 8.1 illustrates the expected recommendation quality of a tag-based recommender system. When no tags exist, recommendation quality is likely to be poor, shown at point n_0. As the number of tags available increases however, recommendation quality will improve, until eventually after point n_1 the quality is higher than the content-based recommender system. This point n_1 is the threshold at which a track

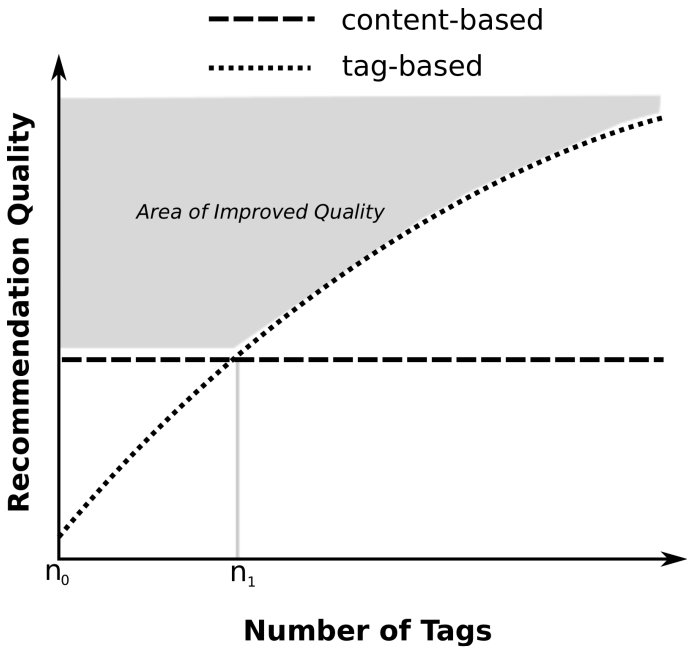


Figure 8.1: The Cold-Start Problem

moves from being under-tagged to well-tagged.

If the recommendation method is switched from content- to tag-based at point n_1, then the cold-start problem for tracks below n_1 may be avoided. However, the point n_1 is in practice not easy to define, since the point at which tag-based recommendations outperform content-based recommendations will vary across each track. This means that this hybrid system of 'swapping' the recommendation method used is not practical. In any case, the recommendation quality of cold-start tracks is unlikely to be as strong as well-tagged tracks, since it is limited to the quality of content-based methods.

A much more desirable goal for a hybrid recommender system is to provide recommendation quality that is better than either content or tags, regardless of how well-tagged a track is. This goal is illustrated by the shaded area in Figure 8.1, labelled as the 'Area of Improved Quality'. To provide recommendations that are in this area, a hybrid system must be able to learn from the content- and tag-based representations that are available, and infer a new, stronger representation.

In this chapter the motivation for using pseudo-tags, not content, in a hybrid representation is discussed. The two important stages in developing the hybrid representation will be presented; how to select the best pseudo-tags, and how to weight pseudo-tags in the

hybrid representation. This representation will then be evaluated by comparing recommendation quality with that of a tag-based system. The results of this evaluation will also discuss the effects that each representation has on cold-start discovery, providing further insight into how the hybrid representation behaves. Finally, conclusions of this work and an evaluation will be presented.

8.1 Motivation for Pseudo-Tags in Hybrid Representations

Many existing hybrid representations incorporate content into a tag-based representation, similar to the model developed in Chapter 6. The advantages of this approach are that content is always available, and contains knowledge that describes a track in a meaningful way. This knowledge is used to forge an implicit link between content and tags, and thus introduces enough knowledge into a representation to reduce cold-start.

However, a closer look at the specific knowledge that content and tags describe shows that this approach may not be ideal for hybrid representations. Figure 8.2 illustrates the information captured and described by each representation. Audio content features describe musicality and audio quality, while tag features describe higher level knowledge, such as meta-data, classes, opinions and contexts.

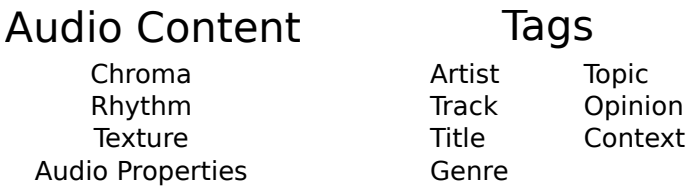


Figure 8.2: Audio Content and Tag Knowledge

This clear mis-match between content and tags limits the potential of hybrid representations that try to link both directly. The fundamental reason for this mis-match is that content representations are generated objectively, and tag representations are primarily generated subjectively. This introduces a degree of noise into any relationships that can be learned, but also means that certain relationships will be extremely difficult to learn. Noise is introduced as a consequence of disagreement between peoples opinions. An example of a relationship that is extremely difficult to learn is context; one person may listen to metal while working, and another may listen to classical while working. The content

representations of these genres will be very different, and thus no clear link may be found.

Pseudo-tags, as described in Chapter 7, offer an alternative approach to constructing hybrid representations. Using these pseudo-tags allows an intermediate step to be introduced into the process of defining a hybrid representation. This intermediate step, the learning of pseudo-tags, can help overcome some of the problems introduced when using content.

A hybrid representation using content and tags assumes there is a direct link between content features and tag features. However, pseudo-tags make a different assumption; pseudo-tags assume that tracks with similar content features will also have similar tag features. While subtle, this differing assumption has large implications when defining a hybrid representation. Firstly, pseudo-tags are able to learn tags that content cannot link to directly, such as context. Pseudo-tags also present a consistent representation structure to work with, removing the obstacles encountered when combining the differing structures of content and tags.

The hybrid representation defined in this chapter therefore combines the learned pseudo-tag vector for each track with the track's tag vector. These pseudo-tags are always available, and are thus able to address the cold-start problem. However, these pseudo-tags also capture a more knowledge-rich representation, and so this hybrid approach aims to improve recommendation quality for all tracks. In order to construct this representation there are two main stages that will be discussed; selecting which pseudo-tags to include in the hybrid representation, and the method of integrating the two representation into a hybrid representation.

8.2 Selecting the Best Pseudo-Tags

The pseudo-tagging method described in Chapter 7 adds every tag that occurs in a track's K nearest neighbours to the track's pseudo-tag vector. Each pseudo-tag is weighted based on where it appears in the nearest neighbour list, and the strength of its original assignment to the nearest neighbour. For even very small values of K, this means that the number of pseudo-tags which are learned can be very large for certain tracks, for example, when each of its nearest neighbours are well-tagged. In comparison, the number of tags that

exist in a track's tag vector is likely to be much smaller, even for well-tagged tracks. For example, if a track has an average number of tags, and each of the nearest neighbours used for pseudo-tagging also have an average number of tags, one may expect the pseudo-tag vector to contain many more values than the tag vector.

Cosine similarity will be used to make recommendations, since it is able to calculate similarity based on the relative strength of tags assigned to each track. However, since both pseudo-tags and tags are to be included in the hybrid representation, a difference in the number of non-zero dimensions in each representation can damage recommendation quality. For tracks that are well-tagged, the strong recommendation power of tags is desirable, however too many pseudo-tags will reduce this. For tracks that are under-tagged, the few tags they do have are important; it is necessary to introduce pseudo-tags, but too many may make existing tags redundant. While the introduction of all pseudo-tags available may improve recommendation quality overall, it is unlikely to increase recommendation quality into the grey shaded area in Figure 8.1.

To overcome these problems the number of pseudo-tags to be included in a hybrid representation should be restricted. Each pseudo-tag that is learned has a corresponding strength that can be used to help decide which pseudo-tags to include in a hybrid representation. This decision should also take into account how many tags exist for a track, and balance the influence of tags and pseudo-tags in a meaningful way. To investigate how many pseudo-tags should be retained, a simple experiment is conducted. The recommendation quality achieved by each track is measured when varying levels of pseudo-tags are used. The number of pseudo-tags used are chosen at intervals of 10, in the range 0 to 100. For each track, the number of tags is known, and tracks are grouped into buckets of 10 tags. The best performing number of pseudo-tags for each tag bucket is measured.

The solid red line in Figure 8.3 shows the results of this experiment. Each point along the line shows the average number of pseudo-tags that provide the best recommendation quality for tracks with a varying number of tags. The dashed black line shows the line-of-best-fit through these data points, and is the function that is used to select the number of pseudo-tags to retain. These results agree with the intuition that tags and pseudo-tags must be balanced correctly, removing a negative bias towards either representation. For well-tagged tracks a low number of pseudo-tags are used, and for under-tagged tracks a

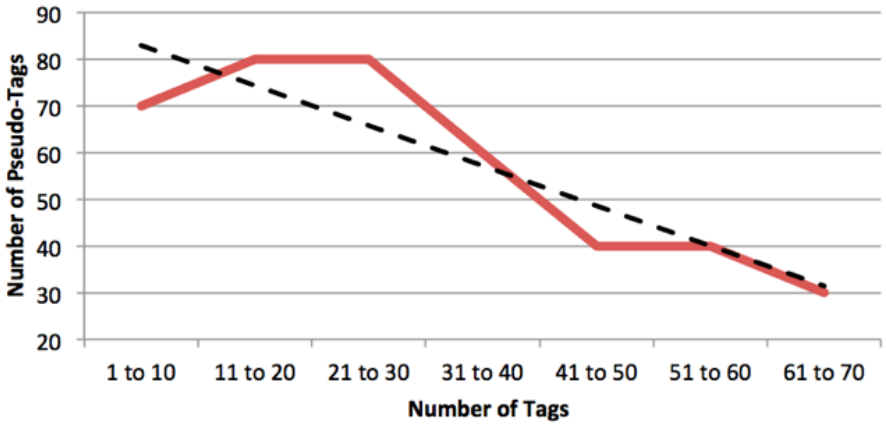


Figure 8.3: Selecting the Best Pseudo-Tags

high number of pseudo-tags are used.

An interesting and less expected observation from these results is the number of pseudo-tags required for well-tagged tracks. For tracks with between 61 and 70 tags, high recommendation quality is achieved using 30 pseudo-tags. This shows that pseudo-tags are able to contribute to a hybrid representation not only for under-tagged tracks, but also for well-tagged tracks.

8.3 Weighting Pseudo-Tags and Tags

Once the appropriate number of pseudo-tags for each track has been determined, the next step is to decide how to weight each representation. Similarly to selecting pseudo-tags, if a track is well-tagged then tags should be the most influential representation, and if a track is under-tagged, pseudo-tags should be most influential. This weighting is achieved by introducing a weighting parameter when combining the two representations. The hybrid vector h is therefore defined as

$$h_i = \alpha p_i + (1 - \alpha) t_i \tag{8.1}$$

where p is the pseudo-tag vector, t is the tag vector, and both p and t are normalised per-track prior to computing the hybrid vector h. The parameter α controls the influence of both pseudo-tags and tags. α is set to be in the range 0 to 1, where 0 means tags have

complete control, and 1 means pseudo-tags have complete control.

To set the value of α a further experiment was conducted to understand how the influence between pseudo-tags and tags should change as the number of tags a track has increases. The recommendation quality achieved by each track is measured for varying levels of α, which was set at intervals of 0.1 in the range 0 to 0.5. The upper-bound value of 0.5 was chosen to guarantee that tags will always have at least a weighting equal to pseudo-tags; tags are the stronger representation and thus should never have a lesser influence. Since cosine similarity is used, in cases where a track has no tags the pseudo-tag vector effectively has a weight of 1 because of normalisation.

The solid blue line in Figure 8.4 shows the results of this experiment. As before, the dashed black line shows the line-of-best-fit through the data points, and is the function that will be used to set the value of α. This line of best fit confirms that the weight of pseudo-tags should decrease as the number of tags a track has increases.

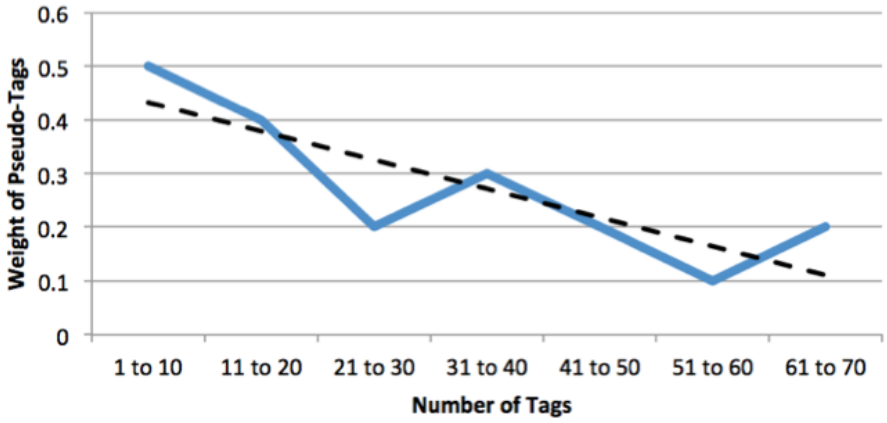


Figure 8.4: Weighting Pseudo-Tags and Tags

Comparing Figure 8.4 with Figure 8.3 shows that well-tagged tracks should be assigned 30 pseudo-tags, but that the weight of these pseudo-tags should only 0.1. This means that pseudo-tags may be used to refine the tag representation, and therefore recommendations, but they do not have a large influence. Under-tagged tracks must be assigned more pseudo-tags, and these must have a higher weighting in the hybrid combination, thus allowing pseudo-tags to have a much larger influence over recommendations generated. Less expected is that tracks with an average number of tags, 34, should be assigned approximately 60 pseudo-tags, and these should have a weight of approximately 0.25. This

is encouraging, since it hints that the inclusion of pseudo-tags in the hybrid representation may indeed be able to offer improved recommendation quality regardless of how well-tagged a track is.

8.4 Evaluation of Hybrid Representation

The hybrid representation that has been developed will be evaluated by comparing recommendation quality with when a tag-based representation is used. Recommendation quality across the entire dataset is evaluated first, allowing the effects of using the hybrid representation to be understood. Next, these results will be presented and analysed for tracks with varying levels of tagging. This allows further insights into how the hybrid representation behaves, both for cold-start and well-tagged tracks. Finally, the effects of the hybrid representation on cold-start discovery are investigated, helping to explain the effects hybrid representation has on recommendation.

8.4.1 Recommendation Quality

Figure 8.5 illustrates the recommendation quality achieved using the hybrid and tag representations. The horizontal axis shows the number of recommendations made, and the vertical axis shows the mean association score across all tracks. The hybrid representation, illustrated by the solid line, is able to achieve significantly higher quality than the tag representation for all numbers of recommendations.

This improved quality over tags is achieved because of the new knowledge introduced to the representation through pseudo-tags. For well-tagged tracks this new knowledge is able to better order recommendations, and for under-tagged tracks it introduces a more controlled approach to recommendation. Rather than simply matching the small number of tags that may be available, hybrid offers a more refined approach that allows pseudo-tags learned from other tracks' tags to be fully exploited.

To gain further insight into where the hybrid representation is achieving this improvement over tags, Figure 8.6 shows the results split by how well tagged each query is. In this figure the vertical axis shows the improvement in hybrid recommendation quality over the tag-based approach. Improvement is the difference between the quality of recommen-

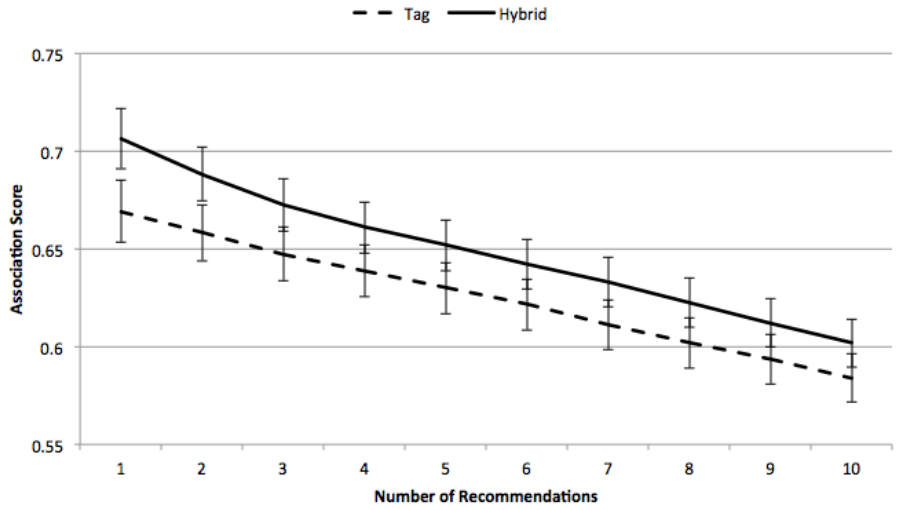


Figure 8.5: Recommendation Quality of Hybrid Representation

dations made using the hybrid and tag representations when each representation has a limited number of tags available.

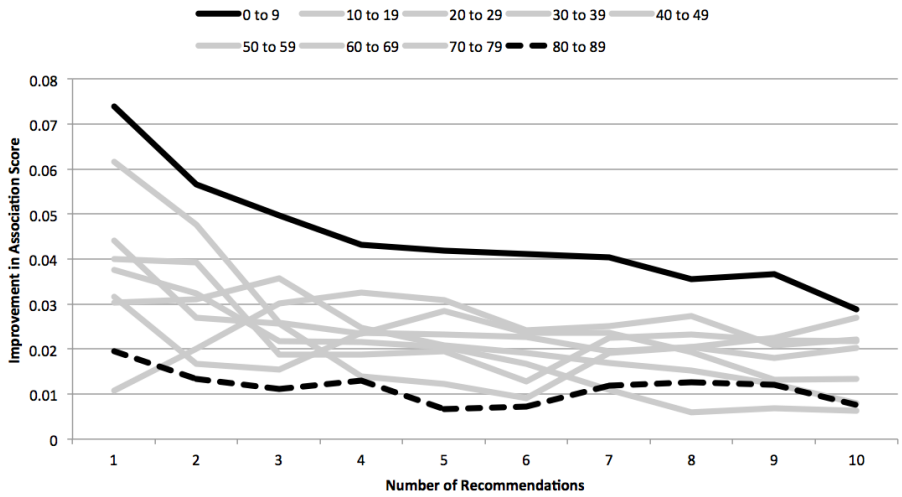


Figure 8.6: Improvement by Number of Tags Using Hybrid Representation

The solid black line shows improvement for under-tagged queries that have between 0 and 9 tags, and the dashed black line shows improvement for well-tagged queries that have between 80 and 89 tags. Both these lines are extremes of tagging level within the collection. As expected, the largest improvements are made for very under-tagged queries. When using the tag representation for recommendation, tracks with 0 tags can only generate random recommendations, which are of a very low quality. When a track has 1 tag,

then a very small number of tracks that also have this tag will be recommended before the randomly generated tracks, increasing quality only slightly. This trend continues for tag-based recommendations until a reasonable number of tags are available, making recommendations of reasonable quality.

When using the hybrid representation however, this initial phase of making random recommendations is skipped, since pseudo-tags always allow informed recommendations to be made. Thus, the hybrid representation is given a head-start over the tag representation. Similar to the tag representation, as the number of tags increases so does the quality of recommendations made using the hybrid approach. The difference between improvement over the tag representation for when a hybrid representation has 0 to 9 tags, and when it has 80 to 89 tags, indicates that the rate of improvement for hybrid is not as steep as with tags. However, this rate of improvement does not have to be as steep since the initial recommendation quality is significantly improved.

The solid grey lines in Figure 8.6 show the improvement in recommendation quality for tracks with between 10 and 79 tags, at 10 tag intervals. These reveal that the hybrid representation is able to provide higher recommendation quality than tags for all tracks, regardless of how well tagged the track is. The reason that the hybrid representation is able to achieve this is because of the subtle influence that pseudo-tags introduce for reasonably well-tagged tracks. Upon inspection of the recommendations from tags and hybrid as a group, only a few differences of recommendation are typically found. However, when looking at these recommendations as a ranked list, the order they are presented in is often very different. Pseudo-tags within the hybrid representation are able to introduce refinements into the recommendation lists that can greatly improve quality.

8.4.2 Effects on Discovery

Improvements in recommendation quality are achieved by including pseudo-tags into the hybrid representation. To fully understand how pseudo-tags are able to offer this improvement, the effects that hybrid representation has on discovery can be examined. In Chapter 6 discovery is measured as the rate tracks with zero tags are included by the recommender system. This was possible because extreme cold-start was injected into the collection, meaning 25% of tracks had no tags at all. The view of cold-start in this chapter is that it

is a progressive problem, affecting not only tracks with zero tags, but also those that are under-tagged.

To measure the effects each representation has on this progressive view of cold-start, Figure 8.7 shows the discovery rate of tracks which have less than 10, 20, and 30 tags for each representation. The layout of this Figure is as with previous figures, where the horizontal axis shows the number of recommendations that were measured. A solid line always shows the hybrid representation, and a dashed line shows the tag representation. The rate of discovering tracks with fewer than 10 tags is illustrated by the grey lines. In this case there is no real difference between the two methods; the curves fluctuate between tag and hybrid making more discoveries. However, only approximately 16 of every 100 recommendations will have fewer than 10 tags.

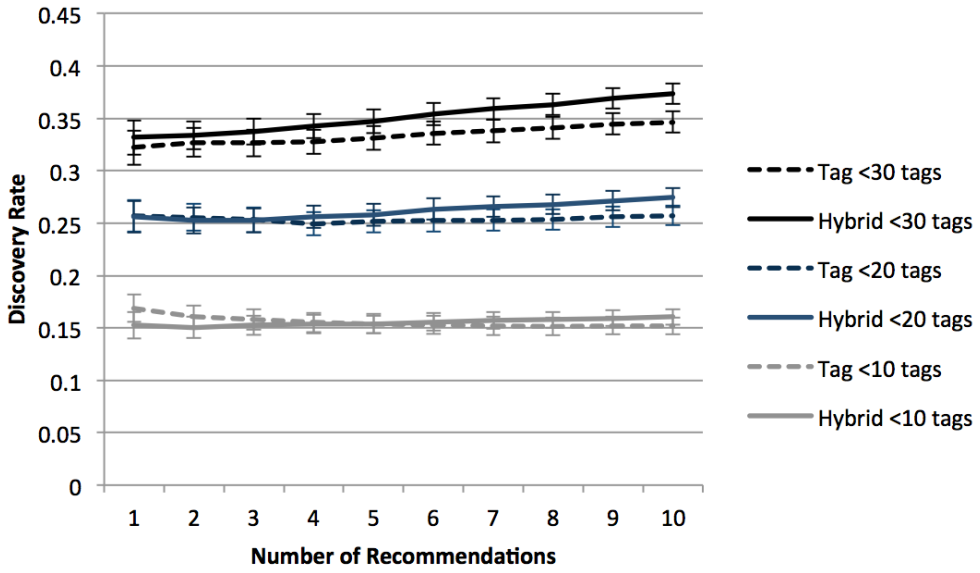


Figure 8.7: Rate of Recommending Under-Tagged Tracks

The middle two curves show the rate of discovery of tracks with fewer than 20 tags. For the first 5 recommendations both tags and hybrid are comparable, but for 6 or more recommendations hybrid achieves a significantly improved discovery rate. A similar trend is observed by the discovery rate of tracks with fewer than 30 tags, where hybrid offers a significant improvement when more than 2 recommendations are measured. No improvement is observed for the first few recommendations in each case because the tag component in the hybrid representation is most influential. There are a small number of

tracks that have many tags in common with each query, and so these are very likely to be recommended first using both representations. However, after these very strong recommendations are made, the hybrid method is able to find tracks with fewer tags that may be recommended. Further, these cold-start tracks are also of high quality, as reported in the previous section.

Figure 8.7 has shown the ability of hybrid to include tracks that are under-tagged. It is also interesting to measure the preference of each system towards under-tagged or well-tagged tracks. Figure 8.8 shows the lesser tagged rate achieved by each representation. This is the rate at which tracks with fewer tags than the query track are recommended. The rate that is achieved by a random recommender system is approximately 0.5: if a system achieves a higher rate then its preference is towards recommending lesser tagged tracks; if a system achieves a lower rate then its preference is towards recommending more tagged tracks.

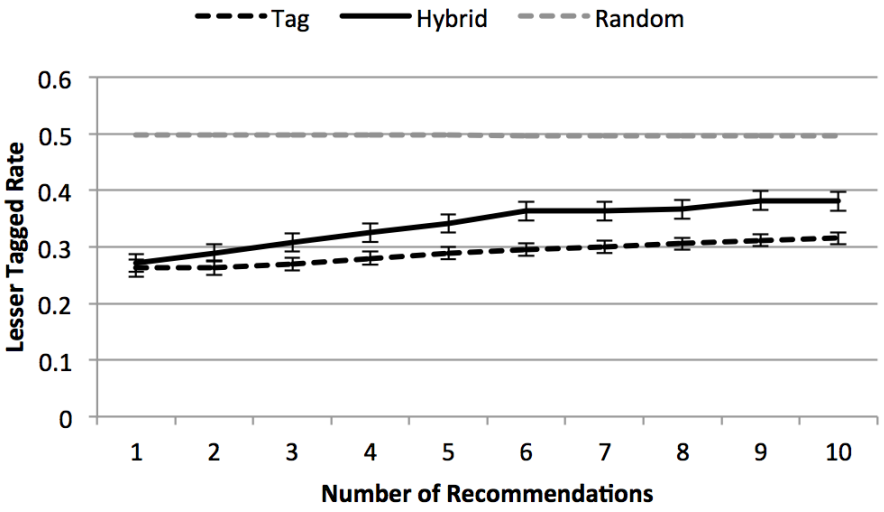


Figure 8.8: Rate of Recommending Lesser Tagged Tracks

The dashed line in Figure 8.8 illustrates the lesser tagged rate of recommendations using tags. Initially this is 0.26, and rises to 0.32 for 10 recommendations. In comparison, the solid line shows the rate using hybrid representation, which is initially 0.27, and increases to 0.38 when 10 recommendations are made. The two methods are comparable when a single recommendation is made, but when more than 1 recommendation is made the hybrid approach provides a significantly improved rate over tags.

Both methods achieve a lower rate than the random approach, showing that both methods are more likely to recommend tracks with more tags than the current query. Importantly however, the hybrid method has a weaker preference towards more tagged queries, and thus is able to offer increased cold-start discovery rate and improve recommendation quality.

8.5 Summary

The cold-start problem affects most tracks to a lesser or greater extent. As the number of tags a track has increases, so does the average recommendation quality. This opens up potential to improve the recommendation quality for all tracks in a system if a strong approach to reducing sparsity within tag representations can be realised.

When designing an approach to reducing sparsity within a tag-based representation, it is not necessarily desirable to include content directly. Content-based representations describe the musical facets of a track, but tag-based representations often describe non-musical facets, such as contexts and opinions. This provides incompatibilities between the two representations which can be difficult to overcome. However, content is always available for a track, and so may be used. The solution to this incompatibility is not to include content into a hybrid representation, but to use content to learn a hybrid representation.

The hybrid representation presented in this chapter takes advantage of pseudo-tags. The inclusion of this knowledge-rich pseudo-tag representation also removes some complex processing steps that are often required when building a hybrid representation, such as generalisation. Instead, more focus is directed towards how the two similar representations may be integrated into a single hybrid representation.

Recommendations made using the hybrid representation are of significantly higher quality than those made using a tag-based representation. This allows strong recommendations to be made even when tracks have a very low number of tags available, since these are reinforced by pseudo-tags in the hybrid approach. Further, this hybrid representation is able to improve recommendation quality regardless of how well- or under-tagged a track is.

Recommendation quality is improved because the rate of discovery of cold-start tracks is increased in a controlled manner. The hybrid representation is more likely to recommend tracks that are under-tagged than the tag-based representation. Hybrid is also able to improve significantly the rate that under-tagged tracks are introduced into recommendations. The problem being addressed was initially described in Figure 8.1, and the objective was to design a recommender system that is able to provide better recommendation quality than both content- and tag-based methods. The hybrid representation has successfully achieved this, always providing recommendations in the grey shaded area of Figure 8.1.

Chapter 9

User Evaluation

The development of new approaches to representation and recommendation have been evaluated using Association score. This score can provide a good indication of how an approach to recommendation will perform, but this is only an indication. The goal of a recommender system is to provide good recommendations to users, and thus the final systems should be evaluated by users. A user evaluation has been constructed to test the recommendation quality of three separate recommender systems:

- Pseudo-Tag - as described in Chapter 7;
- Hybrid - as described in Chapter 8; and
- Tag - the baseline approach used throughout previous chapters.

This user evaluation also measures the novelty of recommendations provided by each system, complementing the Discovery measure evaluated previously. Simple demographic data is also collected to help put these results into further context, and understand more about how each approach performs.

In this chapter the design of this user evaluation is first described, including what is measured and how. This is followed by a discussion of user participation, analysing the basic demographics of users that completed the evaluation. Next, the results for recommendation quality are presented, from which several unexpected observations are made. The following section investigates the reasons behind these observations, which helps to

further understand how each system behaves. The results for novelty are presented, and the balance between recombination quality and novelty is explored for each approach. Finally, this study is summarised, and some conclusions are made.

9.1 Evaluation Design

The objective of this user study is to evaluate both recommendation quality and novelty of three different approaches. To achieve this there are several important aspects that must be considered: how to make and present recommendations in such a way that it does not introduce a bias, or influence feedback; how to evaluate quality; how to evaluate novelty; and what demographic data it is desirable to collect.

9.1.1 Making and Presenting Recommendations

The user study is presented to users through a web interface, that allows users to evaluate recommendations. Each page displayed to a user consists of a query track, and a list of the top 5 recommendations made using a single recommendation method, for example, hybrid. The order that recommendations appear on screen is randomised, to ensure that users do not know the actual ordering, removing the possibility that users will simply rate recommendations by their position. It is possible to check if users are attempting this, and exclude the results from users who consistently rate according to presentation order.

There is an equal chance of each method being used to make the recommendations, and the users cannot know which method is currently being used. The query track presented is selected at random, with an equal chance that is is either from a fixed pool, or from the entire dataset. This pool of tracks contains 3 randomly selected tracks from each of the 11 genres in the dataset, and thus 33 potential queries. The reason for including such a pool is to increase the likelihood that queries will be evaluated by more than a single person, providing a strong consensus opinion of how each approach performs with a fixed set of queries. It is likely that tracks not included in this pool will only be evaluated by a single user, and thus this data is biased towards that single users opinion.

Users are allowed to evaluate as many queries as they choose. If they evaluate enough queries to exhaust the pool, then queries will always be taken from the entire dataset,

avoiding repetition. The parameters regarding which approach is being used, and the order in which recommendations are made, are always stored server side. There is no way to access this information from the HTML file that is rendered in a browser, or from cookies, thus removing the possibility that a technically minded user could use this to influence his interactions.

9.1.2 Capturing Quality

Users are asked to evaluate the quality of each of the 5 recommendations that are made. To do this, the user can listen to a 30 second sample of the track; the samples are taken to start 15 seconds before the mid-point of the track, which is typically enough to include snippets of both verse and chorus. The user can also listen to a 30 second sample of the query track.

Figure 9.1 illustrates what is shown to a user for each recommendation. The play button can be used to hear the 30 second clip, and the track title and artist are shown. The sliding bar allows user to answer the question "Given the following track, how good are these recommendations". The user can rate the recommendation on a scale of 'Very Bad' to 'Very Good'.

Figure 9.1: Evaluating Recommendation Quality

The sliding scale allows users to score each track in the range 0 to 1000, although these values are never shown to the user. This high granularity makes the slider feel seamless, and allows the user to express their opinions beyond the more common 5 point scale, or like/dislike. The slider was chosen as opposed to a radio-button based scale to allow greater flexibility and expression to each user. When a 5 point scale is used then users may chose every point across a set of 5 recommendations, as a mechanism of ordering the results. However, a user may also be of the opinion that every track is a good recommendation, just to a greater or lesser extent than the others. This method of input allows the user to express both preferred order and quality.

To avoid influencing either positive or negative feedback, each slider is set to an initial

point of 500, exactly midway. This consistent setting also makes it easy to detect queries for which the user did not move the slider for any recommendations, but instead used the submit button to skip the track.

9.1.3 Capturing Novelty

For both the query track and each recommendation, the user is asked to declare what they know about the track. They are given three possible options.

1. Know Artist and Track
2. Know Artist but not Track
3. Do not know Artist or Track

This information helps to identify which queries and recommendations the users know, and which they do not. Based on this, it is possible to understand the proportion of tracks each approach recommends that are novel.

9.1.4 Demographic Questionnaire

Recommendations may be perceived differently by different user groups. To help understand if this is the case or not, and also to understand the sample of users that took part, each user is asked to complete a questionnaire prior to evaluating recommendations. The questions asked are shown in Table 9.1, and users could not continue to recommendations until they complete this questionnaire.

9.2 User Participation

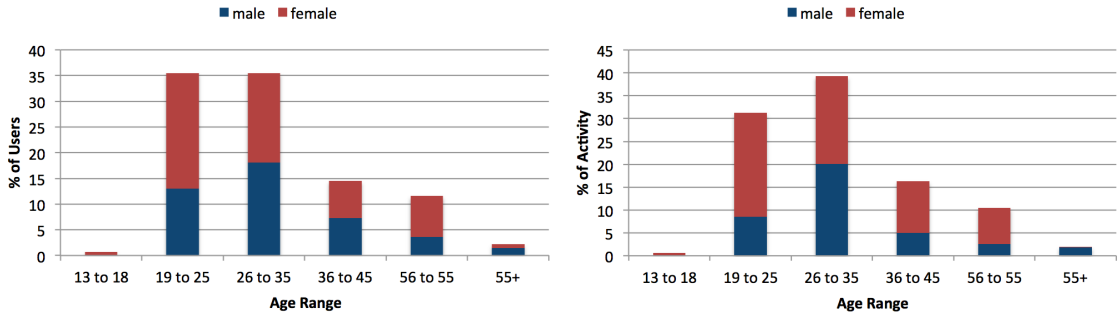
The study was published as a website, which was available for 30 days. The link to this site was distributed through social media and mailing lists. 132 people took part in the study, evaluating a total of 1444 queries. From these, 47% of queries were selected from the track pool. This is less than the 50% chance of getting a query in the pool because users were allowed to skip queries they didn't want to evaluate. The most queries that were completed by a single user is 88.

Question	Input Type	Options
Gender	Single Selection	Male Female
Age Range	Single Selection	under 13 13 - 18 19 - 25 26 - 35 36 - 45 46 - 55 55+
Musical Knowledge	Single Selection	None - no particular interest in music related topics Basic - lessons at school, read music magazine or blogs, etc Advanced - play an instrument, edit music on computer, etc Professional - professional musician, audio engineer, etc
Daily Listening Hours	Single Selection	less than 2 between 2 and 4 more than 4
Genres you typically listen to	Multiple Selection	Rock Pop Alternative Metal Electronica / Dance Folk R & B Country Rap & Hip-Hop Easy Listening Classical Holiday

Table 9.1: User Questionnaire

It was found that all recommendations for 386 queries were given a score of 500, suggesting that the users clicked 'submit' to skip the query. These queries were discarded, leaving 1058 valid query evaluations. The most valid queries evaluated by any single user is 29, and on average each user evaluated 6.24 valid queries.

Figure 9.2a shows the age range of users that took part in the study, split by genre. This reveals that 70% of users were between 19 and 35 years old, 57% of users were female, and 43% were male.



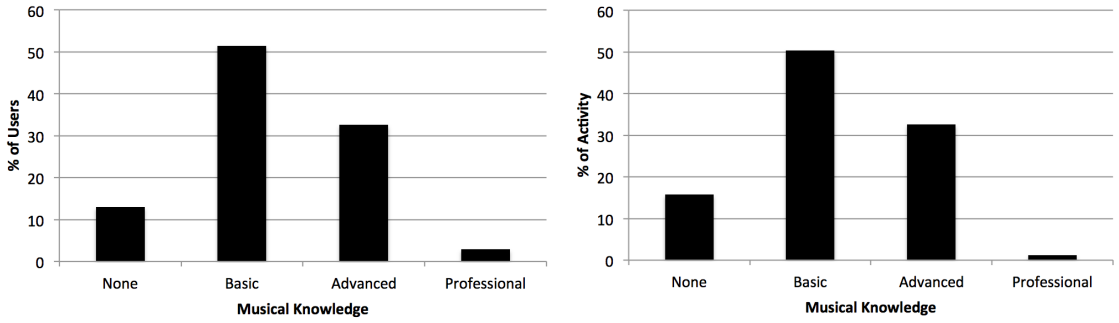
(a) Users (b) Activity

Figure 9.2: Age Range by Gender

Figure 9.2b shows user activity in terms of age range and gender. The levels of activity for users outwith the 19 to 35 age range matches what is expected based on Figure 9.2a. Users within the range 19 to 25, especially males, were less likely than others to complete several queries. Female users within the age range 26 to 35 are those who engaged most with the study. These results show a reasonable spread across both age and gender of participants, and that results are not strongly biased towards any single user group.

Figures 9.3a and 9.3b show the percentage of users and activity respectively split by musical knowledge. Unlike age and gender, there does not seem to be any correlation between musical knowledge and engagement with the user study. Approximately half of all users have a basic knowledge of music, for example reading music magazines and blogs. Around a third of users have advanced musical knowledge, meaning that they play an instrument. This shows that the users evaluating the system are knowledgeable, and thus are those likely to be familiar with, and use, music recommender systems.

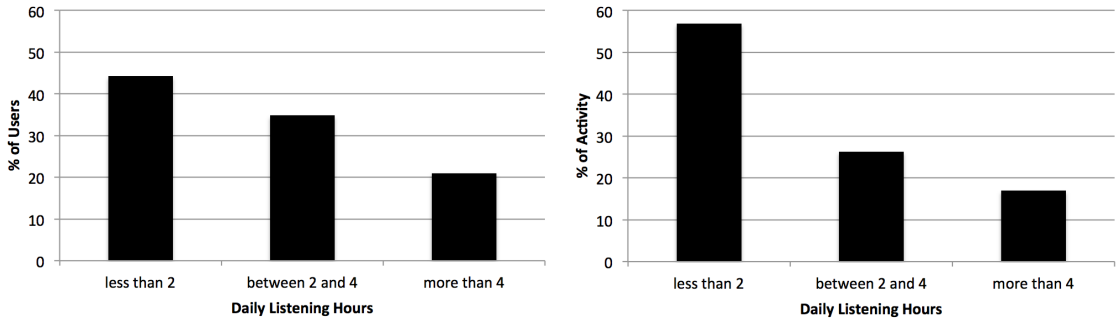
Figure 9.4 shows the number of hours each user spends listening to music for each day, and activity by these listening hours. 44% of users typically listen to less than 2 hours,



(a) Users (b) Activity

Figure 9.3: Musical Knowledge

and just over a third listen to between 2 and 4. The most active users are those who listen to less than 2 hours, who contribute over 55% of all evaluation data. This is somewhat surprising, since one would expect those who listen to the most music to be most active.



(a) Users (b) Activity

Figure 9.4: Daily Listening Hours

The final piece of information provided by users is the genres that they typically listen to. These genres are illustrated in Figure 9.5, and as may be expected, rock, pop, and alternative are the most popular genres. After these three, the next most popular genres were dance and classical, with almost 40% of users saying that they like each. This distribution of genres that users like is encouraging, since it closely matches the distribution of genres of tracks throughout the collection.

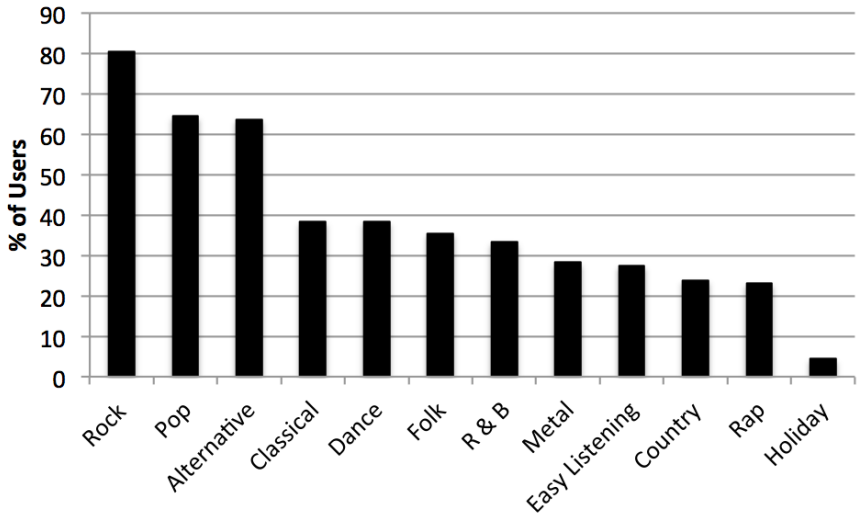


Figure 9.5: Genres Users Listen To

9.3 Ground Truth Comparison Results

The data obtained from users during the study allows for a detailed evaluation of Association Score, and investigation into how predictive of user feedback this is in practice. The further ground truth scores of genreScore, yearScore, and culturalSim (see §4.1.5) will also be examined to allow comparison between these differing approaches to ground truth. The goal of each score is to indicate how users will rate recommendations, and this is achieved by assigning query-recommendation pairs scores.

To compare ground truth with user feedback, the mean average score (MAS) of each query is calculated as

$$\text{MAS}(Q) = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{K} \sum_{k=1}^{K} \text{score}(q, R_k) \tag{9.1}$$

where Q is an individual query track, and $q \in Q$ is an instance of this query being evaluated by a single user. K is the number of recommendations, which is fixed to 5. R_k is the k^{th} recommendation for query instance q, and $\text{score}(q, R_k)$ is the score that an individual user provided for query q and recommendation R_k. Only queries where $|Q| \geq 3$ are included, i.e. queries that have been evaluated by 3 or more users. This score is also calculated for association, genreScore, culturalSim, and yearScore of tracks Q, where $|Q| = 1$ since there is only a single score that could be assigned by each ground truth.

9.3.1 Mean Average Score by Recommendation Method

Figure 9.6 shows the MAS for each ground truth score, averaged across all queries. The bars show the MAS for each of hybrid, tag, and pseudo-tag. The leftmost three bars show the mean average userScore, which is the scores provided by users during the evaluation. These results show that hybrid achieves the highest score, followed closely by tag, and pseudo-tag achieves the lowest score. The aim of each of the ground truth scores is to indicate these results observed by userScore.

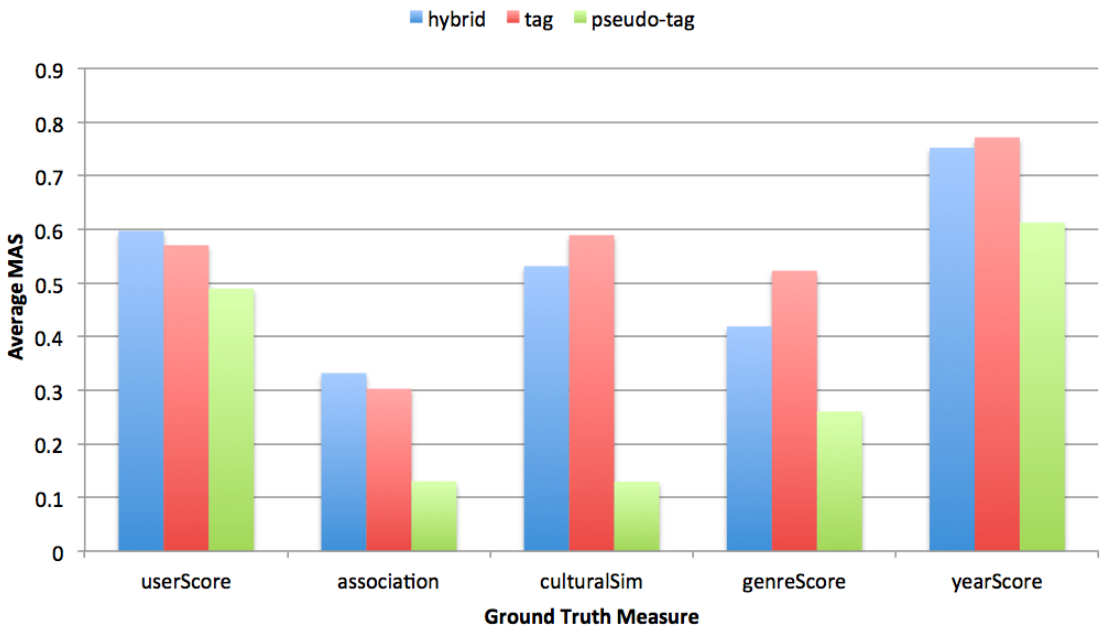


Figure 9.6: Comparison of Ground Truth Measure Scores

The results for association score show that the measure is able to accurately indicate the order of quality for each recommendation method. The exact values achieved are not necessarily important in this case, it is the interpretation of these values that is important. Association score correctly shows that hybrid performs best overall, followed by tag and then pseudo-tag. Each of the culturalSim, genreScore, and yearScore methods correctly predict that hybrid and tag outperform pseudo-tag. However, in each case tag is predicted to outperform hybrid, which is incorrect. When used as a ground truth score, this mis-prediction can have significant implications. Hybrid is a new approach, and it is crucially important to know if it offers improvements over the other methods. If these scores were

used, then hybrid would have been developed differently, and may not have improved recommendation quality from the user perspectives.

The culturalSim method is most similar to association score, in that it employs user activity as a source of knowledge. The difference between hybrid and tag is important, but culturalSim has hugely under-estimated the performance of pseudo-tag. The reason for this lies in the normalisation term of culturalSim, in Eq. (4.3). This term normalises based on the number of users who have liked each track. When this number of users is very similar, regardless of popularity, then a weighting of close to 1 is calculated. When the number of users is very different, then the weighting cost can approach 0. The effect is that culturalSim penalised recommending a popular track from a rare query, or a rare track from a popular query. In practice however the user evaluation shows this is not a good strategy to adopt.

9.3.2 Ground Truth Correlation with User Score

Figure 9.6 shows the average MAS score of each recommendation method. To gain further insight into how each of these scores correlates with the user evaluation, the MAS of each individual query for all methods is shown for association score in Figure 9.7. The line shown is the line-of-best-fit through all queries.

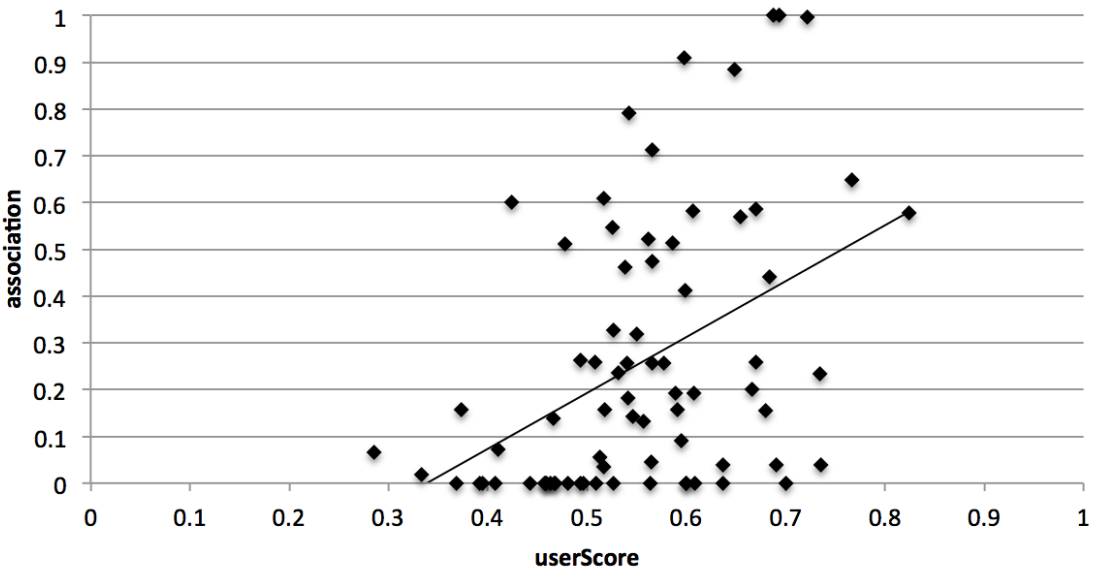


Figure 9.7: Comparison of User Score and Association Score

The range of values observed for association score is between 0 and 1, whereas the user scores are between approximately 0.3 and 0.8. This shows that the user average is likely to rate more modestly than the ground truth, and will not typically declare a recommendation to be extremely bad or good. Within these ranges however the line-of-best-fit shows that there is a positive relationship between the userScore and association. However, the association scores observed are spread widely around this line, showing that the individual query-recommendation scores are not reliable. From Figure 9.6 though the average of these scores provides a strong overall estimate. It is important therefore that evaluations include and average over a substantial number of queries.

Figure 9.8 shows the same results for culturalSim. Note that the scores here are much lower than in Figure 9.6 because the average scores in the previous figure were scaled up to make the figure readable. In this case many of the scores are extremely low, in part due to the normalisation term of culturalSim. Scores below the line-of-best-fit are quite close to this line, whereas scores above this line consist of a very few outliers. These outliers are tracks which are indeed good recommendations, but also share the same level of popularity. In this case culturalSim is allowing the pairs to influence the overall score too much, thus explaining further why the measure does not produce indicative average results.

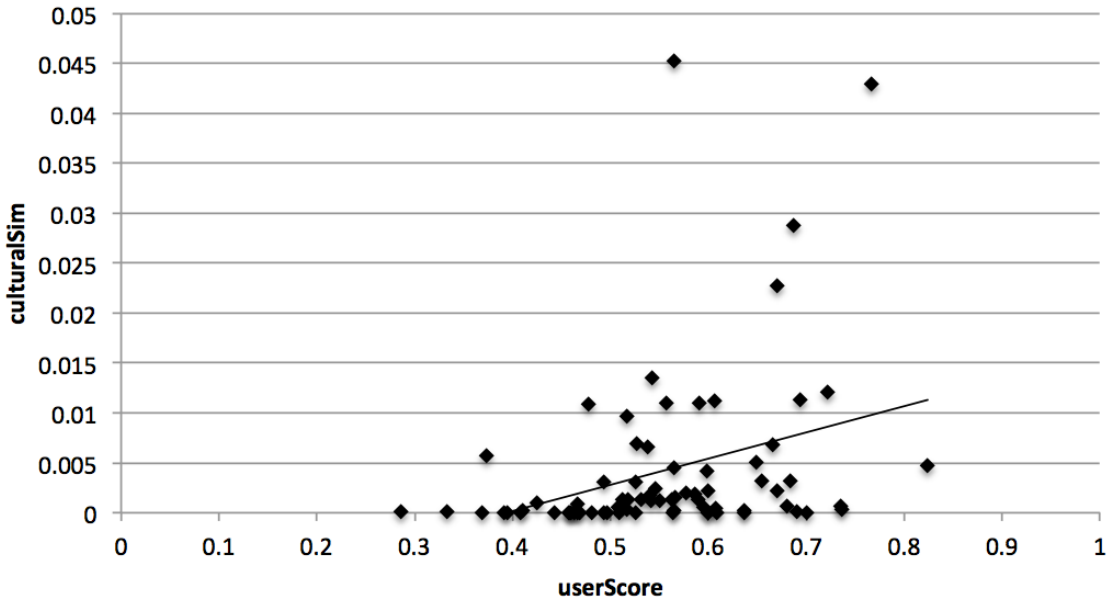


Figure 9.8: Comparison of User Score and culturalSim

9.3.3 Coefficient of Determination

The R^2 coefficient of determination can provide further insight into how each ground truth method is able to predict the user scores that will be observed. R^2 is calculated as

$$R^2 = 1 - \frac{\sum_{j=1}^{Q}(\text{userScore}_j - \text{MAS}_j)^2}{\sum_{j=1}^{Q}(\text{userScore}_j - \overline{\text{userScore}})^2} \tag{9.2}$$

where Q is the number of queries that are evaluated, userScore_j is the observed userScore for query j, MAS_j is the ground truth score associated with query j, and $\overline{\text{userScore}}$ is the average userScore for all j. The numerator calculates the residual sum of squares, and the denominator calculates the total sum of squares. Thus, a value of 1 indicates that the ground truth data perfectly fits the userScore, and a value of 0 indicates that there is no correlation between the scores.

Figure 9.9 shows this R^2 coefficient of determination for each of the ground truth scores evaluated. Association is the score most correlated with the user feedback, achieving an R^2 value of 0.19. As expected, this is followed by culturalSim, with a value of approximately 0.11. This reduced degree of correlation observed for culturalSim is in part due to the outliers observed in Figure 9.8, where the residual sum of squares will be large. Nevertheless, this lack of correlation with userScore ultimately detracts from culturalSim's ability to indicate how users will rate recommendations.

The R^2 values observed for genreScore and yearScore are perhaps more unexpected, both achieving a score of approximately 0.04. Such a small value indicates that there is very little correlation between the scores predicted by the scores and userScore. The reason that genreScore does not correlate with userScore is that, even when averaged over 5 recommendations, approximately 50% the queries are given a score of either 0 or 1. In §4.1 it was argued that granularity is an important property of any ground truth evaluation score, which genreScore is unable to introduce. Interestingly, the complete opposite effect is observed with yearScore, where the distribution of values observed is flat. This is the same distribution that one would observe from a random ground truth, which yearScore is only marginally better than.

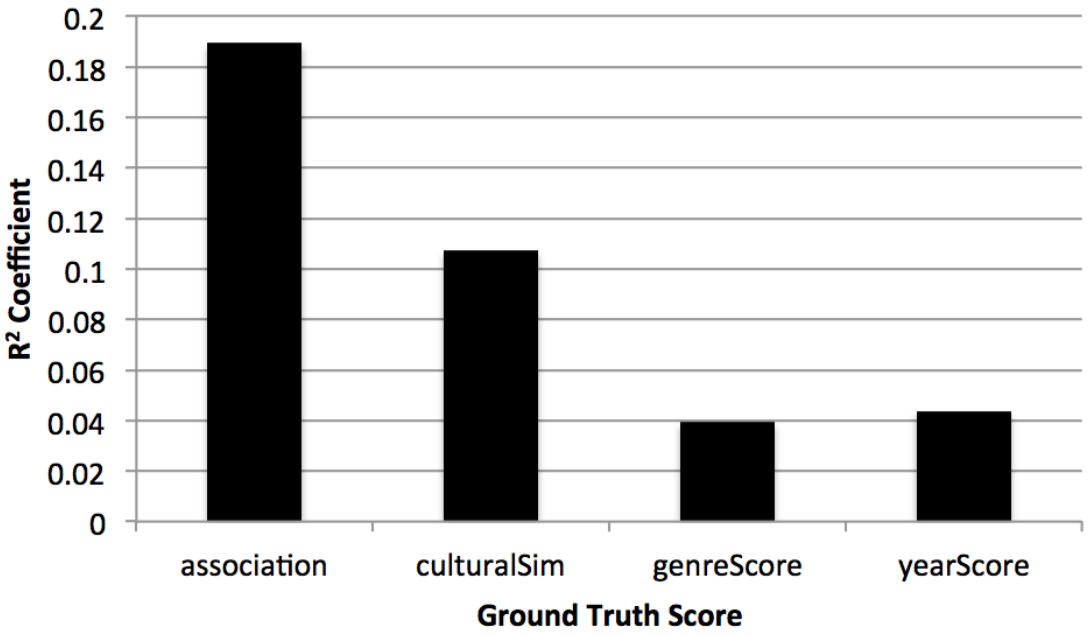


Figure 9.9: R^2 of User Score and Ground Truth Scores

9.4 Recommendation Quality Results

It has been shown that association score correlates strongly with the scores that users provided during the user study. In this section these user scores will be examined by themselves, allowing a more detailed analysis of how each method performs according to user opinion.

Figure 9.10 shows the average MAS across all queries for each recommendation method, including those that were evaluated by less than 3 users. The horizontal axis shows the number of recommendations made, and the vertical axis shows the average MAS score for this number of recommendations made. For example, scores at 3 show the average score of the first 3 recommendations.

The solid black line shows the quality of recommendations made using the hybrid representation. These results are roughly flat across the 5 recommendations made, with a score of around 580. The dashed black line shows the quality of recommendations made using the tag approach. The quality of tags is around 575, a small margin lower than hybrid. However, there is no significant difference between these two methods. The solid grey line shows the performance of pseudo-tag, which as expected is much lower than both

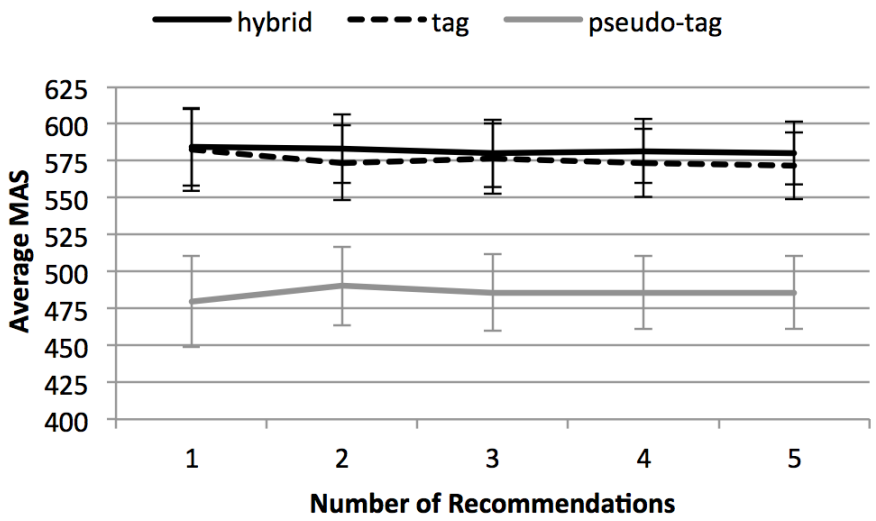


Figure 9.10: Recommendation Quality For All Tracks

tag and hybrid.

9.4.1 Pool Recommendation Quality

Figure 9.10 shows recommendation quality for all tracks that were evaluated, of which approximately 53% were not tracks from the pre-determined pool. For these non-pool tracks, each track is typically evaluated only by a single user. The opinions of one user do not necessarily match the opinions of other users, and therefore it is interesting to examine the results when only tracks in the pool are measured. Figure 9.11 shows the average MAS for queries which were part of the track pool, and therefore each is evaluated by more than 1 user.

As observed previously, hybrid performs better than tags, but the difference is not significant. However, the recommendation quality of hybrid is stronger than previously, and the quality of tags is weaker. For the first recommendation, there is a difference of 29 between hybrid and tags, compared with a difference of only 2 previously in Figure 9.10.

The performance of pseudo-tags for pool-only tracks is almost identical to the performance of pseudo-tags across all tracks. This observation is interesting because it raises the question of why the performance of hybrid and tag is different, but pseudo-tags is unchanged. One may expect a similar degree of variation across all methods, but this is not the case. This observation hints that there must be something beyond simply the

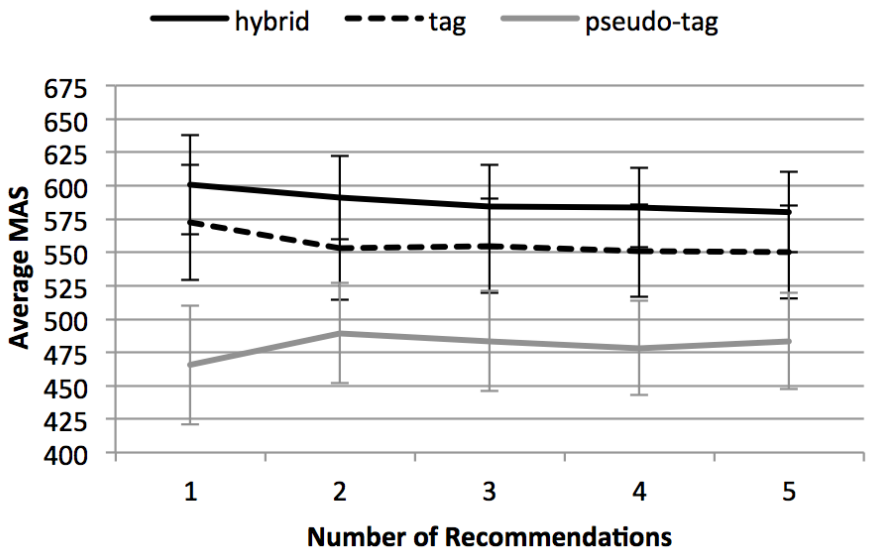


Figure 9.11: Recommendation Quality For Pool Tracks

recommendation approach that is influencing the results observed, and explain why hybrid improves and tags are weakened.

9.4.2 Evidence of Bias Within Results

To help further understand the results observed in Figure 9.11 compared with those in Figure 9.10 the other data that was collected can be used. This data includes answers from the questionnaire, and data on what tracks each user knows. Demographic data was analysed first, to see if there was any correlations between age and gender for example, and how users score recommendations. However, no such correlations were found, since queries were generated randomly, and thus well distributed across each demographic.

Figure 9.12 shows the recommendation quality for each approach, split by what the user knows about each recommended track. The first section along the horizontal axis shows recommendation quality for tracks when the user knew both the artist and the track that was recommended. In this case tag performs strongest, but the performance of hybrid is comparable. As expected pseudo-tag is significantly weaker than either tag or hybrid.

The second section of the horizontal axis shows quality for recommendations where the track was not known, but the artist was. For these novel recommendations hybrid provides

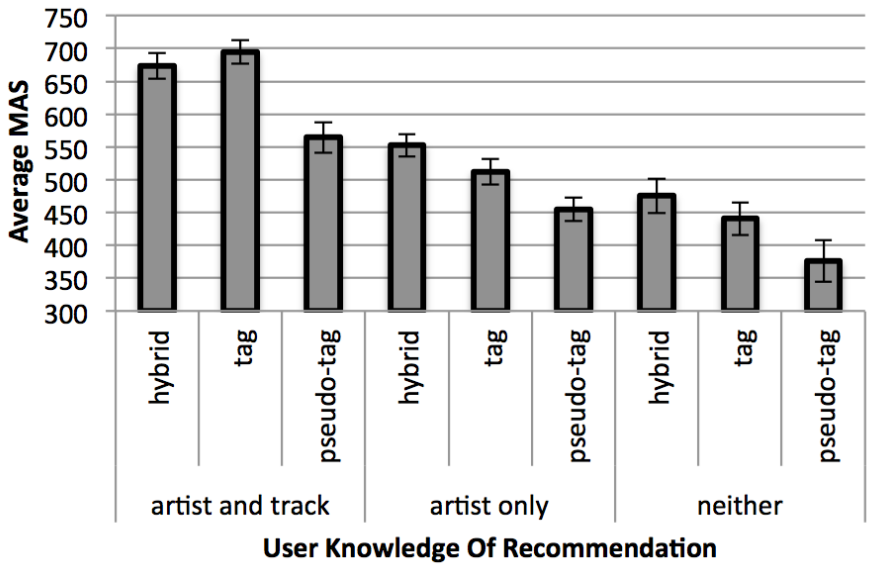


Figure 9.12: Recommendation Quality by What is Known For All Tracks

significantly better recommendations than both tag and pseudo-tag, and tag also provides a significant improvement over pseudo-tag. The final section shows quality when neither artist nor track are known to the user. Similarly to when the track is not known, hybrid performs best. However, the improvement over tag is not significant. The improvement of both hybrid and tag over pseudo-tag is significant.

Looking at the relative performance of each method by these groups reveals a trend towards giving higher scores to recommendations where the track is known. Also, users generally give higher scores to unknown tracks when they know the artist that recorded the track. It makes sense that users will provide higher scores to tracks that they already know, since for many people these will indeed be viewed as a good recommendation.

9.5 Novelty Results

Novelty is the ability of each recommendation method to include unknown tracks in recommendations. This is measured as the rate at which unknown recommendations are made, when the query track is known to the user. Figure 9.13 shows novelty for each method over 5 recommendations. The format of this figure is the same as previously, but the vertical axis now shows the rate of including novel tracks. For example, a score of 0.5

indicates that 50% of recommendations made are tracks that the user does not already know.

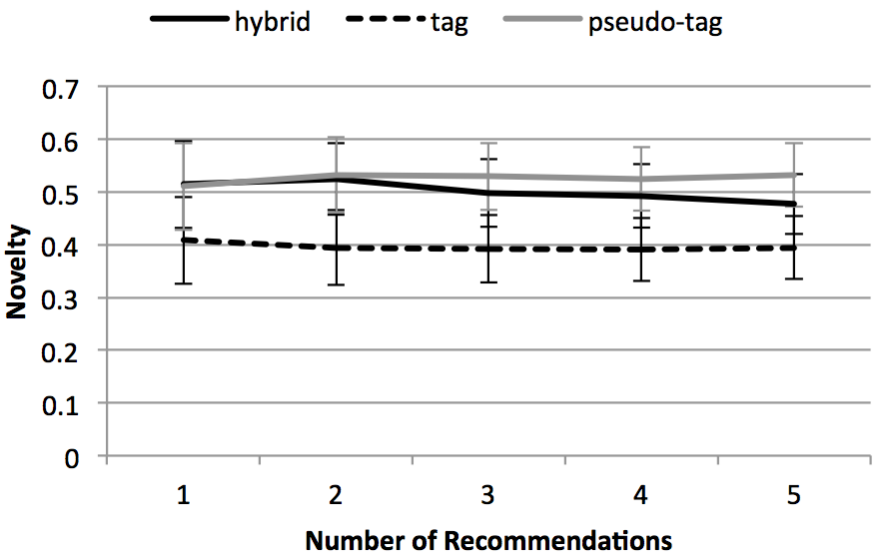


Figure 9.13: Novelty

Hybrid, illustrated by the solid black line, recommends novel tracks around 50% of the time. Pseudo-tag produces a greater amount of novel recommendations, but the difference is not significant. Recommendations made using tag are only novel around 40% of the time, a significantly lower rate than both hybrid and pseudo-tag. The reason that tag makes fewer novel recommendations is because it tends to recommend well-tagged tracks. These tags are created by users of the Last.FM system, and thus well-tagged tracks also tend to be well known.

9.5.1 Balance Between Quality and Novelty

An increased novelty rate is good for cold-start recommendation, allowing new tracks to be recommended to users. However, these novel recommendations must be of comparable quality to tracks that are already known to users. Ultimately, if users do not like the tracks being recommended, regardless of whether they are known or not, then they will not continue to use the system. Results in Figure 9.12 indicate that hybrid provides better recommendations than tags for unknown tracks, and Figure 9.13 shows that hybrid makes more novel recommendations than tags.

Figure 9.14 shows the relationship between the quality, as average MAS on the horizontal axis, and the novelty of recommendations on the vertical axis. Each point shows the average scores over different numbers of recommendations. The right-most point for each approach is when 1 recommendation is made, and the left-most is the average over 5 recommendations.

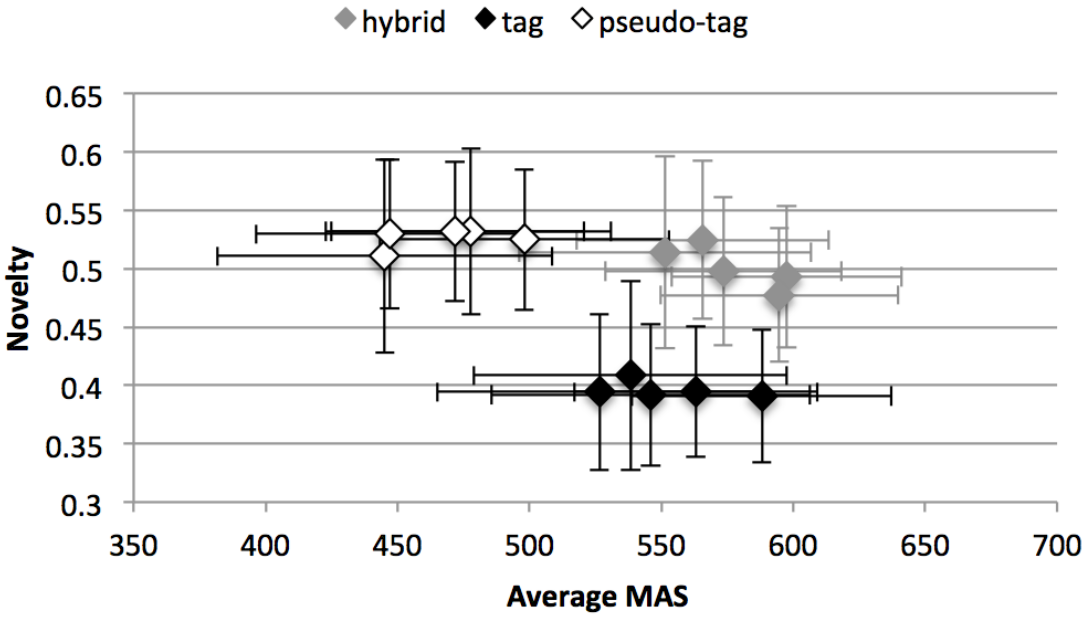


Figure 9.14: Balance Between Quality and Novelty

Hybrid and tag both achieve comparable recommendation quality of between 525 and 600. However, in terms of novelty, hybrid offers a significant improvement over tag-based recommendation. This figure clearly illustrates that hybrid is able to take the strongest properties of each of its constituent parts, and exploit these properties for recommendation. Tags provide good recommendation quality, but low novelty, and conversely pseudo-tags provide good novelty, but low recommendation quality. Through dynamically combining both tags and pseudo-tags in a meaningful way, hybrid is able to offer a good novelty rate, while maintaining strong recommendation quality.

These results show that the hybrid representation is able to effectively bridge the semantic gap between content, through pseudo-tags, and tags. When used as part of a recommender system, this hybrid representation allows users to discover more new music, that importantly, is of high quality, and likely to be of interest.

9.6 Summary

A user study was conducted that evaluates both recommendation quality and novelty of the hybrid, tag, and pseudo-tag representations. In this study, users were first asked to complete a short questionnaire to provide background demographic information, and help understand the user sample. After this, each user was presented with a series of queries, and for each query asked to evaluate 5 recommendations made using a single recommendation approach.

A high level of participation and engagement was achieved during the 30 days that the evaluation was available online. 132 users took part, and collectively evaluated over 1000 queries. The questionnaire data collected showed a good spread of users across both age and gender, and the musical interests of the user base aligns closely with the tracks contained in the dataset.

Previously, each method had been evaluated using association score, developed in Chapter 4. This user study allowed the comparison of association score and other ground truth scores, with the scores that user provided to be made. Results show that association score was the only method which could accurately indicate the relative performance of three separate recommendation methods. This is important, since if other methods were used, then incorrect refinements to new recommendation methods would have been made. The primary reasons that association score was able to out-perform the other ground truth scores is that it is not biassed by track popularity, and that it is able to provide a suitable level of granularity of scores.

User results show that hybrid and tag both achieve a comparably high level of recommendation quality, with an average score of approximately 600 on a 1000 point scale. As expected the pseudo-tag representation achieved lower recommendation quality than both tag and hybrid, but provides a good baseline for comparison. The evaluation also included queries that were included from a fixed pool of tracks, allowing a consensus opinion of the quality of each query to be understood. The results for the pool tracks shows the quality of hybrid improving, and the quality of tag decreasing. This highlighted the need to further investigate the results, and understand why this change is observed.

Further results show that tag performs well when recommending known tracks, but

is weaker than hybrid when recommending novel tracks. Hybrid representation does not suffer from cold-start, and so has a better understanding of each track that could be recommended. This in turn allows hybrid to make higher quality novel recommendations. Results looking at the rate of making novel recommendations show that 50% of tracks recommended by hybrid are unknown, compared with 40% by tag. This increased novelty rate is related to cold-start, since cold-start tracks are likely to also be unknown to users.

Through increasing cold-start discovery rate, hybrid representation allows more novel recommendations to be made at the same level of quality as a tag-based recommender system. This type of recommender system allows users to discover more novel tracks, and widen their musical tastes, in a way that maintains high quality. As these tracks are recommended it becomes increasing likely that users will tag the tracks, thus reducing the level of cold-start throughout the collection. Of course, new tracks are always being released, but this recommender system would be able to more quickly introduce these to relevant users, and shorten the time that tracks remain cold-start.

Chapter 10

Conclusions and Future Work

The work presented in this thesis has investigated approaches to describing musical texture, developed pseudo-tags to bridge the semantic gap, and described methods to integrate content and semantic representations for music recommendation. In this concluding chapter, a discussion of the contributions that have been made, and future directions of this research is presented.

10.1 Achievements and Contributions

The achievements and contributions of this work can be best described by revisiting the initial research objectives. In this section the achievements and contributions that have emerged from each research objective are discussed in turn.

Analyse the advantages and disadvantages of both content- and semantic-based representations for making music recommendations

This research examines some of the strengths and weaknesses of both content and semantic representations. Through experiments this work highlights the strengths of content representations, which are able to offer high levels of discovery in recommendations. However, the same experiments also show that the quality of content is weak, in part because it is not suitably constructed for describing music.

Analysis of semantic representations, and the process by which these representations

are generated, highlights the strength of tags for describing music. Tags are able to capture high level musical and cultural meaning suitably, and from this knowledge strong recommendations may be made. However, analysis of real world data also shows that these tag representations are lacking, and many tracks suffer from the cold-start problem.

This analysis of both content- and semantic-based representations motivates the direction of the further research objectives. The focus of the work is on reducing cold-start within collections, and increasing the discovery of high quality recommendations.

Develop novel content-based representations, which take advantage of the perceptual qualities of music that make good recommendations

This research focusses on the importance of modelling texture relating to music, rather than using texture representations from the field of speech recognition. A critical analysis of the difference, and what is required for music, allows the contribution of a new music-inspired texture representation, MFS, which was presented at the AAAI conference (Horsburgh, Craw & Massie 2012b). Experimental results show the benefits of using this MFS representation for recommendation, and highlight how the representation may be used within different recommendation models. This contribution allows music researchers the ability to describe the texture of music more meaningfully, and this texture may have many more uses beyond recommendation.

Bridge the semantic gap by integrating content and semantic representations, and use this bridge to improve recommendations

The development of a generalised concatenated representation, integrating both content and tags, contributes an approach to bridging the semantic gap. Previous approaches focus on integrating clustered content representations and tags. The approach presented in this research shows how content may be integrated without clustering, allowing concepts to be learned for each part of content representation. The advantage in doing so is that correlations between parts of a musical sound and tags may be established, not just correlations with the overall musical sound.

This method also contributes a new approach to increasing the discovery rate of tracks that suffer from the cold-start problem. The concepts learned contain information about

both content and tags, and when a track with no tags is projected into this concept space, the concepts it is associated with introduce new tag knowledge to the track. This work was presented at the IJCAI conference (Horsburgh et al. 2011).

The development of pseudo-tags contributes both a new approach to auto-tagging, and a new method by which to achieve this. Many auto-tagging approaches are built upon modelling the distributions observed for track content, and for track tags. Using these learned distributions new tags are propagated to tracks. An investigation of such approaches however realises the short comings of such an approach when the cold-start problem is present. A new approach to tackling this problem is developed, where it the distributions of content and tags are not assumed. Instead, content was used as a black box to retrieve similar tracks, and then from these similar tracks to learn pseudo-tags.

The main contribution of this approach is that it allows tags that are not directly related to content to be learned. This is an important step towards bridging the semantic gap, and towards capturing both musical and non-musical semantic information. This approach also allows for a guaranteed number of tags to be learned for a track, which importantly are each given a weight. This is different from auto-tagging, which simply learns tags without a weight as a classification task. Results show that this pseudo-tag representation offers improved quality over content representation, and provides a step towards the closer integration of content and semantic representations. Initial work towards this approach was presented at the Third Annual Digital Economy All Hands Conference (Horsburgh, Craw & Massie 2012a).

The hybrid representation presented in Chapter 8 achieves a bridge between content and tags based on the mid-level pseudo-tag representation. This contributes a new approach to bridging the semantic gap, where content and tags are not integrated directly, but through an intermediate step. Pseudo-tags are integrated into a tag representation dynamically, where both the number and weight of pseudo-tags that contribute to the hybrid representation are parameters. This is an important step forward from bridging between content and tags directly, and creates the idea that a bridge should be dynamic, and only used when appropriate. Results from the offline and user study show that this approach is able to increase the discovery rate of cold-start tracks, and therefore novelty, while also maintaining high recommendation quality.

Exploit user feedback data to evaluate the important qualities of music recommendation

The evaluation of music recommender systems is an important part of developing new methods for recommendation. Investigation of existing evaluation strategies highlighted that strong evaluations are difficult to achieve without repeatedly conducting user trials. This investigation highlighted that evaluation measures suffer from lack of complete data, and are biased by popularity in listening habits. The focus of this research is to examine discovery, which typically includes lesser known tracks, and so such evaluation methods were not suitable.

A contribution of this research is a new approach to evaluation recommendation quality, association score. The data required for this score can be gathered for any working collection, and through the novel use of both implicit and explicit feedback, the bias of popularity is hugely reduced. A user study is constructed to evaluate association score, and compares it with other existing ground truth scores. This shows that association score provides a strong indication of how users will rate recommendations. This contribution therefore allows researchers the ability to more effectively develop and optimise new techniques, based on much stronger feedback than previous ground truth measures provide.

10.2 Future Directions

This thesis focussed on the use of texture representation, and forming a bridge between texture and tag representation. The motivation for this bridge was to increase discovery of cold-start tracks, and novelty from a user perspective. In this section some of the limitations of this work are identified, and future research directions and challenges described.

10.2.1 Expanding Content Representation

Texture was identified as the strongest method of describing musical content for making recommendations. However, texture is not the only representation of content that is

available. Further information, such as chroma, rhythm, and perhaps even melody could be learned for each track. Incorporating this information into the content description of a track may allow stronger overall representation to be achieved. The introduction of further knowledge may also allow new bridges to be formed to semantic content, although these will most likely be limited to music related knowledge.

10.2.2 Expanding Tag Representation

The tag representation used throughout this research is a simple, but effective approach to describing semantic knowledge. However, this tag representation could also be further expanded, perhaps making use of natural language processing, or through linking to ontologies. Introducing this further knowledge layer to semantic representation may open up new possibilities when bridging the gap between content and semantics. For example, concepts such as synonymy, sentiment, and domain knowledge open up a more fine grained representation. Perhaps in some cases this granularity would allow a stronger bridge to be formed, for example between chroma and sentiment, based on the musical scale being used.

10.2.3 User Profiles

The recommendation approach used throughout this research is to assume that a user provides a single track that they like as an input to the recommendation process. In practice, this would typically be the track they are currently, or have most recently listened to. However, users listen to many tracks, and future work could investigate how these groups of tracks may be used to form a stronger user profile, from which to make recommendations. These user profiles would open up the possibility of looking at the semantic gap for sets of songs that are related, rather than for each song individually. This expanded query representation may allow for further discoveries to be made, and the quality of these discoveries may indeed be higher than the quality of known recommendations.

10.2.4 Applying Dynamic Hybrid to Further Representations

The dynamic hybrid representation that has been developed integrates pseudo-tags with existing tag representations. Viewed more generally however, this approach integrates a

pseudo-representation that has been learned, into a real representation that is provided. Future work may examine how this approach could be utilised with further representation, such as user profiles. Collaborative filtering methods suffer from the cold-start problem from a user's perspective, rather than the track's. If pseudo-profiles could be learned successfully, then perhaps this cold-start problem could be reduced in a similar way. Eventually, a three-way model between users profiles, content, and tags may even be established, and pseudo content may be learned across this model to enhance the knowledge described.

10.3 The Future of Music Recommendation

Music recommendation has evolved from the pre-digital age where close friends made recommendations to each other, into the digital age where anonymous artificial friends made recommendations. These artificial friends, the algorithms and data structures that produce recommendations, are becoming an increasingly more popular method for discovering new music. This anonymity has had real tangible impacts on how music recommendation occurs, and may even be dramatically changing the way that people interact with music.

Discovering new music is still as important as it always has been, but discovering new social identities and sub cultures seems to be disappearing into the past. Semantic information allows people to describe their identity, and the culture surrounding music, but this culture is increasingly becoming an online presence. What are the modern equivalents of the punks, mods, rockers, and hippies? The real and online worlds seem to be drifting further apart, and groups of people who share the same musical influences and social concerns do not appear as vocal or prominent as in the past. Perhaps this is because people now express their opinions online, or perhaps access to near unlimited access of music has resulted in many smaller sub cultures emerging. Perhaps it is enough to be part of a musical collective in an online world, and the desire to express this in the real world is diminished.

However, where does this leave live music scenes? These are the next major challenge for music recommendation. Music is recommended online as part of 'social' communities, but the real social community for music is in the venue, dancing or jumping around to the next exciting thing. Just like a bridge between content and semantics is required, the

next bridge to be built is between the online and real world. Can music recommendation be used to inspire local music scenes, introduce people to new and exciting artists on their doorstep, and allow real musical communities to re-emerge?

10.4 Conclusions

In this work the semantic gap between content and tags has been bridged through integrating the representations. Increased discovery of high quality novel tracks has been achieved through exploiting pseudo-tags in a dynamic hybrid representation. In future work this hybrid representation could be expanded to include user profiles, and the introduction of data that drives social interaction. If event listings and local artists can be included, then music recommendation may be used as the driving force behind discovering music at a more local level. Further, the recommendation task could be expanded to include event recommendation, based on a combination of information in the existing hybrid, and the inclusion of event listings, friendships, and temporal information. This work offers a strong starting point for bridging the next gap, the social gap for music.

Bibliography

Amatriain, X., Pujol, J. M. & Oliver, N. (2009). I like it... i like it not: Evaluating user ratings noise in recommender systems, *User Modeling, Adaptation, and Personalization*, Springer, pp. 247–258.

Amatriain, X., Pujol, J. M., Tintarev, N. & Oliver, N. (2009). Rate it again: increasing recommendation accuracy by user re-rating, *Proceedings of the third ACM conference on Recommender systems*, ACM, pp. 173–180.

Aucouturier, J., Pachet, F. & Sandler, M. (2005). The way it sounds: Timbre models for analysis and retrieval of music signals, *IEEE Transactions on Multimedia* **7**(6): 1028–1035.

Balabanović, M. & Shoham, Y. (1997). Fab: content-based, collaborative recommendation, *Communications of the ACM* **40**(3): 66–72.

Barragáns-Martínez, A. B., Costa-Montenegro, E., Burguillo, J. C., Rey-López, M., Mikic-Fonte, F. A. & Peleteiro, A. (2010). A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition, *Information Sciences* **180**(22): 4290–4311.

Barrington, L., Turnbull, D. & Lanckriet, G. (2008). Auto-tagging music content with semantic multinomials, *Proceedings of International Conference on Music Information Retrieval*, pp. 297–302.

Bartsch, M. A. & Wakefield, G. H. (2001). To catch a chorus: Using chroma-based representations for audio thumbnailing, *IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, IEEE, pp. 15–18.

Benetos, E. & Dixon, S. (2013). Multiple-instrument polyphonic music transcription using a temporally constrained shift-invariant model, *The Journal of the Acoustical Society of America* **133**: 1727–1741.

Bertin-Mahieux, T., Eck, D. & Mandel, M. (2010). Automatic tagging of audio: The state-of-the-art, *Machine Audition: Principles, Algorithms and Systems* pp. 334–352.

Bertin-Mahieux, T., Ellis, D. P., Whitman, B. & Lamere, P. (2011). The million song dataset, *Proceedings of the 12th International Society for Music Information Retrieval Conference*, University of Miami, pp. 591–596.

Bischoff, K., Firan, C. S., Nejdl, W. & Paiu, R. (2008). Can all tags be used for search?, *Proceeding of the 17th ACM conference on Information and knowledge management*, ACM, New York, NY, USA, pp. 193–202.

Bogdanov, D., Haro, M., Fuhrmann, F., Gómez, E. & Herrera, P. (2010). Content-based music recommendation based on user preference examples, *1st Workshop On Music Recommendation And Discovery (WOMRAD), ACM RecSys, 2010, Barcelona, Spain.*

Bogdanov, D. & Herrera, P. (2011). How much metadata do we need in music recommendation? a subjective evaluation using preference sets, *Proceedings of 2011 International Society for Music Information Retrieval Conference*, pp. 97–102.

Breese, J. S., Heckerman, D. & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering, *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, Morgan Kaufmann Publishers Inc., pp. 43–52.

Bu, J., Tan, S., Chen, C., Wang, C., Wu, H., Zhang, L. & He, X. (2010). Music recommendation by unified hypergraph: combining social media information and music content, *Proceedings of the international conference on Multimedia*, MM '10, ACM, New York, NY, USA, pp. 391–400.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments, *User modeling and user-adapted interaction* **12**(4): 331–370.

Cai, R., Zhang, C., Zhang, L. & Ma, W.-Y. (2007). Scalable music recommendation by search, *Proceedings of the 15th international conference on Multimedia*, ACM, pp. 1065–1074.

Casey, M. A., Veltkamp, R., Goto, M., Leman, M., Rhodes, C. & Slaney, M. (2008). Content-based music information retrieval: Current directions and future challenges, *Proceedings of the IEEE* **96**(4): 686–696.

Celma, O. (2010). *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*, Springer-Verlag.

Celma, O. & Cano, P. (2008). From hits to niches?: or how popular artists can bias music recommendation and discovery, *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, ACM, pp. 1–8.

Celma, Ò. & Herrera, P. (2008). A new approach to evaluating novel recommendations, *Proceedings of the 2008 ACM conference on Recommender systems*, ACM, pp. 179–186.

Crossen, A. & Budzik, J. (2006). Promoting social interaction in public spaces: The flytrap active environment, *Consuming Music Together*, Springer, pp. 111–128.

Crossen, A., Budzik, J. & Hammond, K. J. (2002). Flytrap: intelligent group music recommendation, *Proceedings of the 7th international conference on Intelligent user interfaces*, ACM, pp. 184–185.

Degara, N., Pena, A., Davies, M. E. & Plumbley, M. D. (2010). Note onset detection using rhythmic structure, *IEEE International Conference on Acoustics Speech and Signal Processing*, IEEE, pp. 5526–5529.

Dörfler, M. (2001). Time-frequency analysis for music signals: A mathematical approach, *Journal of New Music Research* **30**(1): 3–12.

Eck, D., Lamere, P., Bertin-Mahieux, T. & Green, S. (2007). Automatic generation of social tags for music recommendation, *Advances in neural information processing systems* **20**: 385–392.

Ellis, A. (1885). On the musical scales of various nations, *Journal of the Society of Arts* pp. 485–527.

Ellis, D., Whitman, B., Berenzweig, A. & Lawrence, S. (2002). The quest for ground truth in musical artist similarity, *Proc. International Symposium on Music Information Retrieval ISMIR-2002*, pp. 170–177.

Evans, B. G. & Iverson, P. (2004). Vowel normalization for accent: An investigation of best exemplar locations in northern and southern British English sentences, *The Journal of the Acoustical Society of America* **115**: 352.

Firan, C., Nejdl, W. & Paiu, R. (2007). The benefit of using tag-based profiles, *Web Conference, 2007. LA-WEB 2007. Latin American*, pp. 32 –41.

Gjerdingen, R. O. & Perrott, D. (2008). Scanning the dial: The rapid recognition of music genres, *Journal of New Music Research* **37**(2): 93–100.

Grimaldi, M. & Cunningham, P. (2004). Experimenting with music taste prediction by user profiling, *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, ACM, pp. 173–180.

Hauser, M. D. (2000). The sound and the fury: Primate vocalizations as reflections of emotion and thought, *The Origins of Music. The MIT Press, Cambridge, MA, USA* pp. 77–102.

Hershey, J. & Olsen, P. (2007). Approximating the Kullback Leibler divergence between Gaussian mixture models, *IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. IV, pp. 317–320.

Hoashi, K., Matsumoto, K. & Inoue, N. (2003). Personalization of user profiles for content-based music retrieval based on relevance feedback, *Proceedings of the eleventh ACM international conference on Multimedia*, ACM, pp. 110–119.

Hoffman, M., Blei, D. & Cook, P. (2009). Easy as CBA: A simple probabilistic model for tagging music, *Proceedings of the International Symposium on Music Information Retrieval*, pp. 369–374.

Horsburgh, B., Craw, S. & Massie, S. (2012a). Cold-start music recommendation using a hybrid representation, *Digital Economy All Hands Conference*.

Horsburgh, B., Craw, S. & Massie, S. (2012b). Music-inspired texture representation, *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 52–58.

Horsburgh, B., Craw, S., Massie, S. & Boswell, R. (2011). Finding the hidden gems: Recommending untagged music, *Twenty-Second International Joint Conference on Artificial Intelligence*, pp. 2256–2261.

Hu, B., Guo, M. & Zhang, H. (2009). A hybrid music recommendation system by M-LSA, *Computational Intelligence and Natural Computing, 2009. CINC'09. International Conference on*, Vol. 1, IEEE, pp. 129–132.

Itoyama, K., Ogata, T. & Okuno, H. G. (2012). Automatic chord recognition based on probabilistic integration of acoustic features, bass sounds, and chord transition, *Advanced Research in Applied Artificial Intelligence*, Springer, pp. 58–67.

Jawaheer, G., Szomszor, M. & Kostkova, P. (2010). Comparison of implicit and explicit feedback from an online music recommendation service, *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, ACM, pp. 47–51.

Kim, J., Tomasik, B. & Turnbull, D. (2009). Using artist similarity to propagate semantic information, *Proceedings of the 10th International Conference on Music Information Retrieval*.

Knees, P., Pohle, T., Schedl, M., Schnitzer, D. & Seyerlehner, K. (2008). A document-centered approach to a natural language music search engine, *Advances in Information Retrieval*, Springer, pp. 627–631.

Koenigstein, N., Dror, G. & Koren, Y. (2011). Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy, *Proceedings of the fifth ACM conference on Recommender systems*, ACM, pp. 165–172.

Krothapalli, S. R. & Koolagudi, S. G. (2013). Speech emotion recognition: A review, *Emotion Recognition using Speech Features*, Springer, pp. 15–34.

Lamere, P. (2008). Social tagging and music information retrieval, *Journal of New Music Research* **37**(2): 101–114.

Last.fm dataset (2012). the official song tags and song similarity collection for the million song dataset, http://labrosa.ee.columbia.edu/millionsong/lastfm.

Law, E. & Von Ahn, L. (2009). Input-agreement: a new mechanism for collecting data using human computation games, *Proceedings of the 27th international conference on Human factors in computing systems*, pp. 1197–1206.

Lee, C.-H., Shih, J.-L., Yu, K.-M. & Lin, H.-S. (2009). Automatic music genre classification based on modulation spectral analysis of spectral and cepstral features, *Multimedia, IEEE Transactions on* **11**(4): 670 –682.

Lee, S., Cho, Y. & Kim, S. (2010). Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations, *Information Sciences* **180**(11): 2142–2155.

Levy, M. & Bosteels, K. (2010). Music recommendation and the long tail, *1st Workshop On Music Recommendation And Discovery (WOMRAD), ACM RecSys, 2010, Barcelona, Spain.*

Levy, M. & Sandler, M. (2007). A semantic space for music derived from social tags, *Proceedings of the International Conference on Music Information Retrieval*, Vol. 1, p. 12.

Levy, M. & Sandler, M. (2009). Music information retrieval using social tags and audio, *IEEE Transaction on Multimedia* **11**(3): 383–395.

Li, Q., Myaeng, S. H. & Kim, B. M. (2007). A probabilistic music recommender considering user opinions and audio features, *Information processing & management* **43**(2): 473–487.

Li, S. Z. (2000). Content-based audio classification and retrieval using the nearest feature line method, *IEEE Transactions on Speech and Audio Processing* **8**(5): 619–625.

Li, T. & Chan, A. (2011). Genre classification and the invariance of MFCC features to key and tempo, *Advances in Multimedia Modeling* pp. 317–327.

Logan, B. (2000). Mel frequency cepstral coefficients for music modelling, *International Symposium on Music Information Retrieval*, Vol. 28.

Logan, B. (2004). Music recommendation from song sets, *Proceedings of the International Conference on Music Information Retrieval*, pp. 425–428.

Logan, B. & Salomon, A. (2001). A music similarity function based on signal analysis, *IEEE International Conference on Multimedia and Expo*, pp. 745–748.

Lops, P., de Gemmis, M. & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends, *Recommender Systems Handbook*, Springer, pp. 73–105.

Mandel, M. & Ellis, D. (2008). A web-based game for collecting music metadata, *Journal of New Music Research* **37**(2): 151–165.

McFee, B., Bertin-Mahieux, T., Ellis, D. P. & Lanckriet, G. R. (2012). The million song dataset challenge, *Proceedings of the 21st international conference companion on World Wide Web*, ACM, pp. 909–916.

McKay, C., McEnnis, D. & Fujinaga, I. (2006). A large publicly accessible prototype audio database for music research, *Proceedings of the International Conference on Music Information Retrieval*, pp. 160–3.

Mermelstein, P. (1976). Distance measures for speech recognition, psychological and instrumental, *Pattern Recognition and Artificial Intelligence* **116**: 91–103.

Miotto, R., Barrington, L. & Lanckriet, G. (2010). Improving auto-tagging by modeling semantic co-occurrences, *Proceedings of the International Conference on Music Information Retrieval*, pp. 297–302.

Musto, C., Semeraro, G., Lops, P., de Gemmis, M. & Narducci, F. (2012). Leveraging social media sources to generate personalized music playlists, *E-Commerce and Web Technologies*, Springer, pp. 112–123.

Nam, J., Herrera, J., Han, B.-j. & Lee, K. (2012). An audio and music similarity and retrieval system based on sparse feature representations, *Extended Abstract to MIREX* .

Nam, J., Herrera, J., Slaney, M. & Smith, J. (2012). Learning sparse feature representations for music annotation and retrieval, *Proceedings of the International Symposium on Music Information Retrieval*, pp. 565–570.

Nanopoulos, A., Rafailidis, D., Symeonidis, P. & Manolopoulos, Y. (2010). Musicbox: Personalized music recommendation based on cubic analysis of social tags, *IEEE Transactions on Audio, Speech, and Language Processing* **18**(2): 407–412.

Neumayer, R. & Rauber, A. (2007). Integration of text and audio features for genre classification in music information retrieval, *Advances in Information Retrieval*, Springer, pp. 724–727.

Nyquist, H. (1928). Certain topics in telegraph transmission theory, *Transactions of the American Institute of Electrical Engineers* **47**(2): 617–644.

Pachet, F. & Aucouturier, J. (2004). Improving timbre similarity: How high is the sky?, *Journal of negative results in speech and audio sciences* **1**(1): 1–13.

Pampalk, E., Pohle, T. & Widmer, G. (2005). Dynamic playlist generation based on skipping behavior, *Proceedings of the International Conference on Music Information Retrieval*, Vol. 2, pp. 634–637.

Panagakis, Y. & Kotropoulos, C. (2012). Automatic music tagging by low-rank representation, *IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, pp. 497–500.

Pauws, S. & van de Wijdeven, S. (2005). User evaluation of a new interactive playlist generation concept, *Proceedings of the International Conference on Music Information Retrieval*, Vol. 5, pp. 638–643.

Pohle, T., Pampalk, E. & Widmer, G. (2005). Evaluation of frequently used audio features for classification of music into perceptual categories, *Proceedings of the International Workshop on Content-Based Multimedia Indexing*.

Rendle, S. & Schmidt-Thieme, L. (2010). Pairwise interaction tensor factorization for personalized tag recommendation, *Proceedings of the third ACM international conference on Web search and data mining*, ACM, pp. 81–90.

Seyerlehner, K., Schedl, M., Knees, P. & Sonnleitner, R. (2012). A refined block-level feature set for classification, similarity and tag prediction, *Extended Abstract to MIREX* .

Shardanand, U. & Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth", *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press/Addison-Wesley Publishing Co., pp. 210–217.

Sigurdsson, S., Petersen, K. B. & Lehn-Schiler, T. (2006). Mel frequency cepstral coefficients: An evaluation of robustness of MP3 encoded music, *Proceedings of the Seventh International Conference on Music Information Retrieval*, pp. 286–289.

Skowronek, J., McKinney, M. & Van De Par, S. (2006). Ground-truth for automatic music mood classification, *Proceedings of the International Conference on Music Information Retrieval*, pp. 395–396.

Skowronski, M. & Harris, J. (2002). Increased MFCC filter bandwidth for noise-robust phoneme recognition, *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, IEEE, pp. I–801.

Sordo, M., Laurier, C. & Celma, O. (2007). Annotating music collections: How content-based similarity helps to propagate labels, *Proceedings of 8th International Conference on Music Information Retrieval*, pp. 531–534.

Sordo, M. et al. (2012). *Semantic annotation of music collections: A computational approach*, PhD thesis, Universitat Pompeu Fabra.

Stevens, S. S., Volkman, J. & Newman, E. (1937). A scale for the measurement of the psychological magnitude of pitch, *Journal of the Acoustical Society of America* **8**(3): 185–190.

Sturm, B. L. (2012). A survey of evaluation in music genre recognition, *Proceedings of the International Workshop on Adaptive Multimedia Retrieval* pp. 1–41.

Su, J.-H., Yeh, H.-H., Yu, P. S. & Tseng, V. S. (2010). Music recommendation using content and context information mining, *Intelligent Systems, IEEE* **25**(1): 16–26.

Tardieu, D., Charbuillet, C., Cornu, F. & Peeters, G. (2011). Mirex-2011 single-label and multi-label classification tasks: Ircamclassification2011 submission, *MIREX Extended Abstract* .

Turnbull, D., Barrington, L. & Lanckriet, G. (2008). Five approaches to collecting tags for music, *International Society for Music Information Retrieval 2008*, pp. 225–230.

Turnbull, D., Barrington, L., Torres, D. & Lanckriet, G. (2008). Semantic annotation and retrieval of music and sound effects, *IEEE Transactions on Audio, Speech, and Language Processing* **16**(2): 467–476.

Uitdenbogerd, A. & van Schyndel, R. (2002). A review of factors affecting music recommender success, *Proceedings of 3rd International Conference on Music Information Retrieval, Paris, France*, pp. 204–208.

Vargas, S. & Castells, P. (2011). Rank and relevance in novelty and diversity metrics for recommender systems, *Proceedings of the fifth ACM conference on Recommender systems*, ACM, pp. 109–116.

Yan, Q., Zhou, Z. & Li, S. (2012). Chinese accents identification with modified MFCC, *in* T. Zhang (ed.), *Instrumentation, Measurement, Circuits and Systems*, Vol. 127 of *Advances in Intelligent and Soft Computing*, Springer Berlin / Heidelberg, pp. 659–666.

Yoshii, K. & Goto, M. (2009). Continuous pLSI and smoothing techniques for hybrid music recommendation, *Proceedings of the 10th International Society for Music Information Retrieval Conference*, pp. 339–344.

Yoshii, K., Goto, M., Komatani, K., Ogata, T. & Okuno, H. (2008). An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model, *IEEE Transactions on Audio, Speech, and Language Processing* **16**(2): 435–447.

Zhang, Y. C., Séaghdha, D. Ó., Quercia, D. & Jambor, T. (2012). Auralist: introducing serendipity into music recommendation, *Proceedings of the fifth ACM international conference on Web search and data mining*, ACM, pp. 13–22.

Zhang, Z.-K., Zhou, T. & Zhang, Y.-C. (2011). Tag-aware recommender systems: a state-of-the-art survey, *Journal of Computer Science and Technology* **26**(5): 767–777.

Zheng, F., Zhang, G. & Song, Z. (2001). Comparison of different implementations of MFCC, *Journal of Computer Science and Technology* **16**(6): 582–589.

Appendix A

Published Papers

Horsburgh, B., Craw, S., Massie, S. & Boswell, R. (2011). Finding the hidden gems: Recommending untagged music, *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2256–2261.

Horsburgh, B., Craw, S. & Massie, S. (2012). Music-inspired texture representation, *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 52–58.

Horsburgh, B., Craw, S. & Massie, S. (2012a). Cold-start music recommendation using a hybrid representation, *Digital Economy All Hands Conference*.

Finding the Hidden Gems: Recommending Untagged Music

Ben Horsburgh and **Susan Craw** and **Stewart Massie** and **Robin Boswell**
IDEAS Research Institute
Robert Gordon University, Aberdeen, UK
{b.horsburgh, s.craw, s.massie, r.boswell}@rgu.ac.uk

Abstract

We have developed a novel hybrid representation for Music Information Retrieval. Our representation is built by incorporating audio content into the tag space in a tag-track matrix, and then learning hybrid concepts using latent semantic analysis. We apply this representation to the task of music recommendation, using similarity-based retrieval from a query music track. We also develop a new approach to evaluating music recommender systems, which is based upon the relationship of users liking tracks. We are interested in measuring the recommendation quality, and the rate at which cold-start tracks are recommended. Our hybrid representation is able to outperform a tag-only representation, in terms of both recommendation quality and the rate that cold-start tracks are included as recommendations.

1 Introduction

Over recent years a vast number of online music services have appeared and grown. Unlike high-street music stores, listeners now have instant access to all recorded music. Traditional methods of finding music do not scale to large online music collections, and so an area of Music Information Retrieval which has been given much attention recently is recommendation. Every major online music service now has a music recommender system available to users, helping them navigate music collections. Such systems have been keenly adopted by users, to the extent that artists can now become massively popular solely based on a viral online interest.

Core to current state-of-the-art music recommender systems is social meta-data. This is typically in the form of free-text tags which describe any musical entity, such as an artist or track. Recommendations are then made based upon the similarity of social tags which are applied to each track.

Tag-based recommender systems have proven to be very powerful, but there are scenarios where they do not perform well. One such scenario is the well-known cold-start problem, where tracks in a collection do not have any tags applied to them. In these situations the untagged tracks will never be recommended. Often this means that the track will also never be tagged, since no one has been recommended the track, creating a Catch-22 style scenario. When the iTunes store opened in 2003, $200,000$ tracks were available; in 2010 $13,000,000$ tracks[1] were available. This illustrates the steep increase in volume of tracks available online, all of which must be tagged to be included in a recommender system. In this paper we show how the recommendation discovery of cold-start tracks can be increased by incorporating audio content-based data into a recommenders' representation.

Measuring the quality of a recommender system is typically achieved using genre, mood or artist classification accuracy. While these measures may be part of a good recommendation, they do not directly measure quality. We present a new evaluation measure which is not based on classification accuracy, but instead measures the level of positive association between two tracks based on user listening data.

The paper is structured as follows. In Section 2 we discuss recent related work on recommendation and evaluation. Our method for including content-based data in a tracks representation is presented in Section 3. Our evaluation measure is described in Section 4. Section 5 describes the experiments which we run to evaluate our representation, and presents our results. In Section 6 we draw some conclusions.

2 Related Work

Knowledge which can be gathered online has proven to be invaluable to music recommendation [Plaza and Baccigalupo, 2009]. Central to many state-of-the-art recommender systems are social tags [Nanopoulos *et al.*, 2010]. While these systems perform very well when data is available, tracks which do not have data available are left out. Reducing this cold-start problem, and enabling users to discover cold-start tracks, has been the focus of much recent work.

In general, there are two approaches to increasing discovery in tag-based recommenders. The first is to use audio similarity to propagate tags throughout a collection, therefore allowing every track to have a tag representation [Bertin-Mahieux *et al.*, 2008]. The limitation of this approach is that content similarity does not directly correlate to tag similarity, and therefore many erroneous tags are propagated. The second approach is to directly incorporate a content-based representation into the tag representation, therefore always presenting data on which to compute a similarity. Bu *et al.*

[1] http://www.apple.com/itunes/features/

[2010] present a hypergraph model which combines content with tags. This is an intuitive way to combine many different types of representations, but no new concepts are learned from this combination. Levy and Sandler [2009] construct a representation matrix which combines tags and clustered content representations, which they name *muswords*. They then employ probabilistic latent semantic analysis to learn hybrid concepts which generalize both tags and content. All notion of content-based similarity however is lost in their method, due to the *muswords* being treated as a bag of words.

3 Music Representation

We develop a new representation which combines tags and content, designed to increase the rate of discovering untagged tracks in a recommender system. The collection we are using consists of 3495 tracks which span 951 artists and 16 genres, as defined by Gracenote[2]. For each track we collect tags and extract standard content-based representations. These representations are then used to create a new hybrid representation, designed to reduce the cold-start recommendation problem by increasing cold-start discovery.

3.1 Basic Representations

Our tag representation is built using data downloaded from Last.fm, using the API they provide[3]. We store the top 20 tags for each track, along with their frequency when applied to the track. In total 3537 distinct tags are collected, and on average a track has 18 tags available. Last.fm normalizes each tag frequency relative to the most frequent tag, stored as a percentage value; the most frequent tag always has a frequency of 100, and a tag occurring half as frequently is given a frequency of 50. These frequency values are represented as a 3537 wide vector. Each vector is extremely sparse, with an average of 0.51% non-zeros.

To extract content features from audio we process each track in time slices of $750ms$. Beyond providing computational efficiency, this time interval was chosen based on evidence which suggests humans can distinguish genre upon hearing anything more than $475ms$ [Gjerdingen and Perrott, 2008]. This amount of time is not sufficient for the listener to distinguish any rhythmic features, and therefore we only include spatial features. For each time slice we first compute a Hamming windowed Fast-Fourier-Transform (FFT), and then extract our representations. The average over all time slices is then used as the final representation.

The inspiration behind our content features is to describe key spatial aspects of music: pitch, texture and harmony.

- A chroma representation describes the intensities of musical notes, and captures pitch [Kim and Narayanan, 2008]. To represent chroma we extract the intensity of each musical note found within the frequency domain of a time slice. These range from C0 at $8.176Hz$ to B7 at $3951Hz$. This provides a measure of the intensity of each musical note within an 8 octave range, described as a 96 bin vector.
- Mel Frequency Cepstral Coefficients (MFCC), originally used in speech recognition, describe the timbre of sound. MFCCs have proven to be very useful for genre, artist and mood classification, and so we are interested in how they perform for recommendation. We implement MFCCs as described by Sigurdsson *et al.* [2006], using 20 equally spaced triangular Mel filters, and retaining the first 12 MFCC's in our representation.
- To represent harmony we include a discretized representation of each FFT, which we name the Discretized-Frequency-Domain (DFD). When two musical notes are played simultaneously many harmonics of each note interact to create a new harmonic sound, heard as a result of the complex distribution of frequencies. DFD aims to capture this distribution by discretizing the frequency domain of each time slice into 200 equally spaced buckets. The sum of all intensities within each bucket is used.

[2]http://www.gracenote.com/

[3]http://www.last.fm/api

3.2 Tag-Concept Representation

Each of the basic representations capture specific facets of each track. However, they do not contain any knowledge of how those facets relate to the collection of tracks. This is well known throughout text retrieval, where synonymy is a problem. Two terms may be synonymous, but a direct comparison will always treat them as being different.

Latent Semantic Analysis (LSA) is a technique which generalizes the terms used into a conceptual representation, thus capturing synonyms. This is achieved by first constructing a term-document matrix, and then applying singular value decomposition (SVD). SVD decomposes a matrix, M, such that $M = U\Sigma V^T$; matrix U consists of tags and track-concepts, and matrix V^T consists of tag-concepts and tracks.

The tag-concept representation we use is V^T, obtained by decomposing the following matrix:

$$M = \begin{array}{c} \\ t_1 \\ t_2 \\ \vdots \\ t_n \end{array} \begin{array}{c} \begin{array}{cccc} T_1 & T_2 & & T_N \end{array} \\ \begin{pmatrix} f_{11} & f_{12} & \dots & f_{1N} \\ f_{21} & f_{12} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ f_{n1} & f_{n2} & \dots & f_{nN} \end{pmatrix} \end{array}$$

where T_i denotes a track, and t_j denotes a tag. f_{ij} denotes the frequency of tag t_i when applied to track T_j as defined by Last.fm.

3.3 Hybrid-Concept Representation

The tag-concept representation described suffers from the cold-start problem; no generalized concepts can be defined for untagged tracks, and therefore these tracks will not be recommended from tagged tracks. To reduce this problem we create a hybrid-concept representation, where content is included.

The aim for our hybrid representation is to include content in the matrix M in such a way that SVD is able to generalize tag and content features into meaningful concepts. This generalization is extremely important when the cold-start problem exists. Suppose the query track is well tagged, and a frequent tag is happy, generalization will make it possible to

recommend a non-tagged track which shares the same concept of happy, due to its content representation.

To include content in M we extend the number of rows by the number of bins in the content representation being used. This extended matrix is constructed as follows:

$$M = \begin{array}{c} \\ t_1 \\ t_2 \\ \\ t_n \\ c_1 \\ c_2 \\ \\ c_m \end{array} \begin{array}{c} \begin{array}{cccc} T_1 & T_2 & & T_N \end{array} \\ \begin{pmatrix} f_{11} & f_{12} & \dots & f_{1N} \\ f_{21} & f_{22} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ f_{n1} & f_{n2} & \dots & f_{nN} \\ i_{11} & i_{12} & \dots & i_{1N} \\ i_{21} & i_{22} & \dots & i_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ i_{m1} & i_{m2} & \dots & i_{mN} \end{pmatrix} \end{array}$$

where c_i denotes a bin in our content representation, and i_{ij} denotes the intensity of bin c_i for track T_j. The values used in the matrix are the normalized intensity values of a content-based representation. These are normalized in the same way as with Last.fm tags

SVD generalizes well when sparse data is used. For this reason we only include the 20 most intense bins within each tracks' content representation. Our initial experiments did not introduce this sparsity, and obtained a significantly lower evaluation score. While intensity is different from frequency in tags, high valued bins are still the most reflective of a track. For example, a tag-based concept capturing "lots of Bass" may be present in a DFD based form, where specific bins for "bass" may be high, and bins for other frequency ranges may be low.

4 Recommendation Quality

Many evaluations throughout music retrieval use the classification accuracy of artists, genre or mood [Flexer *et al.*, 2010]. It would be much more desirable to be able to conduct a full user evaluation, as with [Firan *et al.*, 2007], but such evaluations however are impractical for iterative evaluations. Ellis *et al.* [2002] propose several methods of replicating a real-world user evaluation. Their first approach is based on using complex network-analysis to define a similarity measure. One drawback with this shortest-path based approach is that popular tracks are linked by one edge to many tracks, making it difficult to know what is truly a good recommendation. The second method they propose uses a peer-to-peer cultural similarity measure, based on listening habits. This measure defines a good recommendation as one which is similarly popular to the query, and occurs in a large percentage of users profiles with the query. Again, the bias introduced by popularity makes this measure unsuitable for evaluation when discovery is important.

In this section we describe our new evaluation measure for recommendation quality, and then provide a discussion of how this measure behaves.

4.1 Measuring Quality

We propose a new evaluation strategy which attempts to replicate a real-world user evaluation, by using available data from internet users. The inspiration for our measure of recommendation quality comes from conversations between two people:

P1: I've been listening to this awesome band called Klaxons.

P2: Me too. Do you like Neon Plastix?

P1: Never heard of them!

P2: You should get their CD, you'd like them.

The key point from this type of conversation is that Person P2 has made an association between Klaxons and Neon Plastix. This association is based upon P2 having listened to both artists, and having liked both artists. The concepts of listening to and liking both artists forms the basis for our measure.

We define a high quality recommendation as one for which Person 2 makes an association, and a low quality recommendation as one which Person 2 does not. We then extrapolate this definition to take account of n peoples' opinions. The proportion of people who agree that there is an association between the two tracks quantifies the strength of association, shown in Eq. 1.

$$\text{association}(t_i, t_j) = \frac{\text{likes}(t_i, t_j)}{\text{listeners}(t_i, t_j)} \quad (1)$$

where t_i and t_j are tracks, $\text{listeners}(t_i, t_j)$ is the number of people who have listened to both t_i and t_j, and $\text{likes}(t_i, t_j)$ is the number of listeners who have liked both t_i and t_j. This definition is similar to many offline user evaluations, where a set of people are asked to rate pairs of tracks. The average of all ratings, or associations, quantifies the strength of the recommendation. We are therefore simulating an offline user experiment where user ratings are binary.

To apply this type of evaluation one must first have users available. Instead of offline users we use online users, the data for which is available using the Last.fm API. We collect data for over 175000 users, over a period of 2 months. For each user the tracks which they have clicked the "thumbs up" button on Last.fm are recorded, providing data on the tracks they like. On average a single user will have liked 5.4 tracks in our collection. Further information collected from Last.fm is the number of distinct listeners of each track.

To determine the level of association in Eq. (1) between two tracks we must first know two important facts; the number of users in our collection who have listened to a given pair of tracks, and the number of users who have liked both tracks. From our collected Last.fm data we know the number of users who have liked both tracks in any given pair. Unfortunately, data on who has listened to both tracks is unavailable from their API. However, the number of users who have listened to each track is available, allowing an estimate of listeners to both.

To estimate the number of users in our collection who have listened to a pair of tracks, we first assume that all tracks are listened to independently. We then estimate the number of users who have listen to each possible pairing of tracks, scaled to our collection size, using Eq. 2.

$$\text{listeners}(t_i, t_j) = \frac{C(t_i)}{|\text{Last.fm}|} \cdot \frac{C(t_j)}{|\text{Last.fm}|} |\text{Collection}| \quad (2)$$

where $C(t_i)$ denotes the count of users who have listened to track t_i, $|\text{Last.fm}|$ denotes the number of users of Last.fm, and $|\text{Collection}|$ denotes the number of users in our collection. Using the estimate in Eq. (2), we are able to calculate the level of association between tracks as described in Eq. (1).

The measure of association defines quality on a scale of 0 to 1. A value of 0 occurs when there is no evidence of users liking a pair of tracks. A value of 1 occurs when the number of users who like a pair of tracks is equal to the number who have listened to the pair. It is possible to obtain a score of greater than 1, since we are using an estimate to determine the number of listeners. As the actual level of association between a pair of tracks increases, so does the amount we under-estimate the number of listeners, due to our assumption of independence. To handle this we set a limit on the measure to 1. When an under-estimate occurs, such that a value of more than 1 is obtained, it is clear that the two tracks display a very high level of association, and therefore a score of 1 reflects this.

4.2 Discussion of Measure

The evaluation measure we have described is based on associations between users having listened to and liked tracks. This approach may be compared to collaborative filtering, but a key difference exists. Collaborative filtering is an approach for finding similar users, and then recommending tracks which may be of interest. Our evaluation measure does not find similar users, but quantifies the global level of agreement between sets of users. Collaborative Filtering has been developed to make recommendations: our method has been developed to evaluate recommendations.

The mean level of association in our collection is 0.064. An association score of 0 is achieved by 87% of all possible recommendation pairs, and a score between 0 and 1 is achieved by 10% of all pairs. In our collection 3% of possible query-recommendation pairs obtain an association score of greater than 1, which we modify to be 1. The distribution of association scores for all possible pairs in our collection is shown in Figure 1. We have excluded values for associations of 0 and 1 so that the meaningful part of the curve can be easily observed.

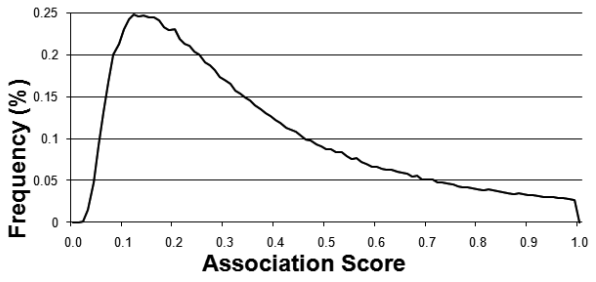


Figure 1: Distribution of association values

The distribution observed centers around the score 0.12, which occurs for 0.25% of all possible query-recommendation pairs. The tail of the curve decays less steeply than expected, due to our estimate of listeners. If an estimate was not used we would expect the tail to decay much faster, and reach a frequency of approximately 0 much earlier, reflecting the level of true agreement. 13% of all possible pairs in our collection obtain a score of greater than 0, with 10% lying in the distribution shown. This shows our measure is suitably discriminant of high and low quality recommendations.

5 Experiments

We evaluate several recommender systems, each using a different representation for tracks. The representations evaluated are as follows:

- Tag is the LSA representation from the tag-track matrix (as described in Section 3.3)
- DFD, MFCC and Chroma use LSA representations from the corresponding matrix where the tags are replaced with the appropriate content-based representation
- Hybrid is the LSA representation from the extended matrix for tags with content features
- Random does not have a representation, but instead randomly selects tracks to recommend

Euclidean distance is used to retrieve and rank retrieved tracks for all representations. This is a metric which is commonly used in content-based retrieval. It is standard that a cosine similarity measure is used for text, because it defines similarity based on terms co-occurring with the same relative frequency. For tags however, the value of frequency values used is important. Euclidean distance is able to capture differences in these values, and is therefore best suited to each of our representations.

We follow a standard experimental design for each representation. We hold-out 30% of tracks as test queries, and the remaining 70% make the set of possible recommendations. For each track in our test set we obtain the top 20 recommendations. We evaluate each recommendation subset in this top 20, that is, we calculate the association score for only 1 recommendation, 2 recommendations, and so on. For each recommender system we repeat the experiment procedure four times. The results presented are an average of the results obtained over all runs. All error bars shown are at a 95% confidence interval.

5.1 Recommender Results

Initially we evaluate each content-based representation to understand which performs best at the task of music recommendation. The expectation was that MFCCs would perform best, due to their strength in other MIR tasks. Figure 2 shows the results for recommender systems built using the DFD, MFCC and Chroma representations. The x-axis is the number of recommendations that were made, and the y-axis is the association score for this number of recommendations. For a recommendation set of up to 8 recommendations, the DFD representation outperforms MFCC's. The first recommendations presented are far more important than those further down the list, and therefore we use the DFD representation as the content component in our hybrid representation.

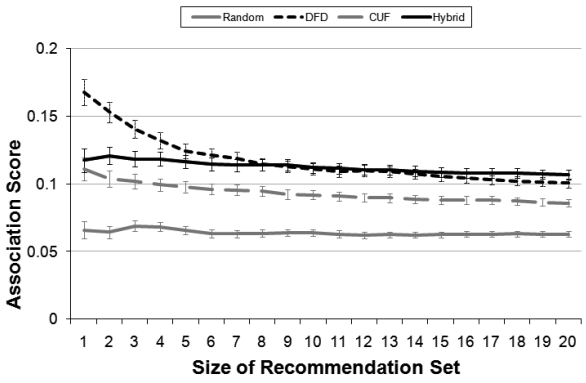


Figure 2: Content-based representations

The second experiment (Figure 3) is intended to understand the performance of the Tag, DFD and Hybrid representations. We construct our Hybrid representation using DFD as the content component, since it outperforms MFCCs and Chroma. It is interesting to note the difference between the Tag and Hybrid association score, and the DFD association score. When DFD only is used, the quality of recommendations made is extremely low. When DFD is integrated into our Hybrid representation however, the quality of recommendations is competitive with the Tag recommender. The reason for this is that the process of SVD is taking advantage of the discriminant power of tags far more than it does for DFD.

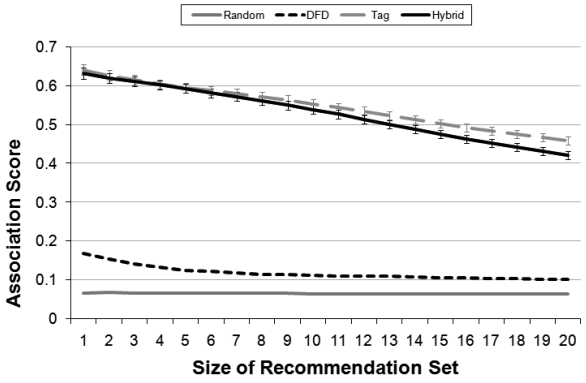


Figure 3: Recommendation quality

Our final experiment shows how each of the representations performs on our collection with injected cold-start. Our collection has only 2% natural cold-start within it. To simulate the cold-start problem we inject the problem into a randomly selected 25% of our collection by removing their tags, allowing us to clearly observe the problems effects.

When the cold-start problem has been injected into our collection, our Hybrid representation outperforms the Tag representation at all recommendation list sizes (Figure 4). The SVD of the original track-representation matrix generalizes the representation, and takes advantage of discriminant concepts. In the case of our Tag representation, 25% of our data is identical; it has no tags. For these data there is no discriminant concepts which can be used, and therefore no meaningful similarity measure can be defined.

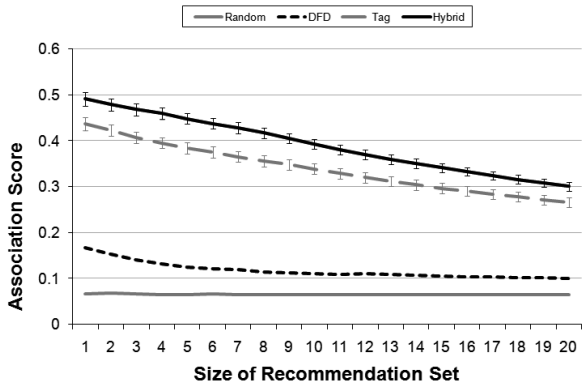


Figure 4: Recommendation quality in cold-start scenario

It should be noted that the injection of cold-start tracks has nevertheless lowered the performance of our Hybrid representation. This is directly caused by the untagged tracks, since tags naturally outperform content representations. In this scenario however, our Hybrid representation still has a representation for the untagged tracks. Further, the representation available for these tracks is always available for the whole collection. This in turn means that SVD is again able to take advantage of discriminant concepts within the entire representation. It is for this reason that our Hybrid representation is able to outperform the Tag representation, and provide a recommender system which can produce higher quality recommendations when the cold-start problem exists.

5.2 Discovery Results

A complementary measure we use is the discovery rate of cold-start items. This is the ratio of untagged items which are recommended, when the query track is tagged.

Figure 5 shows the discovery rate for each representation. The random and DFD based recommenders achieve a consistent discovery rate of approximately 25%. While this at first appears to be good, these results must be interpretted with Figure 4 in mind. The random recommender obtains a high discovery rate, but also obtains a very low association score. DFD obtains the same discovery rate since it is unbiased towards tag representation. The association score is again low however, meaning that the tracks discovered are not strong recommendations.

The Tag representation has a discovery rate of approximately 0 for the first 4 recommendations. This is expected, since the Tag representation will naturally rank tagged tracks as most similar. As the size of the recommendation set increases the tag representation gradually makes more discoveries. The reason for this is that for some query tracks there are no more recommendation tracks which share tag-concepts, and therefore everything obtains a similarity of 0.

The Hybrid representation has a steeper discovery rate than Tags, starting at a rate of 0.03 for 1 recommendation. Un-

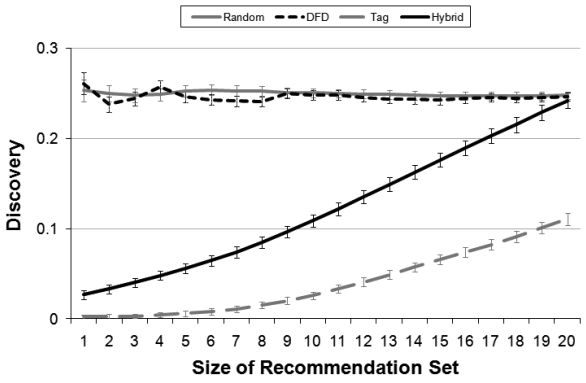


Figure 5: Recommendation of untagged Tracks

like with the random and DFD recommenders, DFD achieves this increase in discovery while maintaining a high association score. This shows that the tracks being discovered are relevant, and good recommendations.

The increased discovery rate of the Hybrid representation is due to the influence of the DFD component. Where no tags exist, a track still has a concept representation, and is therefore able to obtain meaningful recommendations, avoiding the 0 similarity problem found in the tag recommender.

6 Conclusions

We have developed a novel way of incorporating content-based representations into a tag-based recommender system. This has been achieved by extending the tag-track matrix to include content, and then learning hybrid concepts using LSA. Through generalization, the concepts learned make associations between tags and content in such a way that recommendations can still be made when tags are absent.

A new evaluation measure simulating real-world user evaluations of recommender systems has also been developed. This evaluation defines associations between pairs of tracks based upon users having listened to both tracks, and having liked both tracks. In developing this measure, we have shown how recommender systems can be evaluated in terms which are directly relevant to recommendation, not classification.

Our results show that by including content an increased rate of discovering cold-start items can be achieved. Further, this discovery of cold-start improves the overall performance of the recommender system, and the Hybrid representation is able to outperform the Tag representation when the cold-start problem exists. When cold-start tracks do not exist, our recommender is competitive with a tag-only recommender.

The merging of two representations in the way we have developed has promising directions. We have shown how content can be of benefit to tag-based recommenders. Using our approach it may even be possible to include many different sources of knowledge, such as play lists and mined web data, creating a truly all-encompassing representation. Humans make recommendations based on a wealth of information, the next challenge is to expand representations to include this knowledge.

References

[Bertin-Mahieux *et al.*, 2008] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: A model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2):115–135, 2008.

[Bu *et al.*, 2010] J. Bu, S. Tan, C. Chen, C. Wang, H. Wu, L. Zhang, and X He. Music recommendation by unified hypergraph: combining social media information and music content. In *Proc. International Conference on Multimedia*, pages 391–400. ACM, 2010.

[Ellis *et al.*, 2002] D. Ellis, B. Whitman, A. Berenzweig, and S. Lawrence. The quest for ground truth in musical artist similarity. In *Proc. International Society for Music Information Retrieval*, pages 170–177. Ircam, 2002.

[Firan *et al.*, 2007] C.S. Firan, W. Nejdl, and R. Paiu. The benefit of using tag-based profiles. In *LA-WEB 2007. Latin American*, pages 32 –41, 2007.

[Flexer *et al.*, 2010] A. Flexer, M. Gasser, and D. Schnitzer. Limitations of interactive music recommendation based on audio content. In *Proc. 5th Audio Mostly Conference: A Conference on Interaction with Sound*, pages 1–7. ACM, 2010.

[Gjerdingen and Perrott, 2008] R.O. Gjerdingen and D. Perrott. Scanning the dial: The rapid recognition of music genres. *Journal of New Music Research*, 37(2):93–100, 2008.

[Kim and Narayanan, 2008] S. Kim and S. Narayanan. Dynamic chroma feature vectors with applications to cover song identification. *In Proc. IEEE 10th Workshop on Multimedia Signal Processing*, 984–987, 2008.

[Lee *et al.*, 2010] S.K. Lee, Y.H. Cho, and S.H. Kim. Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations. *Information Sciences*, 180(11):2142–2155, 2010.

[Levy and Sandler, 2009] M. Levy and M. Sandler. Music information retrieval using social tags and audio. *IEEE Transactions on Multimedia*, 11(3):383–395, 2009.

[Nanopoulos *et al.*, 2010] A. Nanopoulos, D. Rafailidis, P. Symeonidis, and Y. Manolopoulos. Musicbox: Personalized music recommendation based on cubic analysis of social tags. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2):407 –412, 2010.

[Plaza and Baccigalupo, 2009] E. Plaza and C. Baccigalupo. Principle and praxis in the experience web: A case study in social music. In *The Proc. 8th International Conference on Case-Based Reasoning*, pages 55–63. Springer, 2009.

[Sigurdsson *et al.*, 2006] S. Sigurdsson, K.B Petersen, and T. Lehn-Schiler. Mel frequency cepstral coefficients: An evaluation of robustness of MP3 encoded music. In *Proc. International Society for Music Information Retrieval*. University of Victoria, 2006.

Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Music-Inspired Texture Representation

Ben Horsburgh and **Susan Craw** and **Stewart Massie**
IDEAS Research Institute
Robert Gordon University, Aberdeen, UK

Abstract

Techniques for music recommendation are increasingly relying on hybrid representations to retrieve new and exciting music. A key component of these representations is musical content, with texture being the most widely used feature. Current techniques for representing texture however are inspired by speech, not music, therefore music representations are not capturing the correct nature of musical texture. In this paper we investigate two parts of the well-established mel-frequency cepstral coefficients (MFCC) representation: the resolution of mel-frequencies related to the resolution of musical notes; and how best to describe the shape of texture. Through contextualizing these parts, and their relationship to music, a novel music-inspired texture representation is developed. We evaluate this new texture representation by applying it to the task of music recommendation. We use the representation to build three recommendation models, based on current state-of-the-art methods. Our results show that by understanding two key parts of texture representation, it is possible to achieve a significant recommendation improvement. This contribution of a music-inspired texture representation will not only improve content-based representation, but will allow hybrid systems to take advantage of a stronger content component.

Introduction

Over the last decade the way in which people find and enjoy music has completely changed. Traditionally a listener would first hear about new music either through their friends or mass-media, such as radio and magazines. They would then visit a record store and buy a hard copy of the music. It is now equally as likely that a listener will have been recommended new music by an algorithm, and that they will either stream or buy a soft copy of the music. There are possibly millions of tracks on a website, with only a limited number being of interest to the user. This presents an interesting new challenge: how to decide which tracks should be recommended to a user.

Many current state-of-the-art techniques for providing music recommendations are based on the idea of similarity. Given one or more songs that the listener likes, recommender systems must provide further songs that the listener will like. One of the most popular ways to do this is to compare meta-data about the songs. Such meta-data often includes textual tags, and audio descriptors, which are combined into a hybrid representation for each song. When examining these hybrid representations it becomes clear that tags offer much greater accuracy than audio descriptors. For this reason a lot of current research is focussing on how such audio descriptors can be improved, and how these can be better integrated with tags.

There are two approaches to describing musical audio: in terms of its structure, such as rhythm, harmony and melody; or in terms of its feel, such as texture. The choice of which approach to use can differ depending on the task. However, in almost all tasks, texture has proven to be very popular and successful. The most widely used texture descriptor is the mel-frequency-cepstral-coefficient (MFCC) representation. Although originally developed for speech recognition, MFCCs have proven to be robust across many musical tasks, including recommendation. This foundation in speech processing however can make texture difficult to understand in a musical sense.

We take a novel approach to examining the MFCC representation in a musical context, and make two important observations. The first observation is related to the resolution used at a key step in the algorithm, and how this corresponds to music. The second observation is that in the case of music, summarising the perceptual spectrum used by MFCC is undesirable. Based on these observations we develop a novel approach to music-inspired texture representation, and evaluate its performance using the task of music recommendation. This evaluation also shows that no single fixed-parameter texture representation works best for all recommendation models.

The paper is structured as follows. We first discuss related work on the MFCC representation, how it has been used for music, and some observations others have made on its suitability to music. We then develop our new texture representation by examining the MFCC representation, and discussing key steps. The following section describes our experimental design, and the three recommendation approaches with which we evaluate our approach. We then discuss the results obtained from our experiments, and finally we draw some conclusions from this work.

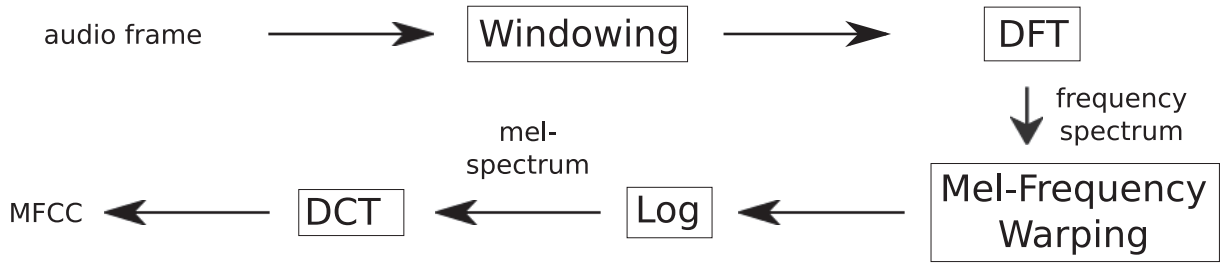


Figure 1: Block diagram of MFCC extraction algorithm

Related Work

One of the most important content-based representations for music recommendation is MFCC (Celma 2010). First developed for speech recognition (Mermelstein 1976), the MFCC representation is based on warping a frequency spectrum to human perception using the mel scale (Stevens, Volkmann, and Newman 1937).

One important difference between practical evaluations of MFCC for speech and music, is that speech can be evaluated in an objective manner (Zheng, Zhang, and Song 2001), while many music tasks are evaluated subjectively (Ellis et al. 2002). However, many authors have successfully evaluated the use of MFCC subjectively for genre classification (Lee et al. 2009) and music recommendation (Yoshii et al. 2008).

Logan (2000) investigates the suitability of the mel scale for musical texture, concluding that the scale is at least not harmful for speech / music discrimination, but perhaps further work is required to examine only music. Logan also concludes that at a theoretical level DCT is an appropriate transform to decorrelate the mel-frequency spectrum, but the parameters used require more thorough examination. We question whether the mel-frequency spectrum should be decorrelated at all, and evaluate the suitability of DCT for describing a music-inspired texture.

Several investigations have concluded that resolution is an important factor affecting MFCCs. It has been shown that computing MFCCs on MP3 audio is only robust if the audio is encoded at 128kbps or higher (Sigurdsson, Petersen, and Lehn-Schiler 2006). At lower encoding resolutions, the MFCCs extracted are not able to describe texture reliably. Pachet and Aucouturier (2004) investigate several parameters of MFCC and how they relate to music. Their first observation is closely linked to that of Sigurdsson, Petersen, and Lehn-Schiler; increasing the input resolution by increasing the sample rate improves the effectiveness of MFCC. In further work Aucouturier, Pachet, and Sandler (2005) examine the relationship between how many coefficients are retained after DCT, and the number of components used in a Gaussian Mixture Model. It is found that increasing the number of coefficients beyond a threshold is harmful for this model.

The MFCC representation involves smoothing the input data based on the mel scale, introducing a further step where resolution may be important. Most commonly 40 mel filters are used for smoothing, often described as 13 linear and 27 logarithmic filters, and 13 coefficients are retained after DCT. One popular toolbox for extracting MFCCs also uses these fixed parameters (Slaney 1998).

Yan, Zhou, and Li (2012) investigate the placement of mel filters for the application of Chinese speech recognition. When analysing MFCC across various regional accents, they discovered that the first two formants are most sensitive to accents. These formants are the lower-frequency section of the mel-spectrum, and so the authors re-distribute the mel filters used to provide a higher resolution in these critical bands.

Li and Chan (2011) also investigate the placement of mel filters while considering genre classification. The authors find that the MFCC representation is affected by the musical key that a track is played in. Rather than modifying the MFCC representation to resolve this issue, they instead normalise the key of each track before extracting MFCCs. In this paper we investigate the relationship between the mel scale and pitch, and develop our music-inspired texture representation from this.

Mel-Frequency Texture Representations

The MFCC representation attempts to describe the shape of a sound, with respect to how humans hear it. In this section we describe and examine this representation, and develop several modifications showing how this approach can be tailored to describe texture for music.

Frequency Spectrum

The MFCC representation is generated from a short time-sample of audio, as illustrated in Figure 1. The input is an audio frame of fixed sample rate and sample size. To reduce the effects of noise introduced by sampling, rather than having a continuous waveform, the initial step in Figure 1 is windowing. This windowed time-domain frame is then converted to a frequency-domain spectrum using the discrete-Fourier transform (DFT).

There are three properties of the frequency spectrum which are important:

- The size of the spectrum is half the size of the audio frame, and for computational efficiency is usually size 2^n.
- The maximum frequency is half the audio frame sampling rate.
- The spectrum bin resolution is the maximum frequency divided by the spectrum size.

For an audio frame with sample rate 44.1kHz and sample size 2^{10} (23ms), the size of the spectrum is 2^9, the maximum frequency is 22.05kHz, and the bin resolution is 43.1Hz.

The windowing and DFT steps are common to many content-based representations, and so this paper focusses on the later steps in Figure 1, which are more specific to texture.

Mel-Frequency Spectrum

The frequency spectrum output by the DFT describes the intensity of discrete frequency ranges. In order to correctly describe texture, one must warp this output so that it describes how humans hear frequencies and their intensities. This is achieved using two operations; frequency warping and intensity rescaling. Intensity rescaling, applied after the frequency warping, is achieved by taking the log of all frequency values. In this section we will focus on how the frequency warping is achieved.

Frequency is the physical scale describing sound oscillations, and must be warped into the perceptual mel scale (Stevens, Volkmann, and Newman 1937), thus mimicking humans interpret frequency. The following equation describes the conversion from frequency f to mel ϕ

$$\phi(f) = 2595 \log_{10}\left(\frac{f}{700} + 1\right) \tag{1}$$

and is illustrated by the curve in Figure 2. The horizontal axis is f, and the left vertical axis is $\phi(f)$.

Frequency warping is not achieved as a direct conversion from f to $\phi(f)$, but as a discretization of f based on $\phi(f)$. To avoid confusion, we will refer to frequency ranges defined by the DFT as bins, and to the ranges used for frequency warping as buckets.

M buckets are equally sized and spaced along $\phi(f)$, as illustrated by the triangles in Figure 2. The buckets on the f axis are equivalent to those on $\phi(f)$. The value of each bucket is a weighted sum of all its frequency bins, where the weights are defined by the triangular filter.

After discretization, the frequency spectrum has been warped into M mel-scaled values. After the log of each value has been taken, this set of values is known as the Mel-Frequency Spectrum (MFS).

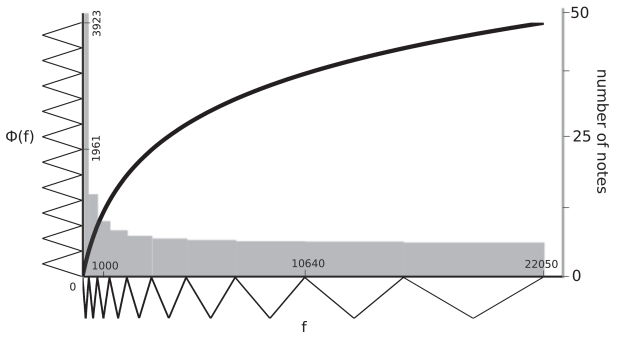


Figure 2: Frequency - mel curve

Mel-Frequency Filter Resolution

When MFCCs are discussed in the literature it is usually in terms of the frequency and mel scales. In this section we first make observations of the MFS in terms frequency and mel, and then in terms of a musical scale, in an attempt to examine its suitability for musical texture.

Continuing with our example using Figure 2, when 10 triangular mel-filters are used for frequency warping, each bucket is 392 mels wide. Converted to f, this means the smallest bucket is 291Hz, and the largest is 6688Hz. Examined as a percentage value, each bucket covers 10% of the mel scale, or between 0.013% and 30% of the frequency scale. At a first glance the smallest bucket perhaps seems too small, and the largest is perhaps too large, however, we must now examine the ranges from a musical point of view.

The most commonly used musical scale is the cents scale, which describes the distance in cents between two frequencies. The following equation shows how to calculate the number of cents between frequency f and reference frequency g.

$$c(f,g) = 1200 \log_2\left(\frac{f}{g}\right) \tag{2}$$

Using this scale, 100 cents is equal to one musical semitone, or adjacent piano key. For the scale to make sense g must correspond to the frequency of a musical note; for our examples we set $g = 16.35$, which is the musical note C. The lowest note on a standard piano is A at 27.5Hz.

If we consider our example of 10 filters again, but this time examining the size of the buckets in cents, we get a completely different picture. The smallest filter covers 4986 cents, which is just more than 49 musical notes, and the largest filter covers 626 cents, which is just more than 6 musical notes. This illustrates something completely different to examining discretization of f; musically the smallest bucket in f seems much too large in cents, and the largest bucket in f seems much too small in cents.

Figure 2 illustrates this conflict of resolution between f and cents. The light grey shading shows the number of musical notes for each triangular filter along f. The right vertical axis shows the scale for these bars. It is clear that the smallest triangular filter on f has the largest number of notes. As the filters on f become larger, the number of notes quickly becomes lower and flattens at around 6 notes.

There are 12477 cents between our reference frequency g and 22050Hz, which is just over 124 musical notes. Re-examining the discretization as a percentage, the smallest bucket covers 39.5% of the musical scale, and the largest bucket covers 0.048%.

In practice the most common value of M for MFCCs is 40. At this value the smallest filter covers 23.5 musical notes, and the largest covers 1 musical note. Intuitively the smallest filter still seems much too large. Most musical notes are played at a relatively low frequency; the higher frequencies often describe harmonics of the lower note. Texture should describe all the elements, which when put together form a sound. If 23.5 notes are all grouped together, one can argue

these elements are too coarsely discretized, and thus the best texture description is not used.

We propose that M should be a more carefully considered parameter, and should be much larger than is typically used. There is one final consideration however; increasing M will decrease the size of the smallest filter. M is bounded by the point at which the size of the smallest filter becomes less than the resolution of the frequency spectrum, at which point duplicate and redundant information will be produced. The selection of M is therefore a balancing act between the resolution of the smallest and largest filters. As more filters are used, the resolution of higher filters will become much less than 1 musical note.

To denote the number of filters which are being used by any representation we will append the number in brackets. For example, when 60 filters are used the representation will be denoted as MFS(60).

Discrete Cosine Transform

The final step in Figure 1 is the Discrete Cosine Transform (DCT), which takes the mel-frequency spectrum as an input, and outputs a description of its shape, the mel-frequency cepstrum. The coefficients of this cepstrum are the MFCC representation.

The description provided by the DCT is a set of weights, each correlating to a specific cosine curve. When these curves are combined as a weighted sum, the mel-frequency spectrum is approximated. To calculate the mel-frequency cepstrum, X, the DCT is applied to the mel-frequency spectrum x as follows

$$X_n = \sum_{m=0}^{M-1} x_m \cos\left[\frac{\pi}{M}\left(m+\frac{1}{2}\right)n\right] \quad (3)$$

for $n = 0$ to $N-1$. Typically M is 40, and N is 13.

Discrete Cosine Transform Observations

The idea behind using DCT for texture representations comes from the speech processing domain. Speech has two key characteristics: formants are the meaningful frequency components which characterise a sound; and breath is the general noise throughout all frequency components, and thus much less meaningful.

For speech, DCT offers strong energy compaction. In most applications breath is undesirable, and so only the first few DCT coefficients need to be retained. This is because these coefficients capture low-frequency changes within the mel-frequency spectrum. The high frequency detail primarily characterised by noisy breath is not needed. Most commonly 13 coefficients are retained when 40 mel filters are used. Retaining 40 coefficients would give a very close approximation of the mel-frequency spectrum. Some authors do not retain the 0th, DC coefficient in their representation.

For music, the concepts of formants and breath do not apply. It is true that a vocalist is common, meaning formants and breath are present, however, music is also present. If only a few coefficients are retained from the DCT, then much information about percussion and the general feel of the music is lost. In music the mel-frequency spectrum is much more rich, and this detail is important to describing texture.

One could argue then that for music more coefficients should be retained from the DCT. The primary reason for using the DCT however still stems from the idea of separating formants and breath, or information from noise. We propose that the DCT should not be used for a mel-frequency based texture representation for music. All of the frequency information is potentially meaningful, and therefore should not be summarised. The special case against this argument is live music, where noise may still be present. However, for general music retrieval, most music is studio recorded, where a lot of effort is taken to remove all noise.

When no DCT is used for describing texture we denote the representation mel-frequency spectrum (MFS). When the DCT is used we denote the representation MFCC. When 40 mel filters are used we denote the representations as MFS(40) or MFCC(40).

Experiment Design

We evaluate how MFS performs at the task of music recommendation. We construct this as a query-by-example task, where 10 recommendations are provided for any given query. 10-fold cross validation is used, and our results are compared to those achieved by MFCC.

Dataset

The dataset we use consists of 3174 tracks by 764 artists, and are from 12 distinct super-genres. The most common genres are Alternative (29%) and Pop (25%). Each track in our collection is sampled at 44.1kHz, and processed using a non-overlapping samples of size of 2^{13} (186ms). Each frequency spectrum computed has a maximum frequency of 22.05kHz, and a bin resolution of 5.4Hz. For each sample we use a Hamming window before computing the DFT, and then extract each of the texture representations.

Each model is constructed using texture vectors, extracted from each sample in a given track. We extract texture vectors for both MFS and MFCC using 40, 60, 80, 100, and 120 filters. For 40 filters the smallest bucket contains just over 25 notes, and the largest contains just over 1 note. For 120 filters the smallest bucket contains 4 notes, and the largest contains 0.5 notes.

Recommendation Models

We construct three well-known models to avoid drawing conclusions specific to one type of recommender model.

Latent-Semantic-Indexing of Mean-Vector (LSA) - A mean texture vector is first computed for each track, where each dimension corresponds to the mean value across all of the track's texture vectors. We then construct a track-feature matrix using these mean texture vectors. The JAMA package (Hicklin et al. 2000) is used to generalise this matrix by LSI, and each query is projected into this generalised search space. Recommendations are made based on Euclidean distance as in previous work (Horsburgh et al. 2011).

Vocabulary-Based Model (VOC) - Vocabulary-Based methods are often found in hybrid recommender systems, and so we examine this popular model. To generate a vocabulary we use the k-means algorithm to cluster 20000 texture vectors selected at random from all tracks. For each track we count the number of samples which are assigned to each cluster, and construct a cluster-track matrix. A TF-IDF weighting is applied, and cosine similarity used to make recommendations.

Gaussian Mixture Model Approach (GMM) - A GMM models the distribution of a tracks' texture vectors as a weighted sum of K more simple Gaussian distributions, known as components. Each weighted component in the GMM is described by its mean texture vector and covariance matrix (Aucouturier, Pachet, and Sandler 2005). We learn a GMM for each track using the Weka implementation of the EM algorithm (Hall et al. 2009).

If each track were represented by a single Gaussian distribution, recommendations can be made using Kullback-Leibler divergence. With GMMs however each track is represented by K weighted Gaussian distributions, and so we make recommendations based on an approximation of Kullback-Leibler divergence (Hershey and Olsen 2007). For each component in a query track's GMM, we compute the minimum Kullback-Leibler divergence to each component in a candidate track's GMM. The estimated Kullback-Leibler divergence between the query track and the candidate recommendation is calculated as the weighted average of the minimum divergence between all components. Recommendations are ranked using this estimated Kullback-Leibler divergence.

Evaluation Method

Our evaluation method uses data collected from over 175,000 user profiles, extracted from Last.fm using the AudioScrobbler API[1] over 2 months. For each user we record tracks which they have said they like. On average, each user in our collection likes 5.4 songs. To measure recommendation quality we use the association score that we developed in previous work (Horsburgh et al. 2011).

$$\text{association}(t_i,t_j) = \frac{\text{likes}(t_i,t_j)}{\text{listeners}(t_i,t_j)} \quad (4)$$

where t_i and t_j are tracks, $\text{listeners}(t_i,t_j)$ is the number of people who have listened to both t_i and t_j, and $\text{likes}(t_i,t_j)$ is the number of listeners who have liked both t_i and t_j. The number of listeners is estimated using statistics from Last.fm, and assumes listens are independent. Using this evaluation measure allows us to understand the likelihood that someone who likes track t_i will also like t_j.

Results

We compare our MFS music-inspired texture with MFCCs for the task of music recommendation using 40 filters. We then investigate the effects of increasing the number of filters used, and compare the best MFS and MFCC representations for each model.

MFS vs MFCC

Figure 3 shows the results using MFS and MFCC for each model. The vertical axis shows the association score, and the horizontal axis shows the number of recommendations evaluated. Each value shown is the average association score at the number of recommendations made. All error bars show significance at a 95% confidence interval.

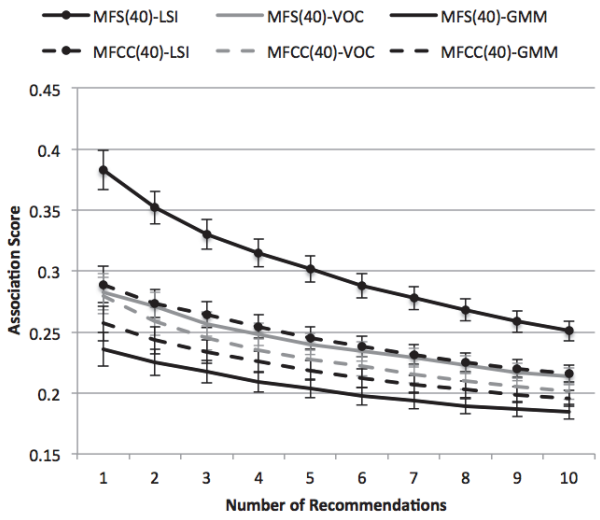


Figure 3: Comparison of MFS and MFCC using 40 filters

MFS-LSI achieves a significantly better association score than all other models and representations. MFS-VOC provides a significant quality increase over MFCC-VOC when 2 or more recommendations are made. MFS out performs MFCC with LSI and VOC because both models group data at the collection level; LSI finds associations between dimensions in the texture vectors, and VOC groups texture vectors into clusters. With MFCC, each texture vector is first summarised by DCT without respect to the collection, and therefore LSI and VOC are not able to model the data effectively.

Unlike LSI and VOC, for 40 filters MFCC-GMM is significantly better than MFS-GMM. The reason for this is that GMM behaves differently; each model is constructed for a single track. We learned each GMM using a diagonal covariance matrix, which has the effect of making dimensions in the texture vector independent. This means associations between dimensions are not learned.

Effect of Number of Filters

We want to explore the effect of increasing the number of filters on recommendation, and so examine a more simple recommendation task. Figure 4 shows the average association score of the top 3 recommendations for each model. The horizontal axis is grouped by the model used, and each bar corresponds to a given number of filters.

[1] http://www.last.fm/api

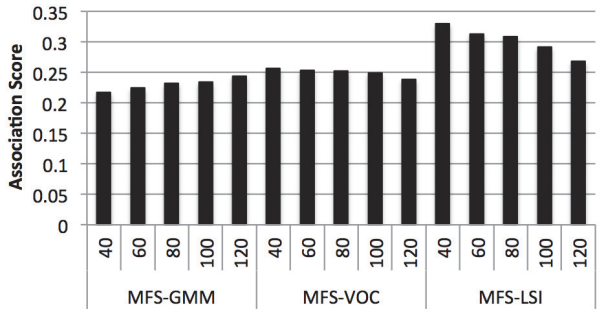


Figure 4: Effect of filters on MFS

Increasing the number of filters significantly increases the recommendation quality of MFS-GMM, does not significantly affect MFS-VOC, and significantly decreases the quality of MFS-LSI. GMM is improved because more independent information is available to the model, and so more meaningful distributions can be learned. VOC does not change because a similar vocabulary is formed regardless of the number of filters. The performance of LSI decreases, showing that the model can generalise a low number of filters more effectively. Figure 5 shows the effect changing the number of filters for MFCC. No correlation appears between the number of filters used and association score.

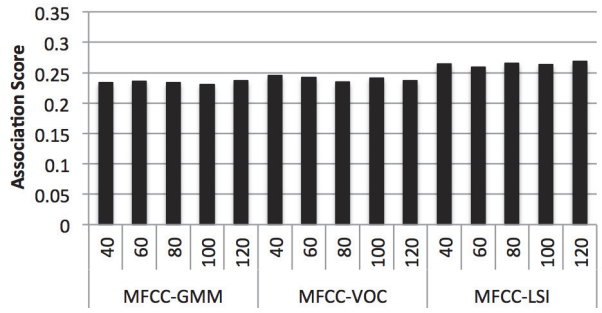


Figure 5: Effect of filters on MFCC

Figure 6 is in the same format as Figure 3, and shows MFS and MFCC when the best performing number of filters are used for each model. We do not show VOC because neither representation was improved by using more filters. Adding more filters improved the MFCC-LSI model, but is still outperformed by MFS-LSI.

The solid black line in Figure 6 shows MFS(40)-GMM, and the solid grey line shows MFS(120)-GMM. The MFS(40)-GMM results are included in the Figure to illustrate the improved recommendation quality achieved by increasing the number of filters for MFS-GMM. For the first 5 recommendations, MFS-GMM is now significantly better than MFCC-GMM. In comparison, there is only a small improvement of MFCC-GMM through increasing the filters used.

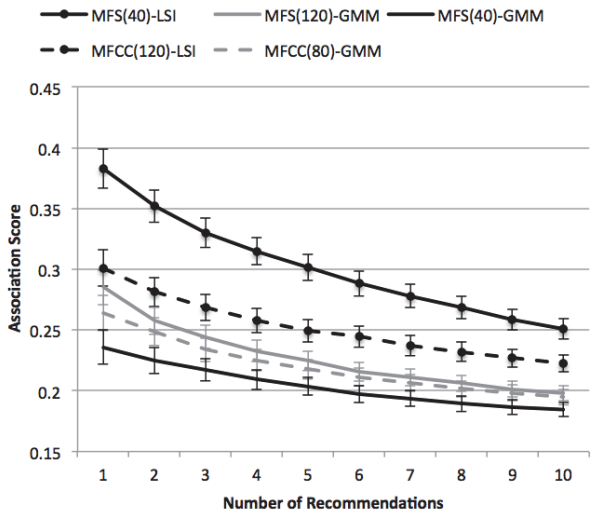


Figure 6: Comparison of best MFS and MFCC by model

Conclusion

The entire mel-frequency spectrum is important when describing the texture of music. When our MFS representation is used, all of the texture information is available to the models we use, leading to improved recommendation quality over MFCC. When MFCC is used, the DCT provides a summarised description of the mel-frequency spectrum, which does not allow a recommender model to learn what is important. Our results show that by not using the DCT, MFS achieves significantly better recommendation quality for each of the three models evaluated.

Traditional music texture representations are extracted with a standard number of filters, and therefore resolution. Our results have shown however that to extract a more meaningful music-inspired texture representation, one must also consider how the textures will be modelled. This link between representation and model is important, and there is no single MFS resolution which is optimal for the three models we have evaluated.

In each of the recommender models presented, the behaviour for MFS is more predictable than MFCC. With GMM more filters are best because the GMM is able to describe the information more meaningfully than DCT. With VOC there is no significant difference, and for LSI fewer filters are best. With MFCC, there are no relationships which emerge between the number of filters used and the recommender model.

The LSI model clearly outperforms both MFS and GMM for texture-based music recommendation. However, both VOC and GMM are commonly found in hybrid recommender systems. Future work therefore will explore how our novel approach to music texture contributes to hybrid recommender systems. It is hoped that by providing a stronger, predictable and robust texture component, increased recommendation quality may be achieved using hybrid representations.

References

Aucouturier, J.; Pachet, F.; and Sandler, M. 2005. The way it sounds: Timbre models for analysis and retrieval of music signals. *IEEE Transactions on Multimedia* 7(6):1028–1035.

Celma, O. 2010. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer-Verlag.

Ellis, D.; Whitman, B.; Berenzweig, A.; and Lawrence, S. 2002. The quest for ground truth in musical artist similarity. In *Proc. International Symposium on Music Information Retrieval ISMIR-2002*, 170–177.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I. 2009. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter* 11(1):10–18.

Hershey, J., and Olsen, P. 2007. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume IV, 317–320.

Hicklin, J.; Moler, C.; Webb, P.; Boisvert, R.; Miller, B.; Pozo, R.; and Remington, K. 2000. Jama: A java matrix package. *URL: http://math. nist. gov/javanumerics/jama*.

Horsburgh, B.; Craw, S.; Massie, S.; and Boswell, R. 2011. Finding the hidden gems: Recommending untagged music. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2256–2261.

Lee, C.-H.; Shih, J.-L.; Yu, K.-M.; and Lin, H.-S. 2009. Automatic music genre classification based on modulation spectral analysis of spectral and cepstral features. *Multimedia, IEEE Transactions on* 11(4):670 –682.

Li, T., and Chan, A. 2011. Genre classification and the invariance of mfcc features to key and tempo. *Advances in Multimedia Modeling* 317–327.

Logan, B. 2000. Mel frequency cepstral coefficients for music modelling. In *International Symposium on Music Information Retrieval*.

Mermelstein, P. 1976. Distance measures for speech recognition, psychological and instrumental. *Pattern Recognition and Artificial Intelligence* 116:91–103.

Pachet, F., and Aucouturier, J. 2004. Improving timbre similarity: How high is the sky? *Journal of negative results in speech and audio sciences* 1(1).

Sigurdsson, S.; Petersen, K. B.; and Lehn-Schiler, T. 2006. Mel frequency cepstral coefficients: An evaluation of robustness of MP3 encoded music. In *Proceedings of the Seventh International Conference on Music Information Retrieval*, 286–289.

Slaney, M. 1998. Auditory toolbox. *Interval Research Corporation, Tech. Rep* 10:1998.

Stevens, S.; Volkmann, J.; and Newman, E. 1937. A scale for the measurement of the psychological magnitude pitch. *Journal of the Acoustical Society of America*. 8(3):185–190

Yan, Q.; Zhou, Z.; and Li, S. 2012. Chinese accents identification with modified mfcc. In Zhang, T., ed., *Instrumentation, Measurement, Circuits and Systems*, volume 127 of *Advances in Intelligent and Soft Computing*. Springer Berlin / Heidelberg. 659–666.

Yoshii, K.; Goto, M.; Komatani, K.; Ogata, T.; and Okuno, H. 2008. An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model. *Audio, Speech, and Language Processing, IEEE Transactions on* 16(2):435–447.

Zheng, F.; Zhang, G.; and Song, Z. 2001. Comparison of different implementations of MFCC. *Journal of Computer Science and Technology* 16(6):582–589.

Cold-Start Music Recommendation Using a Hybrid Representation

Ben Horsburgh
IDEAS Research Institute
Robert Gordon University
Aberdeen, Scotland
b.horsburgh@rgu.ac.uk

Susan Craw
IDEAS Research Institute
Robert Gordon University
Aberdeen, Scotland
s.craw@rgu.ac.uk

Stewart Massie
IDEAS Research Institute
Robert Gordon University
Aberdeen, Scotland
s.massie@rgu.ac.uk

1. INTRODUCTION

Digital music systems are a new and exciting way to discover, share, and listen to new music. Their success is so great, that digital downloads are now included alongside traditional record sales in many official music charts [10]. In the past listeners would rely on magazine, radio, and friends reviews to decide on the music they listen to and purchase. In the internet age, this style of finding music is being superseded by music recommender systems.

The shift from listening to hard copies of music, such as CDs, to online copies like MP3s, presents the interesting new challenge of how to recommend music to a listener. In such recommender systems, a user will typically provide a track that they like as a query, often implicitly as they listen to the track. The system must then provide a list of further tracks that the user will want to listen to. Many websites exist that provide such recommender systems, and many of the systems provide very good recommendations. However, there are still scenarios that these systems struggle to handle, and where recommendations can be unreliable.

Online music systems allow users to tag any track with a free-text description. A recommender system can then determine the similarity between tracks based on these tags, and make recommendations. However, when a track is new to the system it will have no tags. This means that the track is never recommended, and in turn, the track is very unlikely to be tagged. Turnbull et. al [11] show that social tags tend to be very sparse, and that a huge popularity bias exists. This is further confirmed by data released by Last.fm [7] as part of the million song dataset [3]: from a vocabulary of over 500000 tags, each track, on average, has only 17 tags; 46% of tracks have no tags at all.

This scenario is often referred to as the cold-start problem; the results of which means large volumes of music are excluded from recommendations, even if they may be an excellent recommendation. The aim of our hybrid representation is to reduce the effects of the cold-start problem, therefore increasing the recommendation quality of the overall system.

Digital Economy All Hands Conference '12 Aberdeen, UK
.

2. RELATED WORK

Effects of cold-start can be reduced by learning tags for a track, a task referred to as auto-tagging [2, 9]. Most auto-tagging systems first extract audio features for many short time-frames within a track, and then model this bag-of-frames. A growing trend has been to use Gaussian Mixture Models [1], which describe the important clusters within a tracks' bag-of-frames. Tags are modelled in a similar manner, and correlations between the audio and tag models are used to learn auto-tags.

Hybrid representations are also becoming a prominent way of addressing the cold-start problem [8, 12]. In previous work we defined a representation that incorporates both audio and tag features, and generalises the representation using latent-semantic analysis (LSA) [5]. In the LSA concept space, tracks without tags belonged to concepts with tags. This approach increased the inclusion of cold-start tracks in recommendations.

Hybrid methods are good for cold-start, but they are often harmful when a track is well-tagged. The reason for this is the direct inclusion of content in the representation; content is a much weaker representation than tags.

3. HYBRID TAG REPRESENTATION

We propose that content should be used to strengthen other representations, but should not be integrated into other representations. Our new hybrid representation therefore does not include the content representation directly, but instead uses it to learn a stronger representation when appropriate.

To construct our hybrid representation we first define a content-based recommender system. This is then used to learn a pseudo-tag representation, which must achieve better recommendation quality than the content-based method. Our hybrid representation is a dynamic combination of both tags and pseudo-tags. In this section we will describe each representation.

Content. A content representation for a track is created by extracting our music-inspired texture representation Mel-Frequency Spectrum (MFS) [4] from the track audio. For each track, we first extract one MFS texture-vector for every 186ms of audio, and then compute a mean texture-vector. Finally, LSA is used to generalise all of the mean texture-vectors in our collection. Using the generalised MFS vectors, we are now able to provide content-based recommendations by computing the Euclidean distance between tracks' representations; in our evaluation we will refer to this method as **content**.

Pseudo-tags. Recommendation using the content method provides a list of tracks with similar content to a query track. To generate pseudo-tags, we take advantage of this list of similar tracks, and any tags which represent them. Given a list of N nearest neighbours of a track, we compute a pseudo-tag vector p as a weighted sum of the nearest neighbours tag-vector weights,

$$p_i = \sum_{n=1}^{N} \left(1 - \frac{n-1}{N}\right) t_i \quad (1)$$

where n is the position of the track in the nearest neighbour retrieval, and N is the number of nearest neighbours retrieved. t_i is the term-frequency of the ith tag in the tag vector of the track at position n. The bracketed part of the equation acts as a weighting function, giving a higher weight to tracks that are most similar. Recommendations made using these pseudo-tag vectors will be referred to as the **pseudo-tag** method.

In our final recommendation approach, we do not want to use only pseudo-tags, since these ignore tags already present in the collection. We therefore integrate the pseudo-tags we learn with the existing tag-vector of each track. However, we also do not want to decrease the effectiveness of existing track representations that do not suffer from cold-start. When our recommender system is using a tag-only representation, we will refer to this as the **tag** method.

Hybrid. Effective use of pseudo-tags can be achieved by creating a tag + pseudo-tag hybrid, which balances the influence of both tags and pseudo-tags. We therefore determine the number and weighting of pseudo-tags that we use based on how well-tagged a track is. If a track suffers from cold-start, many pseudo-tags will be used, and they will be given a high weighting. Conversely, if a track is well-tagged, very few pseudo-tags will be used, and they will be given a low weighting. Our final hybrid representation h is computed as

$$h_i = \alpha p_i + \beta t_i \quad (2)$$

where α is the weight of pseudo-tag vector p, and β is the weight of tag-vector t. i is the index of the tag being computed. When limiting the number of pseudo-tags being used, we set the weights of unwanted pseudo-tags to 0. The exact thresholds and weightings that we used were learned from a training set in our collection. We will refer to our hybrid representation as **hybrid**.

4. EVALUATION

The objective of our evaluation is to measure the performance of hybrid recommendation, and to examine the effect our representation has on cold-start tracks. A single track is provided as a query, and we measure the quality of recommendations made by our recommender system.

The dataset we are using contains 3174 tracks by 764 artists, split across 12 distinct super-genres. We collect the tag representations of each track using the Last.fm Audioscrobbler API [6]. This API provides a normalised tag-frequency vector, where the most popular tag for any given track has a weight of 100. The average number of tags assigned to a track in our collection is 34, with a standard deviation of 24.4. The maximum number of tags a track has is 100. Cold-start is also present in our collection, with 25% of tracks having fewer than 10 tags, and 3% of tracks having no tags.

We use 10-fold cross validation, and each track occurs in exactly one fold. For each query, our recommender system provides 10 recommendations. To measure the quality of these recommendations we use the association score measure, which we developed in previous work [5]. The data used to calculate the association score was collected for 175,000 Last.fm users over a period of 2 months. This score estimates the relationship between tracks based on the proportion of listeners who liked both tracks, computed as

$$\text{association}(s_i, s_j) = \frac{\text{likes}(s_i, s_j)}{\text{listeners}(s_i, s_j)} \quad (3)$$

where s_i and s_j are the query and recommended tracks. $\text{listeners}(s_i, s_j)$ is the number of people who have listened to both track s_i and track s_j, and $\text{likes}(s_i, s_j)$ is the number of people who have liked both s_i and s_j. A score of 0 indicates no evidence of users liking a recommendation, and a score of 1 indicates that all listeners of the query and recommendation also liked both the query and the recommendation, that is, the recommendation is of a high quality.

5. RESULTS

We first evaluate the performance of our pseudo-tag representation, to validate that it is meaningful. Figure 1 compares the recommendation quality of pseudo-tags and content. The horizontal axis shows the number of recommendations made, and the vertical axis shows the association score. The values shown are the mean average association score at the given number of recommendations, and error bars show 95% confidence.

Content-based recommendation is illustrated by the grey dashed line, and pseudo-tags by the black dashed line. When 1 recommendation is made, both methods are comparable; when 2 or more recommendations are made, pseudo-tags generate significantly better recommendations than content. While pseudo-tags are learned using content, they do not necessarily always describe content, and so tags, which describe other features, will also be present.

Figure 2 shows recommendation quality using tags and our hybrid representation, in the same format as the previous figure. For all numbers of recommendation, the hybrid-based approach offers significantly better recommendations than the tag-based method, and much better than either content or pseudo-tags alone. Increased quality is achieved because cold-start tracks now have a meaningful representation, and non-cold start tracks are still allowed to use the full descriptive power of their tag representation.

6. CONCLUSIONS

The motivation for this work was to increase the performance of a music recommender system by reducing the effects of the cold-start problem. We have shown that recommendations made using our pseudo-tag representation are of higher quality than when a state-of-the-art content-based representation is used. But, integrating these pseudo-tags with tags, to create our hybrid representation, provides significantly better quality recommendations than when using a tag-based representation. In future work we aim to examine in closer detail the effects our hybrid representation has on cold-start tracks.

7. REFERENCES

[1] L. Barrington, D. Turnbull, and G. Lanckriet. Auto-tagging music content with semantic multinomials. In *Proc. of Int. Conference on Music Information Retrieval*, 2008.

[2] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. *Machine Audition: Principles, Algorithms and Systems. IGI Publishing*, 2010.

[3] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[4] B. Horsburgh, S. Craw, and S. Massie. Music-inspired texture representation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 52–58, 2012.

[5] B. Horsburgh, S. Craw, S. Massie, and R. Boswell. Finding the hidden gems: Recommending untagged music. In *Twenty-Second International Joint Conference on Artificial Intelligence*, pages 2256–2261, 2011.

[6] Last.fm API. http://www.last.fm/api, May 2012.

[7] Last.fm dataset, the official song tags and song similarity collection for the Million Song Dataset. http://labrosa.ee.columbia.edu/millionsong/lastfm, may 2012.

[8] M. Levy and M. Sandler. Music information retrieval using social tags and audio. *IEEE Transaction on Multimedia*, 11(3):383–395, 2009.

[9] R. Miotto, L. Barrington, and G. Lanckriet. Improving auto-tagging by modeling semantic co-occurrences. In *Proc. ISMIR*, pages 297–302. 2010.

[10] Official Charts Company. http://www.officialcharts.com/chart-rules/, 2009.

[11] D. Turnbull, L. Barrington, and G. Lanckriet. Five approaches to collecting tags for music. In *International Society for Music Information Retrieval 2008*, pages 225–230, 2008.

[12] K. Yoshii and M. Goto. Continuous plsi and smoothing techniques for hybrid music recommendation. In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, 2009.

APPENDIX

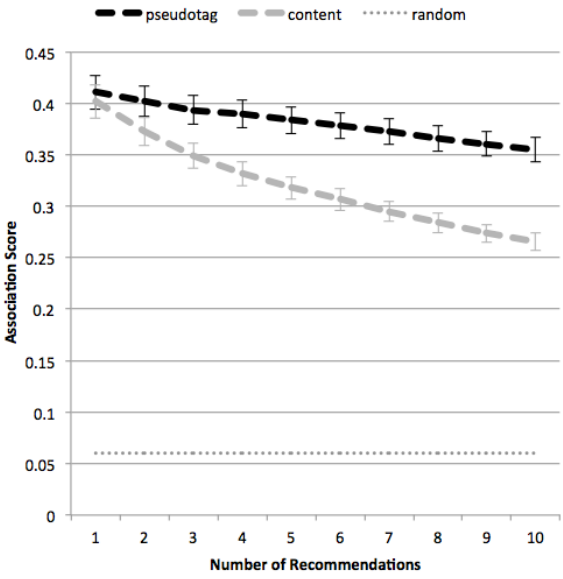


Figure 1: Pseudo-tag vs. Content Quality

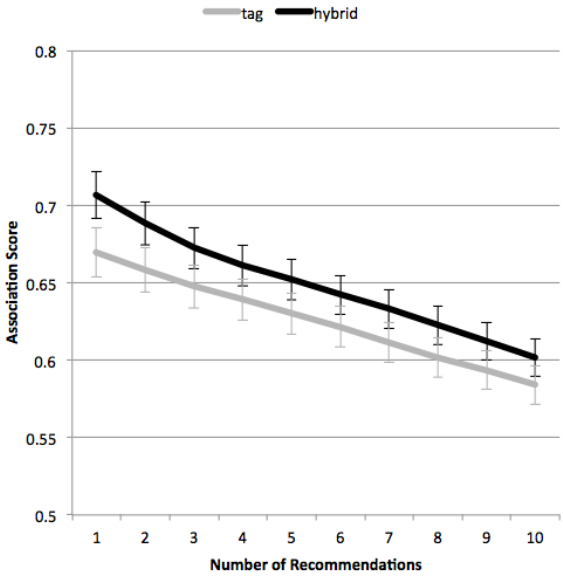


Figure 2: Hybrid vs. Tag Quality