Womrad 2011

Workshop on Music Recommendation and Discovery

Colocated with ACM RecSys 2011

Chicago, IL, USA

October 23, 2011

[1] http://creativecommons.org/licenses/by/3.0/

Organizing Committee

Amélie Anglade, SoundCloud
Òscar Celma, Gracenote
Ben Fields, Musicmetric
Paul Lamere, The Echo Nest
Brian Mcfee, Computer Audition Laboratory, University of California, San Diego

Program Committee

Claudio Baccigalupo, RED Interactive Agency
Dominikus Baur, LMU Munich
Mathieu Barthet, Queen Mary, University of London
Thierry Bertin-Mahieux, Columbia University
Sally Jo Cunningham, the University of Waikato
Zeno Gantner, University of Hildesheim
Fabien Gouyon, INESC Porto
Peter Knees, Johannes Kepler University
Daniel Lemire, Université du Québec
Mark Levy, Last.fm
Markus Schedl, Johannes Kepler University
Alan Said, Technische Universität Berlin
Doug Turnbull, Ithaca College

Preface

Welcome to WOMRAD, the Workshop on Music Recommendation and Discovery being held in conjunction with ACM RecSys.

WOMRAD 2011 is being held on October 23, 2011, exactly 10 years after Steve Jobs introduced the very first iPod. Since then there has been been an amazing transformation in the world of music. Portable listening devices have advanced from that original iPod that allowed you to carry a thousand songs in your pocket to the latest iPhone that can put millions of songs in your pocket via music subscription services such as Rdio, Spotify or Rhapsody. Ten years ago a typical personal music collection numbered around a thousand songs. Today, a music listener has access to millions of songs, drawn from all styles and genres and from all over the world. The seemingly infinite choice today's music listener faces can lead to a rich music listening experience, but only if the listener can find music that they want to listen to.

Traditionally, music recommender systems have focused on the somewhat narrow task of attempting to predict a set of new artists or tracks for purchase based upon an individual's purchase history. As music listeners spend more time interacting with multi-million song music collections, the need for tools that help listeners manage their listening will become increasingly important. Tools for exploring and discovering music especially in the long tail, tools for organizing listening, tools for creating interesting playlists will all be essential to the music listening experience. Other contexts such as groups listening, music learning are emerging as important aspects of recommendation and discovery.

The WOMRAD workshop focuses on next generation of music recommendation and discovery systems. Accepted papers fall into a number of categories:

- Social aspects of music discovery
- Semantics and recommendation
- Group recommendation
- Learning strategies for recommendation

We are pleased to offer this selection of papers and hope that it serves as evidence that their is much interesting and fruitful research to be done in the area of music recommendation and discovery. We offer our thanks to all of the authors who submitted papers to this workshop.

October 2011 The Organizers

Keynote Presentation

Building a Scalable Music Recommender
Douglas Eck, *Research Scientist, Google*

I will discuss techniques for content-based music recommendation at large scale. I'll focus on three steps in a music recommendation pipeline. First I'll present Stabilized Auditory Images (SAIs), acoustic representations based on human auditory processing. Next, I'll look at generating sparse, noise-resistent codes from SAIs using an ordinal coding method called Winner-Take-All (WTA). Finally, I'll describe a multi-class multi-label embedding algorithm, Wsabie for ranking tracks vis-a-vis training labels such as genre and style. The work I discuss is based on work published by a number of researchers at Google. Time permitting, I hope to present several audio demos to illustrate our approach.

Before joining Google in 2011, Douglas Eck was an Associate Professor in Computer Science at University of Montreal. His PhD research (Indiana University Computer Science, 2000) investigated the dynamics of model neurons in response to music-like rhythmic patterns. He went on to do work in computational music cognition, focusing on music generation with recurrent neural networks, meter perception, beat tracking and expressive music performance. More recently he has focused on large-scale machine learning approaches to music recommendation, including work in learning informative representations of music audio. At Google he has been working on the music recommender for Music Beta by Google. Douglas Eck doesn't like bios written in the third person, but since everyone else does it, he does it too.

Contents

Music Discovery with Social Networks

Cédric S. Mesnage
Faculty of Informatics
University of Lugano
Lugano, Switzerland
cedric.mesnage@usi.ch

Asma Rafiq
Centre for Digital Music
Queen Mary University of London
London, UK E1 4NS
a.rafiq@qmul.ac.uk

Simon Dixon
Centre for Digital Music
Queen Mary University of London
London, UK E1 4NS
simon.dixon@eecs.qmul.ac.uk

Romain P. Brixtel
University of Caen
France
rbrixtel@info.unicaen.fr

ABSTRACT

Current music recommender systems rely on techniques like collaborative filtering on user-provided information in order to generate relevant recommendations based upon users' music collections or listening habits. In this paper, we examine whether better recommendations can be obtained by taking into account the music preferences of the user's social contacts. We assume that music is naturally diffused through the social network of its listeners, and that we can propagate automatic recommendations in the same way through the network. In order to test this statement, we developed a music recommender application called Starnet on a Social Networking Service. It generated recommendations based either on positive ratings of friends (*social recommendations*), positive ratings of others in the network (*non-social recommendations*), or not based on ratings (*random recommendations*). The user responses to each type of recommendation indicate that social recommendations are better than non-social recommendations, which are in turn better than random recommendations. Likewise, the discovery of novel and relevant music is more likely via social recommendations than non-social. Social shuffle recommendations enable people to discover music through a serendipitous process powered by human relationships and tastes, exploiting the user's social network to share cultural experiences.

Categories and Subject Descriptors

H.5.5 [**Information Systems**]: Information Interfaces and Presentation—*Sound and Music Computing*; H.4.3 [**Information Systems Applications**]: Communications Applications—*Internet*; H.3.3 [**Information Systems**]: Information Storage and Retrieval—*Information Filtering, Selection Process*; H.3.4 [**Social Networking**]:

WOMRAD 2011 2nd Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2011 (Chicago, US)

General Terms

Design, Experimentation, Performance

Keywords

Music discovery, music recommender systems, social networking services

1. INTRODUCTION

An interesting mechanism of music discovery occurs when family and friends recommend each other music that they discover. The emergence of Social Networking Services (SNS) and Web Music Communities (WMC) provide us the opportunity to develop music recommender applications to support this mechanism. SNS are becoming increasingly popular means for people to socialise online. Music is playing a similar role on these platforms as in real life social networks. It is shared, discussed, recommended and discovered with social contacts. It is noteworthy that music information seeking behaviour has been indicated as 'highly social' [5]. This social aspect of music should be incorporated in current music recommender systems. WMCs like Last.fm[1], Pandora[2] and Ping[3] are playing a vital role in helping music listeners to build relationships with similar music-listeners and get recommendations based on their current music collections. WMCs are very popular among music fans.

Music discoveries often result from passive behaviour [5]. In [2] authors also indicated that music discovery was seldom a conscious activity until the project participants were given the task of writing diaries when they encountered new music. Therefore, it is quite likely that many people are interested in discovering music but not actively seeking for it on WMCs. Possibly due to the region-specific content access restrictions, since sites such as Pandora can only be used within U.S. territories and Last.fm radio is not available without paid subscription to countries other than UK, US and Germany. On the other hand, SNS such as Facebook[4] allow social interaction with family and friends around the globe.

In this work, we model that music discoveries take place

[1] `http://www.last.fm`
[2] `http://www.pandora.com`
[3] `http://www.apple.com/itunes/ping`
[4] `http://www.facebook.com`

via natural diffusion of music through social networks or randomly. An experiment was conducted to reproduce this process so that we could analyse how people respond to the recommendations. The recommendations arising from such processes are either randomly picked from the pool of tracks of the data set or collaboratively from the tracks recommended by other people on the SNS. A successful music discovery occurs when the user of the application likes a track that s/he has never heard before.
This paper has been divided into 7 sections; section 1 is the introduction, section 2 elaborates the rationale behind this research experiment. In section 3, we discuss the methodology. In section 4, the results are presented. In sections 5, 6 and 7, limitations are discussed, the research work is concluded, and future work is proposed respectively.

2. BACKGROUND AND MOTIVATION

Research on finding new music shows that music discovery often occurs with the personal acquaintances playing music to the respondents, and that social networks continue to play a vital role in music discovery in the digital age [10]. Our social contacts may influence our music preferences [3]. Social context plays a significant role in improving music recommendation algorithms [7]. Music can both reflect and define social identity and membership in a given subculture [5]. Information retrieval from social media aids the collection, storage and review of music of the users. It presents opportunities for improved music recommender systems incorporating music preferences of the user's social contacts. An online survey conducted by Entertainment Media Research Company (EMRC) and Wiggin (2009) with 1,608 participants from the United Kingdom, indicated that social networking sites are frequently used for music streaming [1]. Although Last.fm uses collaborative recommendation algorithms, it does not explicitly provide an option to restrict recommendations to the user's social contacts rather than the whole WMC. Interestingly, Pandora has recently attempted to add the feature of "Music Feed" on Pandora One, which shows the activities of friends such as likes, tracks they are listening to and comments. It is a similar concept to Ping but is only available to the paid subscribers and is currently in testing phase [9]. The number of active users on Facebook[5] outnumbers any of the WMCs mentioned above by a significant margin. Therefore, it is more likely to find real life friends on Facebook as compared to the WMCs, forming another motivation to conduct our experiment on Facebook. In [4] authors show that collaborative filtering based on social relationships and tags outperforms standard information retrieval techniques by running simulations on users' listening history. Other research has shown that music listeners sometimes enjoy randomly ordered recommendations [6]. We approach the problem with a different methodology by conducting a live experiment in which users listen and rate the recommendations.

3. METHODOLOGY

The aim of the experiment is to test that discoveries are diffused through the social network. The problem is treated as a recommendation problem defined as follows: given a pool of items, select an item that the subject has not heard and that is relevant to her/him. A collaborative recommendation makes use of items rated previously by other subjects to choose the item to be recommended. If collaborative recommendations based on ratings by people from the social network of the subject lead to more successful recommendations than ratings from people not in the subject's social network, then there is an indication that social recommendations are more appropriate for collaborative recommendations.

3.1 Experimental setting

The idea of the social shuffle is to recommend tracks (see Section 3.2 for more details) and diffuse discoveries through the social network. Figure 1 shows an example of this process when a recommendation is posted by a user in her/his SNS. In this case, Joe gets a random recommendation, gives it a 4 star rating, the track is then diffused to his social network. If a friend of Joe gives a high rating to the same track, it will be diffused to her network as well (in this case, the example of Alice). If Joe's friend does not enjoy the track and gives it 2 or less rating then the social diffusion stops and here Aleks' friends will not get this recommendation.

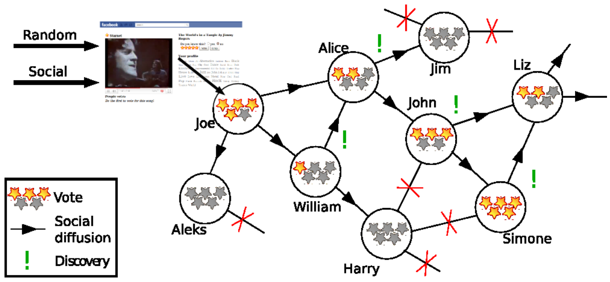


Figure 1: Example of the social shuffle principle.

Each time a track is recommended, the subject of the experiment rates the track on a 0 to 5 Likert scale; a psychometric scale introduced by Rensis Likert [11] visually represented as stars, where selection of 0 indicates dislike and 5 indicates favourite tracks.

3.1.1 Dataset

In order to have a representative pool of available tracks for random selection, the dataset was built from Last.fm containing a million tracks fetched through their API. We explored a subset of the Last.fm of about 300,000 people and their friends. The tag profile of each user was fetched, giving us the list of tags they used to organize tracks, yielding a set of more than 300,000 unique tags. For each tag the API allows to fetch the top 50 most popular tracks tagged with this tag. Only the set of tracks is relevant for this study.
On Facebook, we started by fetching the network of the first author, his friends and friends of friends interested in participating in this experiment. Many people share music using Youtube[6] by posting links on Facebook. The advantage of using music videos from Youtube is that the full track can be played (if the video is available), which is not the case for public users of Last.fm for instance which limits the playback to 30 seconds. The limitation pertaining to Youtube videos is that sometimes these are blocked in various countries and

[5]http://www.facebook.com/press/info.php?statistics

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

removed for copyright violations. The application detects when such errors occur and does not recommend that video afterwards. On the Youtube API, videos were searched for each track with the artist name and track title. We selected the most popular video for each track and restricted to fetch the videos tagged as “Music” only. Using this process about a quarter of the tracks from Last.fm had a video on Youtube, totaling around 252,000 tracks.

3.1.2 *Facebook application*

Starnet is a Facebook application fed by ratings on random selections. A positive rating spawns diffusion through the user’s social network, i.e. the shuffle recommendation becomes social. The interface (Figure 2) consists of the current track description (title and artist name), the music video associated, a tag cloud of the user’s profile and a rating form consisting of 5 stars. Also, it has “next” button to play the next track and a “bail” button which sets the stars to 0 and plays the next track. The user is asked to indicate whether the track is already known, in order to learn whether it is a music discovery. We assume it is a reasonable estimate for true unknown tracks.

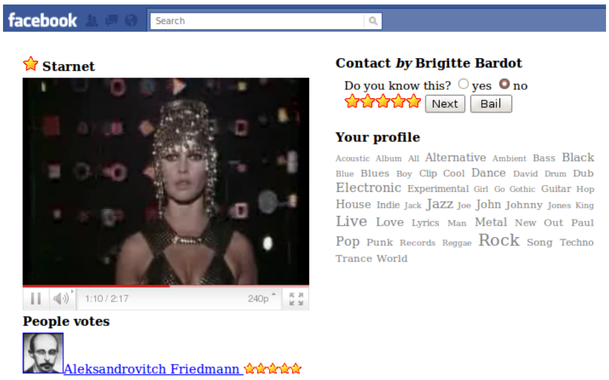


Figure 2: Screenshot of Starnet Application on Facebook.

3.1.3 *Subjects*

The subjects are the people who used the Starnet application. A total of 68 subjects participated in this experiment and allowed access to their social profiles. Each subject acted as his/her own control group by getting and rating random recommendations. In about 4 months (from the 29th June 2010 to the 18th October 2010), 31 subjects made 4966 ratings using the Starnet Application. Participation of other 37 subjects was insignificant in terms of ratings.

3.2 Recommendation Strategy

The recommender system produced the following types of recommendations:

- Random recommendations: The random selection is a query which selects tracks that have not been rated by the subject and orders them randomly.

- Collaborative recommendations: Selects a track randomly from the set of tracks that have been rated with a rating above average (i.e. greater than 2 stars) and not yet rated by the subject. The collaborative recommendation can be social or non-social.

- Social recommendations: The social recommender selects a track randomly, from the tracks that have been rated by friends of the subject, with a rating superior to 2 stars.

- Non-social recommendations: The non-social recommender selects a track randomly, from the tracks that have been rated by people who are not friends of the subject, with a rating superior to 2 stars.

The likelihood for selecting random recommendation, social recommendation and non-social recommendation is 0.5, 0.25 and 0.25, respectively.

4. RESULTS

This section presents the results from the analysis of the ratings made by the subjects of the experiment. The success of a recommendation model to discover new and relevant tracks for a subject can be evaluated without further user input.

Discoveries.

A measurement of the success of a recommendation model to discover new and relevant tracks for a subject is the ratio of already discovered tracks and all rated tracks. Figure 3 represents how these sets relate for a particular user.

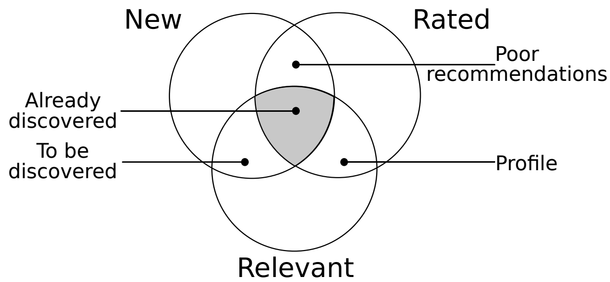


Figure 3: Discovery recall diagram.

Histograms are used to compare distributions of ratings. The x axis represents number of stars/rating and the corresponding percentage of ratings on the y axis. Heat maps are used to show how the ratings of a subject relates to ratings on the same tracks from friends and non-friends. The ratings for social and non-social recommendations of known and unknown tracks are compared, showing that social recommendations give better ratings than non-social ones.

Figure 4 shows the ratings for the tracks known to the subject and for those specified as unknown, respectively represented as black and white bars. This shows a clear distinction between known and unknown tracks, most unknown tracks are disliked whereas most known tracks are liked. In this figure we looked indifferently at all the recommenders, in the next figure we differentiate between collaborative and random recommendations.

Figure 5 represents the proportions of ratings for collaborative and random recommendations. Collaborative recommendation outperforms random recommendation as 45% of

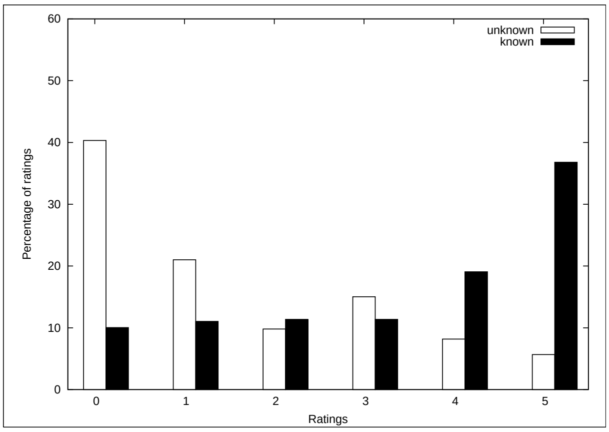


Figure 4: Histogram of ratings for known and unknown tracks across all the users.

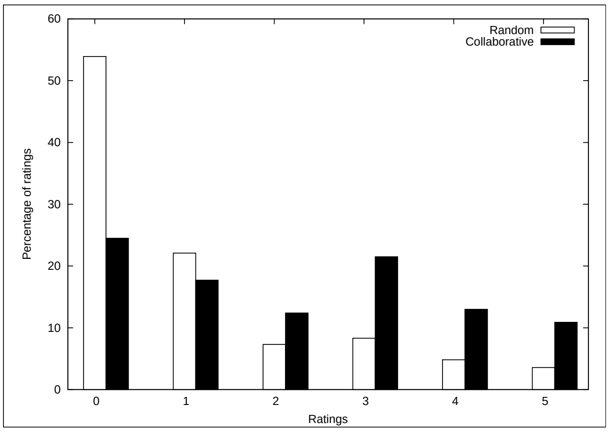


Figure 5: Histogram of ratings for collaborative and random ratings.

its recommendation get 3 star or above ratings. The percentage drops to around 17% in the case of random recommendations. In this visualization, an increased number of ratings above the threshold of 3 stars is evident for collaborative recommendations. This is due to the way collaborative recommendations are made: only tracks which get a rating superior to 2 stars from other users are used as collaborative recommendations. This shows that there is some consistency in users' ratings. We now look only at the collaborative ratings dividing them between social and non-social recommendations.

Figure 6 represents the ratings from two types of collaborative recommendations. The social and non-social ratings have different distributions indicating that subjects react differently to recommendations coming from their friends than from people they do not know. The subjects are not aware of the source of the recommendation. More than 47% of social recommendations get 3 or more stars as compared to the non-social ones, (33%). In this visualization, we see that social recommendations tend to lead to better ratings, but we mix known and unknown tracks.

Figure 7 represents the ratings of collaborative recommendations on known and unknown tracks. 94% of the ratings state that the track is unknown to the subject, which is why the left histogram on Figure 7 is similar to the histogram of Figure 6. In general, the social recommendations lead to more unknown good recommendations (46% at least 3 stars against 34%) than the non-social ones and less bad recommendations (40% at most 2 stars against 60%). The histogram on the right shows ratings where the subject specified she knew the track. Although, the overall rating distributions are very different, the same trend of higher ratings for social recommendations is evident.

Figure 8 shows two heat maps. Each square represents the proportion of ratings made by all users of the application on the same tracks for each rating. The left hand heat map represents the relation between ratings of people who are not friends and the right one the relation between ratings of people who are friends.The contrasted regions on both heat map shows that people agree on ratings on the same tracks. Interestingly, the social heat map is more contrasted, showing that friends agree more with their ratings than people who are not friends.

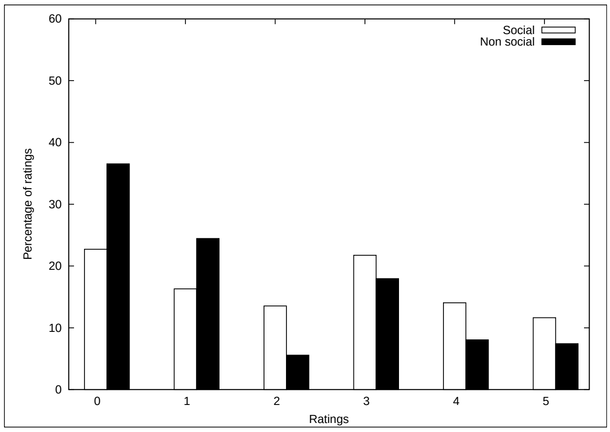


Figure 6: Histogram of social and non-social ratings.

5. LIMITATIONS

Our results show that people tend to prefer the music that their friends prefer. One of the limitations of this work is that it is based on the assumption that the relationships in SNS or WMC are with real life friends and family which might not be true for some subjects. Therefore, they might not be very good source for music recommendation or discovery in that case, as they might not share user's music taste. Results suggest otherwise.

Studies on music behaviour also suggest that sometimes youngsters try to distance themselves from the previous generation [8] in which case, social recommendations from older members of the social contacts might not work well. So, there should be an option for tuning the users' social network for music recommendation, when such a feature is integrated in a music recommender system.

6. CONCLUSION

The analysis of the results of the experiment on social diffusion. lead us to the following conclusions:

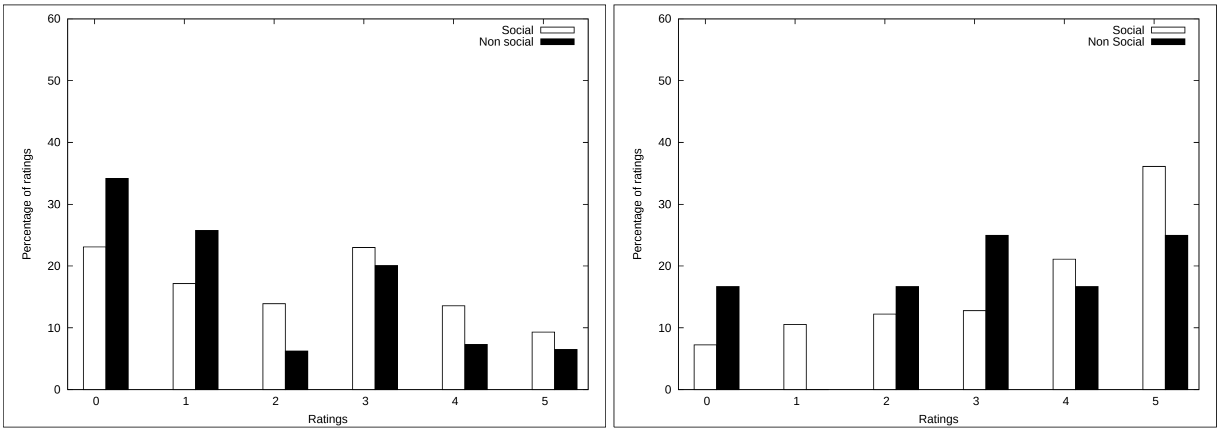


Figure 7: Histograms of ratings of social and non social recommendations on tracks unknown (left) and known (right).

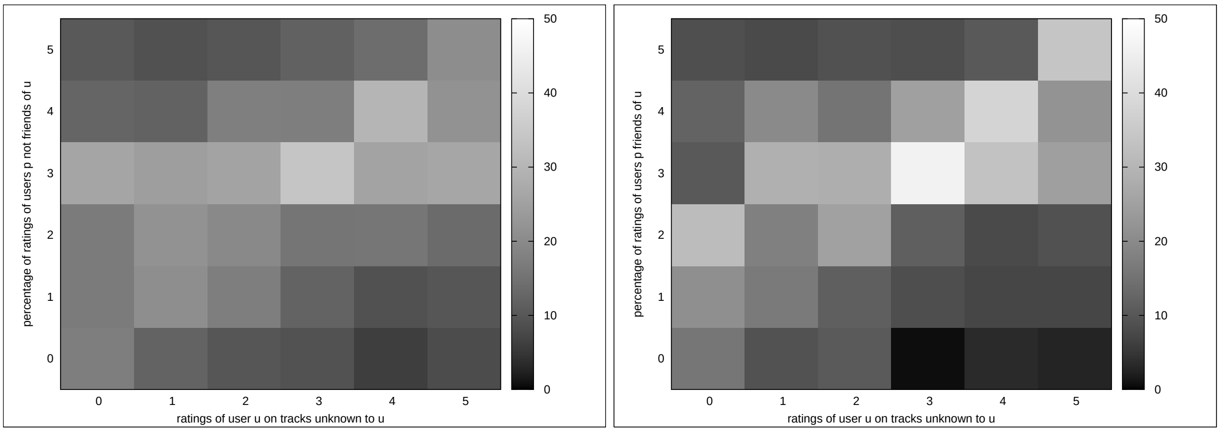


Figure 8: Heat map of ratings made on the same tracks, by people who are not friends and did not know the track on the left and people who are friends on the right.

- Social recommendations were preferred by the users over the non-social recommendations which indicates that people tend to share music taste with their friends on Social Network Services than with people who are not their friends.
- Recommended tracks that were known are mostly liked whereas recommended tracks that are unknown are mostly disliked.
- Collaborative recommendations lead to better recommendations than random recommendations.

These conclusions support the view that social diffusion is a good mechanism for music recommendation and discovery. It is anticipated that it shall form the foundation for the framework of better music recommender systems combined with social media.

7. FUTURE WORK

An improved recommendation system could make use of the genres (or tags) of the tracks previously rated by the subject. We have been working on a different input for music videos. We are now fetching the Youtube videos posted on Facebook by registered people to the Starnet application and their friends. This changes the settings of the application and requires us to focus on interaction mechanisms for people to explore a social network. The new application shall allow users to add and remove social contacts to their network, to tune their personalised social radio.
Another application of interest is a music recommender system that recommends music based on the type of event and music preferences of the people attending it. The initial prototype of the system is available as a Facebook Application[7]. People post events (such as party, wedding, etc.) on SNS. Analysis of music taste of the attendees of the events can enable the event organiser to make better decisions on what type of music shall be enjoyable for most of the attendees. However, event categorisation and determination of suitability of music within that particular category can be a challenge.

[7]`http://apps.facebook.com/music_valley/`

8. REFERENCES

[1] Z. Baker. *Social Networking and Music: the impact of myspace as a Digital Platform for Discovery and Distribution*. Bachelor's thesis, Cardiff School of Journalism, Media and Cultural Studies, Cardiff University, 2009.

[2] S. Cunningham, D. Bainbridge, and D. McKay. Finding new music: A diary study of everyday encounters with novel songs. pages 83–88. Austrian Computer Society, International Society for Music Information Retrieval, 2007.

[3] T. DeNora. *Music in Everyday Life*, pages 4–8. Cambridge University Press, 2000.

[4] I. Konstas, V. Stathopoulos, and J. Jose. On social networks and collaborative recommendation. In *SIGIR '09: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 195–202, New York, NY, USA, 2009. ACM.

[5] A. Laplante. *Everyday life music information-seeking behaviour of young adults: an exploratory study*. PhD thesis, McGill University, Montreal, Canada, 2008. pages 4-65.

[6] T. Leong, F. Vetere, and S. Howard. The serendipity shuffle. In *OZCHI '05*, pages 1–4. Computer-Human Interaction Special Interest Group, 2005.

[7] D. McEnnis and S. Cunningham. Sociology and music recommendation systems. pages 185–186. Austrian Computer Society, 8th International Conference on Music Information Retrieval, 2007.

[8] A. Minks. Growing and grooving to a steady beat: Pop music in fifth-graders' social lives. *Yearbook for traditional music*, 31:77–101, 1999.

[9] M. Siegler. Dancing to the drumbeat of HTML5, New Pandora is brilliant, beautiful. Website, 2011. `http://techcrunch.com/2011/07/12/new-pandora/`.

[10] S. Tepper and E. Hargittai. Pathways to music exploration in a digital age. *Poetics*, 37(3):227–249, 2009.

[11] J. Uebersax. Likert scales: dispelling the confusion. Website, 2006. `http://john-uebersax.com/stat/likert.htm`.

Music recommendation for music learning: Hotttabs, a multimedia guitar tutor

Mathieu Barthet, Amélie Anglade, Gyorgy Fazekas, Sefki Kolozali, Robert Macrae
Centre for Digital Music
Queen Mary University of London, Mile End Road, London E1 4NS
{mathieu.barthet, amelie.anglade, gyorgy.fazekas, sefki.kolozali, robert.macrae}@eecs.qmul.ac.uk

ABSTRACT

Music recommendation systems built on top of music information retrieval (MIR) technologies are usually designed to provide new ways to discover and listen to digital music collections. However, they do not typically assist in another important aspect of musical activity, music learning. In this study we present the application Hotttabs, an online music recommendation system dedicated to guitar learning. Hotttabs makes use of The Echo Nest music platform to retrieve the latest popular or "hot" songs based on editorial, social and charts/sales criteria, and YouTube to find relevant guitar video tutorials. The audio tracks of the YouTube videos are processed with an automatic chord extraction algorithm in order to provide a visual feedback of the chord labels synchronised with the video. Guitar tablatures, a form of music notation showing instrument fingerings, are mined from the web and their chord sequences are extracted. The tablatures are then clustered based on the songs' chord sequences complexity so that guitarists can pick up those adapted to their performance skills.

Categories and Subject Descriptors

H.5.5 [**Sound and Music Computing**]: [Signal analysis, synthesis, and processing]; H.3.5 [**On-line Information Services**]: [Web-based services]; H.5.1 [**Multimedia Information Systems**]: [Video (e.g., tape, disk, DVI)]

Keywords

computer-assisted guitar tuition, automatic chord recognition, guitar tab recommendation, online music service, multimodal, hotttness measure (The Echo Nest), music video tutorial (YouTube), tag cloud, user interface

1. INTRODUCTION

The design of music recommendation systems exploiting context and/or content based information [4] has mainly been undertaken by considering music listening as the central end-user activity. Examples of such systems are the popular online music services Last.fm[1], Pandora[2], and Spotify[3], which provide new ways to experience and discover songs [1]. If, in this view, music recommendation models aim at satisfying listeners' needs and expectations, they discard other major actors of the chain of musical communication: performers. In this article, we present an online music recommendation system targeting music learning rather than music listening, therefore targeting performers rather than listeners.

Music education is one of the humanity subjects emphasised since ancient times [12]. Since the 1970s, many studies have been published in order to build computer-assisted instruction systems in various tasks of music education such as music theory, ear training, performance skills development, music composition or editing, music appreciation, musical instruments knowledge, and harmony. However, most of these systems use different approaches due to the interdisciplinary nature of the field [2, 5]. With the existing high-tech information era and the rapidly growing world wide web, it is easier to combine different musical instructions on a system to provide a good learning environment which not only integrates a variety of learning experiences for performers (e.g. textual materials, images, expert videos, and audio), but also allows individual performers to practice in less stressful conditions (e.g. the typical home setting) when compared to group-based practice [11].

Amongst musical instruments, the guitar stands out as being one of the most popular instruments in the world, with new players taking it up every day (e.g. guitar sales represent 50% of the whole musical instruments' market in France [10]). Amateur guitarists often seek new songs to play solo or with other musicians during a jam session. It is common to spend the whole time devoted to the practicing session trying to select a song adapted to one's musical skills and to find music notations in order to learn it. The proposed Hotttabs[4] application is an online guitar tuition system aimed at solving this problem by recommending popular songs to play and guiding guitarists in the learning process. Hotttabs uses a multimodal approach, relying on video tutorials, chord visualisations, and tablatures (commonly referred to

WOMRAD 2011 2nd Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2011 (Chicago, US)

[1] http://www.last.fm
[2] http://www.pandora.com
[3] https://www.spotify.com
[4] http://isophonics.net/hotttabs/

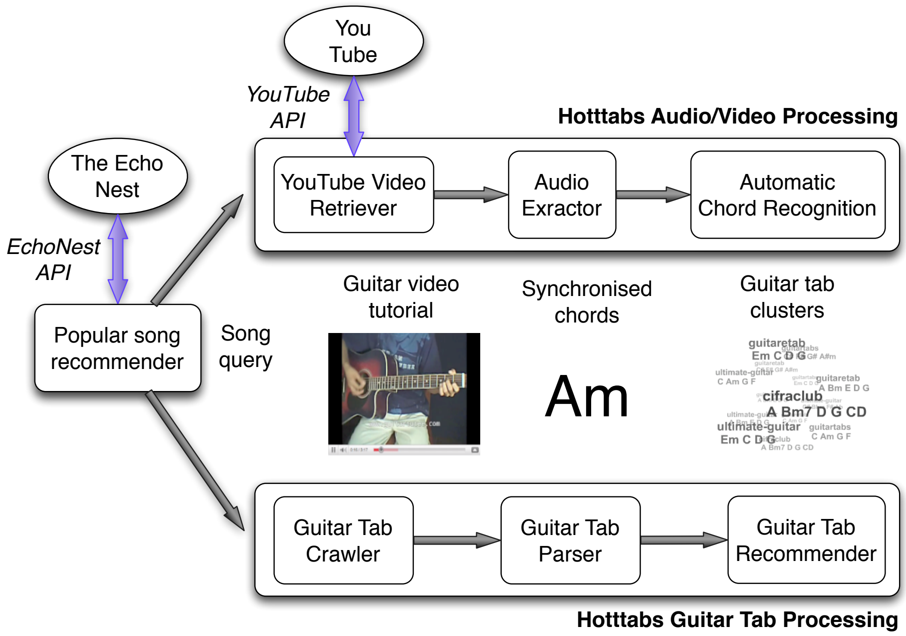


Figure 1: Hotttabs process flow chart. (API: Application Programming Interface).

as "tabs"), a form of music notation representing instrument fingering with numerical symbols rather than the musical pitches commonly used in scores.

The popularity of the guitar may be explained by several reasons: the great versatility of timbres that can be produced on acoustic or electric guitars make the instrument suitable for many different musical genres (blues, rock, reggae, jazz, classical, etc.), the simple accessibility to the instrument (guitars can be obtained at moderate costs and can be easily stored and carried away), and the possibility to learn songs regardless of prior music theory knowledge using tablatures. Since the range of pitches that can be produced on the guitar present some overlap between the various strings, notes of identical pitch can be played at several positions on the finger board (albeit with slightly different timbres due to the differences in the physical properties of the strings). One of the interests of the tablature notation is to alleviate the ambiguity on fingering by proposing an explicit solution. They can thus be considered as more effective than scores to assist beginners in guitar learning [16]. This may be one reason why guitar tabs are by far the most popular means of sharing musical instructions on the internet, largely surpassing online music sheet and MIDI score databases [13]. The Hotttabs application takes advantage of the richness of hand annotated tablatures provided on the web.

Recently, the number of guitar tuition applications for smartphones have blossomed (e.g. Killer Riffs[5], Touch Chords[6], Ultimate Guitar Tabs[7], TabToolkit[8], Rock Prodigy[9]) showing a real interest in new technologies devoted to enhancing music learning. However, most applications provide either short sections of songs, such as riffs, only the tablatures (without visual feedback showing how to play them), or predefined lists that may not contain current chart music.

In the proposed Hotttabs application, these issues are tackled by using The Echo Nest music platform[10] to retrieve the latest popular or "hot" songs based on editorial, social and charts/sales criteria, guitar video tutorials from YouTube[11], the online video sharing platform, and cutting-edge technologies in automatic chord recognition [15] and guitar tab parsing [13], to provide users with symbolic music information assisting them in the learning process. A flow chart showing the processes involved in the Hotttabs application is shown in Figure 1. The application comprises three main components: song recommendation, audio/video processing, and guitar tab processing.

The remainder of the article is organised as follows. In Section 2 we present the song recommendation method. In Section 3 we describe the audio and video processing. In section 4 the guitar tab processing is presented. Section 5 details the web application. In Section 6 we give some conclusions and perspectives on this work.

2. SONG RECOMMENDATION

The application recommends users a list of songs to practice consisting of the twenty most popular songs at the time

[5]http://itunes.apple.com/gb/app/killer-riffs/id325662214?mt=8
[6]http://itunes.apple.com/us/app/touchchords/id320070588?mt=8
[7]http://app.ultimate-guitar.com/ugt/iphone/
[8]http://agilepartners.com/apps/tabtoolkit/
[9]http://www.rockprodigy.com
[10]http://the.echonest.com/
[11]http://www.youtube.com/

of the query. These popular songs are obtained using the "hotttnesss" measure from The Echo Nest music platform. This measure which is expressed in the range [0;1] is based on editorial, social and charts/sales criteria. Editorial criteria rely on the number of reviews and blog posts that have been published about an artist in the last three weeks, providing an indicator of how much impact an artist has. Social criteria are derived from the total number of track plays the artist is receiving on a number of social media sites (for instance using statistics gathered on last.fm[12]) providing an indicator of how often people are listening to this artist. Charts/sales criteria are based on the appearance of the artist on various sales charts providing an indicator of how often people are purchasing music by this artist. A list of the twenty most popular artists is first retrieved. Then, the most popular song from each artist is selected. With such a process, the song recommender relies on a dynamic music chart directly influenced by listeners over the web, mobile or desktop applications, music consumers, and journalists.

3. AUDIO/VIDEO PROCESSING

The song learning methods underlying the application are based on a multimodal approach using audio, video, and symbolic (chord labels) feedback. We detail in this section how these modalities are exploited within the application.

3.1 YouTube guitar video tutorials retrieval

Music video tutorials offer a musical tuition alternative to music teachers since they allow one to see how a real performer plays while listening to the music. Furthermore, they often include spoken parts, giving extra information in how to perform the music or how the music is structured. YouTube provides a rich source of guitar video tutorials which are frequently updated with the latest popular songs by a large community of amateur and professional guitarists. Hotttabs filters the YouTube video database to retrieve relevant guitar video tutorials for the selected songs. To connect Hotttabs with YouTube we use Google's Data API Python client gdata[13] and request videos (YouTubeVideoQuery() function) containing the following keywords: "<song and artist> guitar chords learn".

3.2 Automatic chord recognition

Symbolic information representing music along with the video can facilitate the learning process. Furthermore in some video tutorials, the position of the player's fingers on the guitar neck cannot be seen. In order to tackle this issue, the audio tracks of the YouTube video tutorials are first extracted (using the FFmpeg converter[14]) and then processed with an automatic chord extraction algorithm. Hotttabs utilises the chord recognition algorithm described in [14] to identify the chords played by the performer and displays them on the screen synchronously with the video. This algorithm is a simplified version of the state-of-the-art chord extraction model [15] whose accuracy outperforms that obtained by typical hand annotated guitar tabs from the web [13]: the average chord accuracy (79%) obtained by the automatic method over 180 Beatles tracks is 10 percentage points higher than the chord accuracy (69%) obtained from guitar tabs.

The algorithm in [14] is implemented in the Vamp plugin[15] Chordino/NNLS Chroma[16]. A spectrally whitened log-frequency spectrum (constant-Q with three bins per semitone) is first computed. It is automatically corrected for any deviations from the standard 440 Hz tuning pitch, and an approximate semitone spaced transcription is obtained using a dictionary of notes with geometrically decaying harmonics magnitudes. The resulting semitone spectrum is multiplied with a chroma profile, and mapped to 12 bins corresponding to pitch classes. Using these features, the algorithm provides chord transcription, using a set of profiles (dictionary) to calculate frame-wise chord similarities. The resulting chord sequence is smoothed by the standard hidden Markov model (HMM)/Viterbi approach. The chord dictionary comprises the four main chord classes: major, minor, diminished, and dominant.

4. GUITAR TAB PROCESSING

One of the driving factors behind the growth in online hand annotated tabs is in the ease in which they can be produced and shared by anyone. As a consequence, these tabs do not conform to any standard format and exist in many locations on the internet. As such, we have developed methods for mining the web for these tabs and parsing them to interpret the required data.

4.1 Tab mining

The web crawler of the Hotttabs application uses 911tabs[17], a guitar tablature search engine, to access 4.7 million tabs that have already been categorised by artist, title, tablature type, and rating. Additionally, we crawled 264 common chords from Chordie[18] and Guitarsite[19] to assist in the recognition of chords when parsing tab content.

4.2 Tab parsing

To interpret the tablature text from the HTML code and the chord sequence from the tablature text, Hotttabs does the following:

- Any HTML code is stripped from the tab and "nonbraking space" or "new line" tags are expanded accordingly.
- Chord definitions indicating fingerings are interpreted and added to a chord dictionary for the remainder of the tablature (e.g. "C#m: 45664-"; in this sequence, the numbers indicate the finger positions along the guitar neck for the six guitar strings ordered by decreasing pitch from left to right, and the hyphen indicates that the string must not be plucked).
- The tablature is divided up into sections based on the layout and any structural indicators (e.g. "Chorus").

[12] last.fm not only tracks listeners' musical history on their website but also when they use other services in desktop and mobile applications through what they call "scrobblers": http://www.last.fm/help/faq?category=Scrobbling

[13] http://code.google.com/p/gdata-python-client/

[14] http://ffmpeg.org/

[15] http://www.vamp-plugins.org/

[16] http://isophonics.net/nnls-chroma

[17] http://www.911tabs.com

[18] http://chordie.com

[19] http://www.guitarsite.com/chords

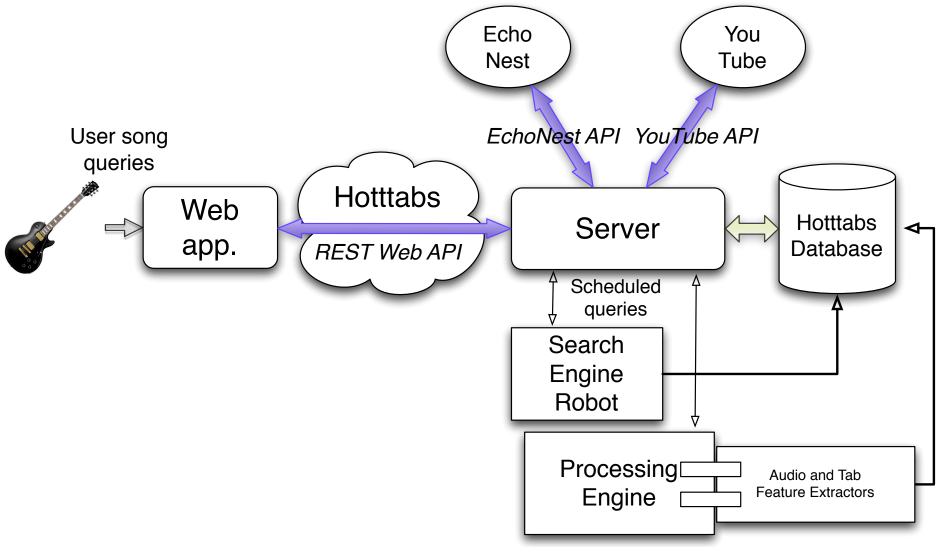


Figure 3: Hotttabs web application flow chart (app.: application; API: Application Programming Interface).

```
[Intro]

Riff1
e---------0--|--3-----3-----5-----5-----|--10----------------------------|--------8---------
B-----3---1--|--3-----3-----7-----7-----|--12-------12--0----0---12---0---|--------10--10--7-
G------------|--4-----4-----7-----7-----|--12-------12--12---12--12---12--|--9-----9---------
D------------|--------------------------|---------------12---12--12---12--|--10--------------
A------------|--------------------------|---------------------------------|-(10)-------------
E------------|--------------------------|---------------------------------|------------------

Riff2
e--3----3----2----2----|--0----0-------(0)-|
B--3----3----3----3----|--3----3---(3)-----|
G--0----0----2----2----|--0----0-----------|
D--0----0----0----0----|--2----2----0---2--|
A--2----2----x----x----|--2----2-----------|
E--3----3----2----2----|--0----0-----------|
   G         D/F#         Em

G           D/F#          Em
Love        love          love
G           D/F#          Em
Love        love          love
D7/A        G             D7/F#     D7/E
Love        love          love
D   C   Riff3
```

Figure 2: Tab Sample. Chords Extracted: G D/F# Em G D/F# Em G D/F# Em D7/A G D7/F# D7/E D C

- Each line is scanned for any occurrence of a chord label.
- For each tab line, the tab elements are decoded accordingly.
- Any indicators of repetitions will be expanded so that "x2" will result in the current section of chords being duplicated.

An example of a tab and the chord sequence extracted can be seen in Figure 2.

4.3 Tab clustering

When learning how to play guitar, one of the difficulties lies in knowing an increasing number of chords and their relative fingerings. Thus the chord vocabulary (i.e. the set of unique chords) used in a guitar tab is of interest to the learning guitarist. Additionally, both the number of chords required to play the song and the specific chords it contains (as some chords tend to be easier to play than others) influence the guitarist when choosing a guitar tab. For any given song it is common to find several guitar tabs with chord vocabularies of varying sizes. Indeed, some simplified (e.g. to make it more accessible and easier to play) or even complexified versions (e.g. to change the style or genre of the song by varying the harmonisation) of the original song are sometimes provided on guitar tabs websites.

Thus to help the user choose between all the tabs retrieved for one seed song we further cluster the guitar tabs into three categories based on the size of their chord vocabulary: easy, medium, and difficult. To do so we rank the tabs by the number of unique chords they each contain and then divide this ranked list into three clusters. The tab clusters are then displayed as tag clouds (aggregated collections of words or phrases used to describe a document), where each tag in the cloud shows the name of the website from which the tab was extracted as well as the chord vocabulary used in the tab (see bottom of Figure 4). Therefore users can know, in one glance, which chords are required to play a given tab and how many chords it contains, without having to browse each tab individually. By clicking on an item in the tab clouds the user is redirected to the full tab in the website where it was originally published. Although the difficulty to play individual chords is not yet taken into account in the tab clustering process (which only uses the size of the chord vocabulary of the tab), displaying the chord vocabulary in the tab cloud helps users to choose the most appropriate tabs for them since they know which chords they have already learned and which chords they find difficult to play. However as most guitarists consider some specific chords to be more difficult or more tiring for the hand than others due to their fingering constraints (e.g. barre chords), we will consider including a measure of chord (fingering) difficulty into future implementations of our tab clustering algorithm.

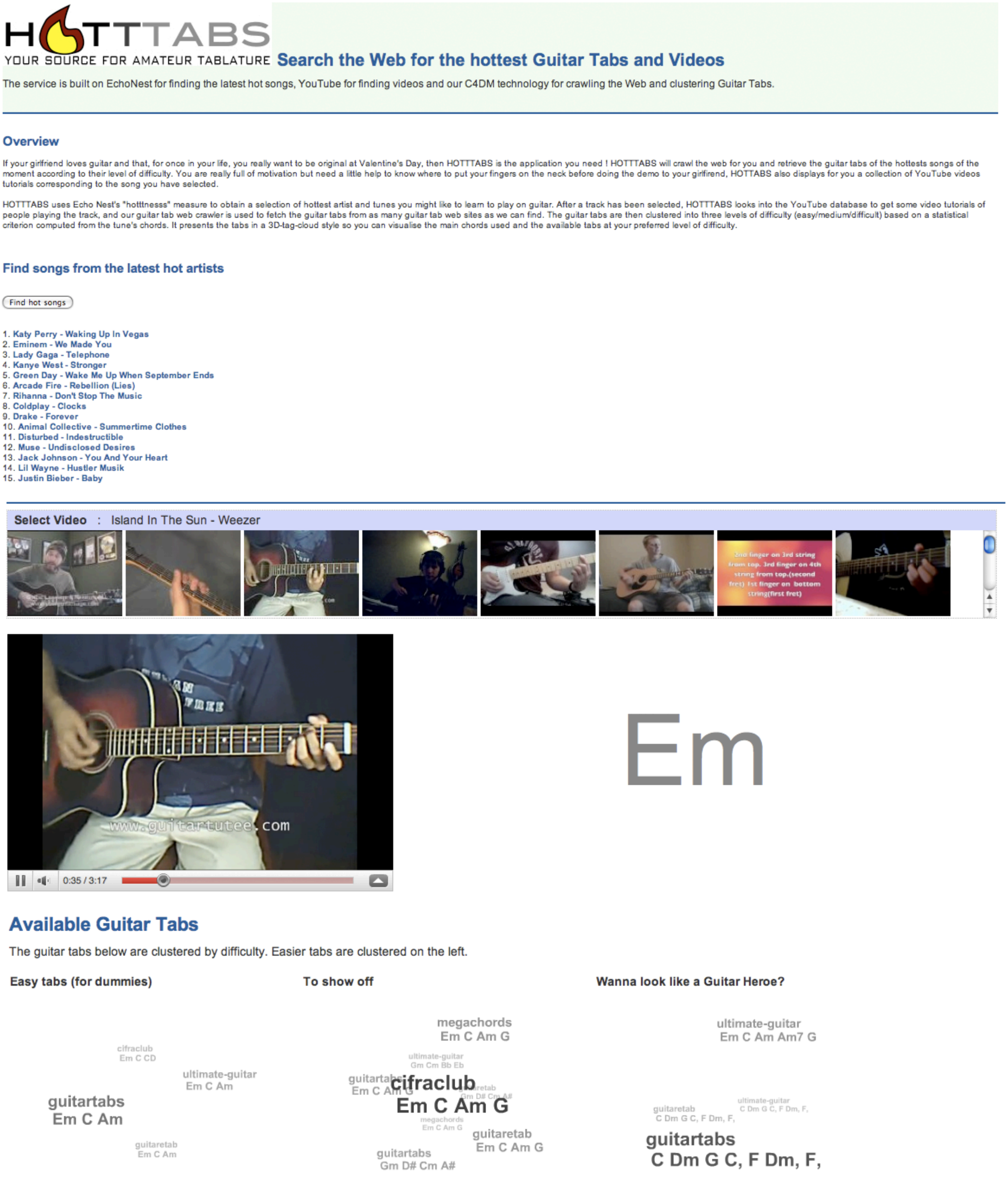


Figure 4: Screenshot of the Hotttabs web application (http://isophonics.net/hotttabs/).

5. HOTTTABS WEB APPLICATION

The Hotttabs application integrates the functionality described in the previous sections in a web-based client-server architecture.

The client runs in most popular web browsers, and provides an easy to use interface (see Figure 4). It allows the user to interact with the application and perform the following actions: *i)* query for popular songs, *ii)* retrieve a list of video tutorials and three sets of tab clusters (easy, medium, and difficult) for the selected popular song, *iii)* play a video, from a list of thumbnails, in an embedded video player, synchronised with automatically extracted chords, *iv)* select and link to a tab from the tab clusters as you would from a search engine.

In response to user interaction, the server performs the core functionality as described in section 5.2. Concerning client-server communication, Hotttabs follows the Representational State Transfer (REST) style web application design (see Figure 3). In this architecture web pages form a virtual state machine, allowing a user to progress through the application by selecting links, with each action resulting in a transition to the next state of the application by transferring a representation of that state to the user [9].

5.1 Front end

The light weight client uses a combination of standard web technologies (HTML, CSS, JavaScript) and makes use of the JQuery[20] library to dynamically load content from the server via XMLHttp requests. This content includes the list of popular songs, and the a list of video thumbnails for a selected song. We developed client-side JavaScript code which interacts with the embedded YouTube player, to display chord names next to the video. The chord sequence is requested when the user starts the video, and returned using JavaScript Object Notation with timing information, which is used to synchronise the chords with the video. The tab clusters are displayed using an adapted version of the WP-Cumulus Flash-based tag cloud plugin[21]. This plugin utilises XML data generated on the server side from the results of the tab search and tab clustering algorithm.

5.2 Back end

The server side of the Hotttabs application builds on semantic audio and web technologies outlined in [8]. The Sonic Annotator Web Application (SAWA) [7], a Python[22] framework for writing web applications involving audio analysis, is used as a basis for Hotttabs. This is extended with modules to access The Echo Nest, YouTube, and perform additional application specific functionality as shown in Figure 3.

The communication between the client and server is coordinated using the Model View Controller (MVC) architectural pattern [6]. Some important domain objects in the MVC model, as well as the Hotttabs database, are provided by the Music Ontology framework [17], such that corresponding data structures are generated from the ontology specification using the Music Ontology Python library [7]. For instance, information about popular artists and their songs (retrieved from The Echo Nest) are stored in objects and database entries corresponding to the `mo:Track`[23] and `mo:MusicArtist` concepts.

Besides user interaction, the server also performs scheduled queries for popular songs to bootstrap the database. This is necessary, since crawling for guitar tabs and the feature extraction process for chord analysis are too computationally expensive to be performed in real-time. This process uses the crawler described in section 4.1, as well as the chord extraction algorithm of [14] implemented as a Vamp audio analysis plugin [3] which can be loaded by the processing engine of SAWA.

6. CONCLUSIONS AND PERSPECTIVES

We presented Hotttabs, an online multimedia guitar tuition service comprised of the following features: (i) the recommendation of popular songs based on The Echo Nest "hotttness" measure, taking into account the artists' popularity dynamically through web data mining, (ii) the retrieval of guitar video tutorials from the YouTube database, (iii) the visual feedback of the chord labels using a content-based music information retrieval technique, and (iv) the recommendation of guitar tablatures targeting users of different levels depending on the vocabulary of chords in the selected song.

We plan to conduct a user survey in order to obtain some feedback to feed into future technical developments of the application. We also intend to model user skills and assess performances in order to adapt which music and guitar tabs are recommended, based on the users learning process. Interesting follow-ups to this work also include the development of a guitar chord fingering dictionary to display various possible chord fingerings along with the chord labels. The chord concurrence measure introduced in [13] could be used to select the most accurate guitar tabs and discard erroneous ones. Future work will also address the development of new tab clustering methods based on the chord sequence parsing, the integration of an audio/video time-stretching technique to allow for the slowing down of the video tutorials, and the synchronisation of guitar tabs and lyrics with the videos using sequence alignment.

7. ACKNOWLEDGMENTS

The authors would like to thank Matthias Mauch for his automatic chord recognition system. This work was partly funded by the musicology for the masses EPSRC project EP/I001832/1, and the Platform Grant from the Centre for Digital Music funded by the EPSRC project EP/E045235/1.

8. REFERENCES

[1] P. Aman and L. A. Liikkanen. A survey of music recommendation aids. In *Proceedings of the 1st Workshop on Music Recommendation and Discovery (WOMRAD) colocated with ACM RecSys*, pages 25–28, Barcelona, Spain, 2010.

[20] http://jquery.com/

[21] http://wordpress.org/extend/plugins/wp-cumulus/

[22] http://www.python.org

[23] In this notation, the namespace prefix **mo:** represents the URL of the Music Ontology.

[2] M. Brandao, G. Wiggins, and H. Pain. Computers in music education. In *Proceedings of the AISB'99 Symposium on Musical Creativity*, Edinburgh, UK, 1999.

[3] C. Cannam. The VAMP audio analysis plugin API: A programmer's guide. Availble online: `http://vamp-plugins.org/guide.pdf`, 2009.

[4] O. Celma. *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.

[5] Y.-K. Cliff Liao. Effects of computer-assisted instruction on students' achievement in Taiwan: A meta-analysis. *Computers & Education*, 48:216–233, 2007.

[6] E. Evans. *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley Professional, 2003.

[7] G. Fazekas, C. Cannam, and M. Sandler. Reusable metadata and software components for automatic audio analysis. *IEEE/ACM Joint Conference on Digital Libraries (JCDL'09) Workshop on Integrating Digital Library Content with Computational Tools and Services, Austin, Texas, USA*, 2009.

[8] G. Fazekas, Y. Raimond, K. Jakobson, and M. Sandler. An overview of Semantic Web activities in the OMRAS2 project. *Journal of New Music Research (special issue on Music Informatics and the OMRAS2 project)*, 39(4):295–311, 2010.

[9] R. T. Fielding and R. N. Taylor. Principled design of the modern Web architecture. *ACM Transactions on Internet Technology*, 2(2):115–150, 2002.

[10] IRMA. Le marché des instruments de musique: une facture bien réglée [the market of musical instruments: a well tuned invoice]. `http://www.irma.asso.fr/LE-MARCHE-DES-INSTRUMENTS-DE?xtor=EPR-75`, Last viewed in September 2011.

[11] A. LeBlanc, Y. C. Jin, M. Obert, and C. Siivola. Effect of audience on music performance anxiety. *Journal of Research in Music Education*, 45(3):480–496, 1997.

[12] S.-J. Lou, Y.-C. Guo, Y.-Z. Zhu, R.-C. Shih, and W.-Y. Dzan. Applying computer-assisted musical instruction to music appreciation course: An example with Chinese musical instruments. *The Turkish Online Journal of Educational Technology (TOJET)*, 10(1), 2011.

[13] R. Macrae and S. Dixon. Guitar tab mining, analysis and ranking. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, Miami, Florida, USA, 2011.

[14] M. Mauch and S. Dixon. Approximate note transcription for the improved identification of difficult chords. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, Utrecht, Netherlands, 2010.

[15] M. Mauch and S. Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1280–1289, 2010.

[16] A. Naofumi, A. Yoshiano, and Y. Tsuyoshi. A study of automated tablature generation for assisting learning playing the guitar. Technical Report 189, IEIC Technical Report (Institute of Electronics, Information and Communication Engineers), Japan, 2000.

[17] Y. Raimond, S. Abdallah, M. Sandler, and G. Frederick. The music ontology. In *Proceedings of the 7th International Society for Music Information Retrieval Conference (ISMIR 2007)*, Vienna, Austria, 2007.

Using Semantic Relations in Context-based Music Recommendations

İpek Tatlı, Ayşenur Birtürk
Department of Computer Engineering, METU
Inonu Bulvari, 06531
Ankara, Turkey
{ipek.tatli, birturk}@ceng.metu.edu.tr

ABSTRACT

In this paper, we describe an approach for creating music recommendations based on user-supplied tags that are augmented with a hierarchical structure extracted for top level genres from Dbpedia. In this structure, each genre is represented by its stylistic origins, typical instruments, derivative forms, sub genres and fusion genres. We use this well-organized structure in dimensionality reduction in user and item profiling. We compare two recommenders; one using our method and the other using Latent Semantic Analysis (LSA) in dimensionality reduction. The recommender using our approach outperforms the other. In addition to different dimensionality reduction methods, we evaluate the recommenders with different user profiling methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*retrieval models, information filtering, selection process*

General Terms

Algorithms, Experimentation, Human Factors

Keywords

Recommendation systems, user profiling, social tagging, semantic relations, dimensionality reduction

1. INTRODUCTION

These days, most social-networking sites let their members participate in content generation. For example, users can label artists, albums and tracks with tags in Last.fm. A tag can be anything but it is actually a short description of the item. Because tags represent the reason why a listener likes an item, but not how much he/she likes it they are better identifiers of user profiles than ratings, which are usually numerical values assigned to items by users. Thus, we concentrate on the tag-based contextual representations of music tracks.

Items are mostly represented in vector spaces in the recommendation systems. In tag-based recommendation systems, users and items are defined in terms of weighted vectors of social tags. When there is a large amount of tags, calculation of the items to be recommended becomes hard, because working with huge is to represent individual tracks (songs) in lower dimensional spaces. In order to reduce the dimensionality, we focus on the genre information of the tags. Each genre has a relationship with some instrumentation, with some subgenre information and with style information each of which may be entered as tags in the music domain. In our work, for each genre Dbpedia[1] (a structured form of Wikipedia[2]) is crawled to set the relationships between genre and its stylistic origins, typical instruments, derivative forms, sub genres and fusion genres. The contributions of our approach are that: (1) we provide a "semantic relations" method for dimensionality reduction in very huge vector spaces and (2) we perform the comparison of our method against the classical Singular Value Decomposition (SVD) method which is the base of Latent Semantic Analysis (LSA). Our method outperforms the traditional one.

2. RELATED WORK

In music recommendation systems, tracks can be profiled in terms of their audio contents (like rhythm, timbre, tonality, instrumentation). In addition to audio descriptions, tracks can be profiled in terms of their text descriptions like their metadata, lyrics, tags and reviews mined from various blogs [1]. Metadata information is mostly supplied by experts. The artist's name, the album's name, genre, duration and year are some attributes in the metadata. Attributes are global descriptions of items and do not change according to users whereas tags are local descriptors and might change from user to user [2]. In our study, we focus on text descriptions, namely tags in track profiling.

Recommender systems either predict ratings for unseen items or predict items that can be liked. Most of the social web-sites like Last.fm do not have a rating mechanism. Instead of explicit ratings, today's recommender systems use implicit ratings (users' listening habits and purchase histories etc.). Thus the rating scale in implicit rating mechanisms is 0-1. Tags can be used in rating-based collaborative

[1]http://dbpedia.org
[2]http://en.wikipedia.org

WOMRAD 2011 2nd Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2011 (Chicago, US)

filtering systems with the help of an implicit rating mechanism [2]. If the tag is used by the user, its rating is 1; otherwise its rating is 0. In most previous studies, 2-dimensional spaces in music space are taken into consideration (item-user or user-tag or item-tag). User-tag and item-tag relations can be used to extend the rating data [2]. A new approach which uses all dimensionalities of the music space is proposed in [3]. Each 'useritem- tag' data is a tensor in this study and the researchers propose a Higher Order Singular Value Decomposition (HOSVD) technique for 3-dimensional social tagging data. The HOSVD method outperforms the classical methods. In contrast, the three 2-dimensional relations among users, tags and items have been used in a new similarity measure generation which outperforms the traditional item-based and user-based collaborative filtering methods [4]. In this approach, neighborhood generation is effected through the similarity of users' tags, similarity of users' items and the similarity of user's tag-item relationships. In addition to user similarities item similarities have been calculated with common tags, common users and common tag-item relationships. Moreover, tags can be clustered and these clusters improve the personalized recommendations [5].

Up to our knowledge, [6] uses a similar approach to ours in extracting top 50 music facets from Wikipedia but the main objective of [6] is to provide an automatic method for uncovering the music facets and to classify tags according to these facets. On the other hand, we create a hierarchical genre structure and evaluate the usefulness of our approach in music recommendation.

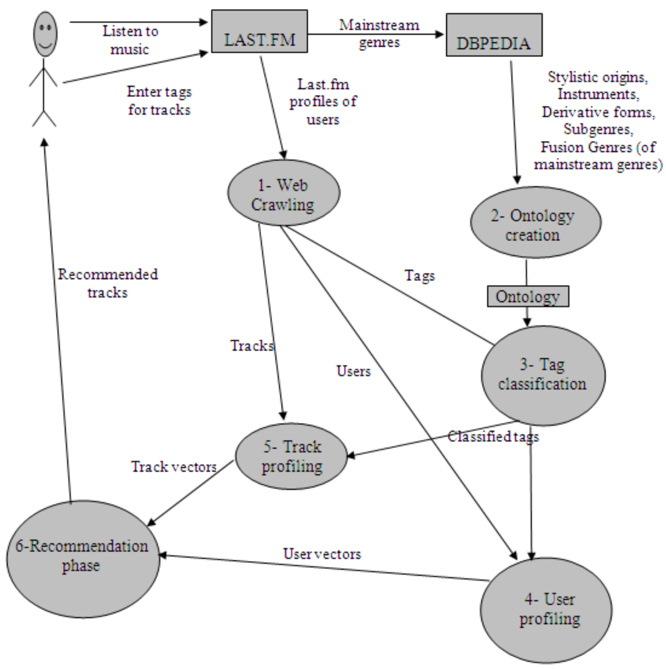


Figure 1: System architecture of the proposed approach

3. OUR APPROACH

Our system performs 6 main tasks, shown in Figure 1: web crawling, creating an ontology of musical genres, classifying tags according to the ontology, track profiling, user profiling and enacting the recommendation process. The circles denote the phases of the system.

Users listen to music and enter tags for tracks in their Last.fm profiles. In the **web crawling phase** of the system, a data set is generated. Details of the data set are given in Section 4.

Tags may be about genre, instrumentation, location, moods, style, personal opinions and/or artists' names [1]. For example, two users of Last.fm tagged some songs as follows: the first one loved listening to "The Wall" from "Pink Floyd" and tagged the track with the words "energetic" and "seen live". The second one loved "Only Girl" from "Rihanna" and tagged "Only Girl" also with the words "energetic" and "seen live". Thus both "Only Girl" and "The Wall" have the same tags. According to the recommendation's similarity function, they appear as very similar tracks, although in most other ways (genre, instrumentation for instance) they are not. Because of such reasons, subjective tags like personal opinions and moods are ignored in the track and user representation in our system. Instrumentation, subgenre, fusion genre, derivative forms and stylistic information are used in our track and user representation. Firstly, we decided the main genres in the musical domain. In [8], 14 mainstream genres (country, folk, jazz, blues, r&b, heavy metal, alternative/indie, punk, rap, electro, reggae, classical, rock and pop) are used. We enriched these genres with Last.fm's mainstream genres, which can be reached on the left frame of the page http://www.last.fm/music. The main genres used in our system are as follows: acoustic, ambient, blues, classical, country, electronic, emo, folk, hardcore, hip hop, indie, jazz, Latin, metal, pop, pop punk, punk, reggae, r&b, rock, soul, world. Having identified our genres, we decided to crawl the Wikipedia page for each main genre but then we switched to Dbpedia since it is more structured for web-crawling. We crawled Dbpedia page for each main genre. Obtained information is illustrated in Table 1 for "rock music".In the **ontology creation phase**, we created a small ontology-a hierarchical structure -with the help of the data crawled from the Dbpedia. Relations in our ontology can be seen in Table 2. In this structure instrumentation, stylistic origins, derivative forms, subgenres and fusion genres are the classes; the crawled data are the instances. For example, "New Age Music" and "Synthpop" are instances of the class "Derivative forms" which can be seen in Table 1.

LSA does not use word order and morphology. In order not to differentiate "electronic" from "electronica", we applied some stemming algorithm. Stemming is a technique to convert similar words to a common base form. This base form does not have to have a meaning from a linguistic point of view (such as reducing synonyms to a single word, or finding the root of the word). Various stemming algorithms exist for the English language. We used the Porter stemmer[3] which is a classical stemming algorithm. By using a stemming algorithm, morphology is taken into consideration in our approach. In the **tag classification phase** of our system, we parsed instances existing in the ontology into single words. We applied the stemming algorithm to each single word. Then we concatenated the stemmed roots with "%" in order to consider "word order" that LSA does not use. Some

[3]http://tartarus.org/ martin/PorterStemmer

Table 1: Wikipedia/Dbpedia page of "rock music"

Rock music
Stylistic origins: Rock and roll, electric blues, folk music, country, blues
Typical instruments: Vocals, electric guitar, bass guitar, drums, synthesizer, keyboards
Derivative forms: New Age Music - Synthpop
Subgenres: Alternative rock - Art rock - Beat music - Britpop - Emo - Experimental rock - Garage rock - Glam rock - Grindcore - Group Sounds - Grunge - Hard rock - Heartland rock - Heavy metal - Instrumental rock - Indie rock - Jangle pop - Krautrock - Madchester - Post-Britpop - Power pop - Progressive rock - Protopunk - Psychedelia - Punk rock - Rock noir - Soft rock - Southern rock - Surf - Symphonic rock (complete list)
Fusion genres: Aboriginal rock - Afro-rock - Anatolian rock - Bhangra rock - Blues-rock - Countryf rock - Flamencorock - Folk rock - Funk rock - Glam Punk - Indo-rock - Industrial rock - Jazz fusion - Pop rock - Punta rock - Raga rock - Rai rock - Rap rock - Rockabilly - Rockoson - Samba-rock - Space rock - Stoner rock - Sufi rock

Table 2: Relations in our ontology

hasStylisticOrigins	Genre&Stylistic Origins
hasInstruments	Genre&TypicalInstrumentation
hasDerivativeForms	Genre&Derivative Forms
hasSubGenres	Genre&Sub Genres
hasFusionGenres	Genre&Fusion Genres

examples of the stemming results can be seen in Table 3.

All the tags in our dataset are saved in the "tags" table. The reason why we use "%" in the new version of instances is that we use these versions of the instances in our SQL statements. We use the newer instances in SQL statements like "select * from tags where tag_name like '%Aborigin%rock%' ". With this usage, we are using about 100000 tags out of 160000 tags in the track representation. In the **track profiling phase**, the size of a track vector is the size of mainstream genres (22 in our case). Last.fm provides integer percentages (between 0 and 100) relative to the most used tags per track. We updated these percentages by adding 1 to each percentage value in order not to discard any having 0 percentage. Each entry in the vector is calculated as follows:

$$Term-Count(g(i,j)) = \sum_k hasInstrumentation(i,j) + \sum_k hasStylisticOrigins(i,j) + \sum_k hasDerivativeForms(i,j) + \sum_k hasSubGenres(i,j) + \sum_k hasFusionGenres(i,j)$$

Table 3: Concatenating the stemmed words of the instances

Tag before stemming	Tag after stemming
electric blues	%eletr%blu%
Aboriginal rock	%Aborigin%rock%

Where, $g(i,j)$ is the i^{th} term (genre) in j^{th} track; and $hasInstrumentation(i,j)$ is the total percentage (between 1 and 101) of the tags of the j^{th} track which are found to be similar to the new instance versions of the instrumentation class of i^{th} genre (with the help of the aforementioned SQL statements). The term count is usually normalized to prevent a bias towards longer documents (which may have a higher term count regardless of the actual importance of that term in the document). The term frequency (TF) value gives local information about a tag. An inverse document frequency (IDF) value is calculated for each different tag in the training set. This is calculated by dividing the total number of tracks by the number of tracks that refer to that feature. The IDF value gives global information of a tag. Thus, tracks in our dataset are represented as a weighted list of genres and the weights of the genres are calculated with TF*IDF.

$$w_i = \frac{n_{i,t}}{n_t} \times log(\frac{|T|}{|T_i|})$$

In the formula above, w_i is the weight of i^{th} genre; $n_{i,t}$ equals the number of times i^{th} genre appears in t^{th} track; n_t is the total number of genres in t^{th} track; $|T|$ is the total size of the tracks, and $|T_i|$ equals the number of tracks in which i^th genre appears. In the **user profiling phase**, 3 different methods are used:

1- using the users' own tags (personal tags) that they entered

2- using the users' friends' tags (friends' tags) that their friends entered

3- using all the tags of the tracks (social tags) that they listened to

In the first method, users are profiled with their own tags. In the second method, users are profiled with their friends' tags. And in the last method, users are profiled with all the tags of the tracks that they listened to. Semantic relations are also used in user profiling method 1 and method 2, just the same as in track profiling. In method 3, a user profile is the sum of the tracks that he/she has listened to. Weights of the genres in user vectors are also calculated with TF*IDF method. The main goal after creating a user profile from the training set is to recommend the items in the test set.

In the **recommendation phase**, we use the common cosine similarity method. The cosine similarity formula is given below:

$$CosSim(track, user) = \frac{track\_vector \times user\_vector}{|track\_vector||user\_vector|}$$

4. DATA SET

We use real Last.fm data in this study. In order to not to use similar users from our own friend lists and in order to achieve diversity, we selected 69 users from an application named "join Last.fm"[4]. In this group, members of the group share their Last.fm nicknames. We crawled their Last.fm profiles with the help of Last.fm API[5]. Since our approach is not collaborative but content-based, this number of users is reasonable. Firstly we gathered their top 300 tracks. Then we extracted their "loved" tracks. For each track, we extracted the singer names and tags. We also gathered the

[4]http://www.facebook.com/group.php?gid=2246697136&v=wall
[5]http://last.fm/api

Table 4: Details of the data set

# of users	69
# of tracks	13312
# of tags	169174
# of singers	4253
Average # of tracks per user	527
Average # of tags per track	45
Average # of tags per user	85
Average # of friend tags per user	451

tag counts per track. Finally for each user we extracted their tags and their friends' tags. Details of our data set can be seen in Table 4.

5. EXPERIMENTS

5.1 Methodology

We performed a 4-fold-cross validation in which the training data size was 75% and the test data was 25%. User profiles were created using the training set and the task of our recommender system was to predict the correct items in the test set.

5.2 Metrics

In this study we used the most common evaluation metric: Precision at the top N ranked results (P@N). Precision is the ratio of relevant tracks selected correctly to the number of tracks selected.

5.3 Results

In Last.fm, although users listen to music, they rarely enter tags for the tracks that they like. Thus, user profiles in Last.fm are smaller than in other social tagging sites, so that the performance of the pure content-based recommendation is not satisfying [7]. In Table 5; two recommenders using LSA with an optimized parameter -k- and our method in dimensionality reduction are compared in terms of recommending the corresponding tracks in the test set. LSA is applied to the track-tag matrix whose size is 13312*169174 (13312 tracks, 169174 tags). On the other hand, the recommender using semantic relations method decreases the matrix size to 13312 * 22 (13312 tracks, 22 genres). In this recommender, each genre is semantically related to instruments, stylistic origins, subgenres, fusion genres and derivative forms. Thus, semantically related tags are counted as the same genre in this representation. As seen in the Table 5, it is obvious that the recommender using semantic relations outperforms the recommender using LSA in dimensionality reduction because it handles the semantic gap problem in social tagging. Moreover, the recommender using users' own tags in user profiling performs better than the recommender using friends' own tags. However, the recommender using all social tags in the user profiling seems to provide the best results because of the increasing number of tags in user vectors.

6. CONCLUSION

User annotated texts, tags in our case, are huge in size, but the representation matrix is very sparse. Using such giant matrices in calculations is a time- and resource- consuming job. For the document categorization or text summarization, LSA has been used for years because it is easy to use and reliable. As an alternative, with the help of Dbpedia, we created an ontology-like semantic relations structure for the music domain. In this paper, we evaluated two methods which can be used in dimensionality reduction. In the evaluation Last.fm dataset was used and the recommenders were evaluated with different user profiling methods. Our method has the advantage of using "word order" and "morphology" with respect to LSA. We plan to extend our work, assigning different weights for different relations. For instance, *hasInstrumentation* and *hasSubgenres* may have different weights in the track profiling.

Table 5: Details of the data set

Dim. reduction method	User profiling method	P@5	P@10	P@20
Semantic relations	Tags of tracks user listened to	0.178	0.168	0.134
Semantic relations	Tags user entered	0.000	0.100	0.175
Semantic relations	Tags friends entered	0.000	0.000	0.000
LSA (with optimal k)	Tags of tracks user listened to	0.079	0.077	0.071
LSA (with optimal k)	Tags user entered	0.000	0.065	0.081
LSA (with optimal k)	Tags friends entered	0.000	0.000	0.016

7. REFERENCES

[1] O. Celma and P. Lamere. Music Recommendation Tutorial.*ISMIR.*Vienna, Austria, 2007.

[2] K.H.L. Tso-Sutter, L.B. Marinho and L. Schmidt-Thieme.Tag-aware recommender systems by fusion of collaborative filtering algorithms. *ACM symposium on Applied Computing.* Brazil, 2008.

[3] P. Symeonidis, M. Ruxanda, A. Nanopoulos and Y. Manolopoulos. Ternary semantic analysis of social tags for personalized music recommendation. *ISMIR.* Philadelphia, USA, 2008.

[4] H. Liang, Y. Xu, Y. Li and R. Nayak. Collaborative Filtering Recommender Systems Using Tag Information. *Web Intelligence and Intelligent Agent Technology.* Sydney, Australia, 2008.

[5] A. Shepitsen, J. Gemmell, B. Mobasher and R. Burke. Personalized recommendation in social tagging systems using hierarchical clustering. *ACM conference on Recommender systems.* Lausanne, Switzerland, 2008.

[6] M. Sordo, F. Gouyon and L. Sarmento. A Method for Obtaining Semantic Facets of Music Tags. *WOMRAD.* Barcelona, Spain, 2010.

[7] I. Cantador, A. Bellogin and D. Vallet. Content-based Recommendation Systems in Social Tagging Systems. *ACM conference on Recommender systems.* Barcelona, Spain, 2010.

[8] M. Levy and M. Sandler. Learning Latent Semantic Models For Music From Social Tags. *Journal of New Music Research.* pages 137-150.

Towards Semantic Music Information Extraction from the Web Using Rule Patterns and Supervised Learning

Peter Knees and Markus Schedl
Department of Computational Perception, Johannes Kepler University, Linz, Austria
peter.knees@jku.at, markus.schedl@jku.at

ABSTRACT

We present first steps towards automatic Music Information Extraction, i.e., methods to automatically extract semantic information and relations about musical entities from arbitrary textual sources. The corresponding approaches allow us to derive structured meta-data from unstructured or semi-structured sources and can be used to build advanced recommendation systems and browsing interfaces. In this paper, several approaches to identify and extract two specific semantic relations from related Web documents are presented and evaluated. The addressed relations are members of a music band ($band-members$) and artists' discographies ($artist - albums, EPs, singles$). In addition, the proposed methods are shown to be useful to relate (Web-)documents to musical artists. For all purposes, supervised learning approaches and rule-based methods are systematically evaluated on two different sets of Web documents.

Categories and Subject Descriptors

J.5 [**Arts and Humanities**]: *Music*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*

General Terms

Algorithms

Keywords

Music Information Extraction, Band-Member Relationship, Discography Extraction

1. MOTIVATION AND INTRODUCTION

Measuring similarity between artist, tracks or other musical entities — be it audio-based, Web-based, or a combination of both — is a key concept for music retrieval and recommendation. However, the type of relations between these entities, i.e., *what* makes them similar, is often neglected. Especially in the music domain, the number of potential relations between two entities is large. Such relations comprise, e.g., cover versions of songs, live versions, re-recordings, remixes, or mash-ups. Semantic high-level concepts such as "*song X* was inspired by *artist A*" or "*band B* is the new band of *artist A*" are very prominent in many users' conception and perception of music and should therefore be given attention in similarity estimation approaches. By focusing solely on acoustic properties, such relations are hard to detect (as can be seen, e.g., from research on cover version detection [7]).

A promising approach to deal with the limitations of signal-based methods is to exploit *contextual* information (for an overview see, e.g., [16]). Recent work in music information retrieval has shown that at least some cultural aspects can be modeled by analyzing extra-musical sources (often referred to as *community metadata* [25]). In the majority of work, this data — typically originating from Web sources and user data — is used for description/tagging of music (e.g., [10, 23, 24]) and assessment of similarity between artists (e.g., [17, 21, 22, 25]). However, while for these tasks standard information retrieval (IR) methods that reduce the obtained information to simple representations such as the bag-of-words model may suffice, important information on entities like artists' full names, band member names, album and track titles, related artists, as well as some music specific concepts like instrument names and musical styles may be dismissed. Addressing this issue, essential progress towards identifying relevant entities and, in particular, relations between these could be made. These kinds of information would also be highly valuable to automatically populate music-specific ontologies, such as the Music Ontology[1] [15].

In this paper, we aim at developing automatic methods to discover semantic relations between musical entities by analyzing texts from the Web. More precisely, to assess the feasibility of this goal, we focus on two specific sub-tasks, namely *automatic band member detection*, i.e., determining which persons a band consists (or consisted) of, and *automatic discography extraction*, i.e., recognition of released records (i.e., albums, EPs, and singles). Band member detection is strongly related to one of the central tasks of information extraction (IE) and named entity detection (NED), i.e., the recognition of persons' names in documents. While person's names typically exhibit some common patterns in terms of orthography and number of tokens, detection of artist names and band members is a bigger challenge as they frequently comprise or consist of nicknames, pseudonyms, or just a symbol (cf. *Prince* for a limited time). Discog-

[1] http://www.musicontology.com

WOMRAD 2011 2nd Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2011 (Chicago, US)

raphy detection in unstructured text is an even more challenging task as song or album names (release names in the following) are not bound to any conventions. That is, release names can consist of an unknown number of tokens (including zero tokens, cf. *The Beatles*'s "white album", or *Weezer*'s "blue", "green", and "red" albums, which might lead to inconsistent references on different sources), just special characters (e.g., *Justice*'s "Cross"), a differential equation (track 2 on *Aphex Twin*'s "Windowlicker" single), or whole paragraphs (e.g., the full title of a *Soulwax* album often abbreviated as *Most of the remixes* consists of 552 characters). Especially the last example demonstrates some of the challenges of a discography-targeted named entity recognition approach as the full album title itself exhibits linguistic structures and even contains another band's name (*Einstürzende Neubauten*). Hence, general methods not tailored to (or even aware of) music-related entities might not be able to deal with such specifics.

To investigate the potential and suitability of language-processing-based approaches for semantic music information extraction from (Web-)texts, two strategies commonly used in IE tasks are explored in this paper: manual tailoring of rule patterns to extract entities of interest (the "knowledge engineer" approach) and automatic learning of patterns from labeled data (supervised learning). Since particularly for the latter, pre-labeled data is required — which is difficult to obtain for most types of semantic relations — band-membership and discography extraction are, from our point of view, good starting points as these types of information are also largely available in a structured format (e.g., via Web services such as MusicBrainz[2]). In addition, the methods presented are also applied to relate documents to musical artists, which is useful for further tasks such as automatic music-focused crawling and indexing of the Web. In the bigger picture, these are supposed to be but the first steps towards a collection of methods to identify high-level musical relations between pieces, like cover versions, variations, remasterings, live interpretations, medleys, remixes, samples, etc. As some of these concepts are (partly) deducible from the audio signal itself, well considered methods for combining information from the audio with (Web-based) meta-information are required to automatically discover such relations.

2. RELATED WORK

The two music information extraction tasks addressed in this paper, i.e., band member and discography extraction, are specific cases of relation extraction. Since in the scenarios considered in this paper, one of the relational concepts is considered to be known (i.e., the band a text deals with), semantic relation extraction is reduced to named entity recognition and extraction tasks (i.e., extraction of band members and released records). Named entity recognition itself is a well-researched topic (for an overview see, e.g., [4]) and comprises the identification of proper names in structured or unstructured text as well as the classification of these names by means of rule-based or supervised learning approaches. While rule-based methods rely on experts that uncover patterns for the specific task and domain, supervised learning approaches require large amounts of labeled training data (which could, for instance, also stem from an ontology (cf. [1]). For the music domain – despite the numerous contributions that exploit Web-based sources to describe music or to derive similarity (cf. Section 1) – the number of publications aiming at extracting factual meta-data for musical entities by applying language processing methods is rather small.

In [19], we propose a first step to automatically extract the line-up of a music band, i.e., not only the members of a band but also their corresponding instruments and roles. As data source up to 100 Web documents for each band B, obtained via Google queries such as *"B" music*, *"B" music members*, or *"B" lineup music*, are utilized. From the retrieved pages, n-grams (where $n = \{2, 3, 4\}$), whose tokens consist of capitalized, non-common speech words of length greater than one are extracted. For band member and role extraction, a Hearst pattern approach (cf. [9]) is applied to the extracted n-grams and their surrounding text. The seven patterns used are 1. M plays the I, 2. M who plays the I, 3. R M, 4. M is the R, 5. M, the R, 6. M (I), and 7. M (R), where M is the n-gram/potential band member, I an instrument, and R a role. For I and R, roles in a "standard rock band line-up", i.e., singer, guitarist, bassist, drummer, and keyboardist, as well as synonyms of these, are considered. After extraction, the document frequency of each rule is counted, i.e., on how many Web pages each of the above rules applies. Entities that occur on a percentage of band B's Web pages that is below a given threshold are discarded. The remaining member-role relations are predicted for B. In this paper, evaluation of the presented approaches is also carried out on the best-performing document set from [19] and compared against the Hearst pattern approach.

In [18], we investigate several approaches to determine the country of origin for a given artist, including an approach that performs keyword spotting for terms such as "born" or "founded" in the context of countries' names on Web pages. Another approach for country of origin determination is presented in [8]. Govaerts and Duval use selected Web sites and services, such as Freebase[3], Wikipedia[4], and Last.fm[5]. Govaerts and Duval propose three heuristics to determine the artist's country of origin using the occurrences of country names in biographies (highest overall occurrence, strongly favoring early occurrences, weakly favoring early occurrences). In [6], Geleijnse and Korst apply patterns like *G bands such as A, for example A_1 and A_2*, or *M mood by A* (where G represents a genre, A an artist name, and M a possible mood) to unveil genre-artist, artist-artist, and mood-artist relations, respectively.

While these music-specific information extraction methods mainly build upon few simple patterns or term frequency statistics, the work presented in this paper aims at incorporating more general methods that take advantage of linguistic features of the underlying texts and automatically learn models to derive musical entities annotated examples.

3. METHODOLOGY

The methods presented in this paper make use of the linguistic properties of texts related to music bands. To assess this information, for both approaches investigated (rule-based and supervised-learning-based), several pre-processing

[2]http://musicbrainz.org/

[3]http://www.freebase.com

[4]http://www.wikipedia.org

[5]http://last.fm

steps are required to obtain these linguistic features. Apart from initial preparation steps such as markup removal (if necessary), text tokenization (i.e., splitting the text into single tokens based on white spaces) and sentence splitting (based on punctuation), this comprises the following steps:

1. **Part-of-Speech Tagging (PoS)**: assigns PoS tags to tokens, i.e., annotates each token with its linguistic category (noun, verb, preposition, etc.), cf. [3].

2. **Gazetteer Annotation**: annotates occurrences of pre-defined keywords known to represent a specific concept, e.g., company names or persons' (first) names. These annotations can be used as look-up information for subsequent steps (see below). For the music domain, in this step, we also include lists of musical genres, instruments, and band roles, as well as a list of country names, cf. [11].

3. **Transducing Step**: identifies named entities such as persons, companies, locations, or dates using manually generated grammar rules. These rules can include lexical expressions, PoS information, look-up entities extracted via the gazetteer, or any other type of available annotation.

For all of these steps the functionalities included in the GATE software package (General Architecture for Text Engineering [5]) are utilized. In GATE's transducing step, detection of the different kinds of named entities is performed simultaneously in an interwoven process, i.e., decisions whether proper names represent persons or organizations are made after a number of shared intermediate steps. For instance, for person detection, information on first names and titles obtained from the gazetteer annotations are combined with information on initials, first names, surnames, and endings detected from orthographic characteristics (e.g., capitalization) and PoS tags. Finally, persons' surnames are removed if they contain certain stopwords or can be attributed to an organization. Details about this process can be found in Appendix F of the GATE User Guide[6].

The transducing step is also where we add additional rule-patterns designed to detect band members, releases, and artist names as described in the following section.

3.1 Rule-Pattern Approach

The first approach to extract music-related entities consists of generating specific rules that operate on the annotations obtained in the pre-processing steps. This requires the labor-intense task of manually detecting textual patterns that indicate certain entities in exemplary documents and writing (generalized) rules suited to capture other entities of the same concept also in new documents. For this purpose, for a set of 83 artists/bands, related Web pages such as band profiles and biographies from Last.fm, Wikipedia, and allmusic[7] are examined. Based on the made observations, rules that consider orthographic features, punctuation, surrounding entities (such as those identified via the gazetteer lists), and surrounding keywords are designed. The rules are formalized as so-called *JAPE grammars*[8] that are used in the transducer step of GATE. The complete set of JAPE grammars for music-specific entity recognition can be found in Appendix B of [11] and can also be obtained by contacting the authors. In the following, we show one exemplary (and easily accessible) rule for each concept to demonstrate idea and structure behind the rule-patterns for band member, media, and artist name extraction, respectively.

For the purpose of band member extraction, a JAPE grammar rule that aims at finding band members by searching for information about members leaving or joining the band is given as:

```
Rule : leftJoinedBand (
( ( MemberName ) ) : BandMember
({Token.string == "had"} | {Token.string == "has"})?
({Token.string == "left"} |
 {Token.string == "joined"} |
 {Token.string == "rejoined"} |
 {Token.string == "replaced"})
)--> :BandMember.Member =
     {kind = "BandMember", rule = "leftJoinedBand"}
```

To extract record releases, the following rule matches patterns that start with the potential media name (optionally in quotation marks) and point to production, release, performance, or similar events in the past or future:

```
Rule : MediaPassivReleased (({Token.string == "\""})?
( ( Medium ) ):Media
({Token.string == "\""})?
({Token.string == "was"} |
 ({Token.string == "will"} {Token.string == "be"}))
({Token.string == "released"} |
 {Token.string == "issued"} |
 {Token.string == "produced"} |
 {Token.string == "recorded"} |
 {Token.string == "played"} |
 {Token.string == "performed"} ))--> :Media.Media =
     {kind = "Media", rule = "MediaPassivReleased"}
```

To identify occurrences of band names, the following rule focuses on the entity occurring before terms such as *was founded* or *were supported*:

```
Rule : Formed (
( ( BandN ) ) : BandName({Token.string == "was"} |
{Token.string == "were"})
({Token.string == "formed"} |
 {Token.string == "supported"} |
 {Token.string == "founded"}))--> :BandName.bandname =
     {kind = "Band", rule = "Formed"}
```

Elaborating such rules is a tedious task and (especially in heterogeneous data environments such as the Web) unlikely to generalize well and cover all cases. Therefore, in the next section we describe a supervised learning approach that makes use of automatically labeled data.

3.2 Supervised Learning Approach

Instead of manually examining unstructured text for occurrences of musical entities and potential patterns to identify them, the idea of this approach is to apply a supervised learning algorithm to a set of pre-annotated examples. Using the learned model, relevant information should then be found also in new documents. Several approaches, more precisely several types of machine learning algorithms, have been proposed for automatic information extraction tasks, such as hidden-markov-models [2], decision trees [20], or support vector machines (SVM) [12]. Since the latter demonstrates that SVMs may yield results that rival those of optimized rule-based approaches, SVMs are chosen as classifier for the tasks at hand (for more details see [12, 13])

[6] http://gate.ac.uk/userguide/
[7] http://www.allmusic.com
[8] Acronym for Java Annotation Patterns Engine

For training of the SVMs, a set of documents that contain annotations of the entities of interest is required. Since also this step can be labor intense, we opted for an automatic annotation approach. For the collection of training documents, ground truth information (on band member history and band discography) is obtained by either manually compiling lists or by invoking Web services such as MusicBrainz or Freebase. Using this information, occurrences of the band name, its members (full name as well as last name only), and releases are annotated using regular expressions.

Construction of the features and SVM training is carried out as described by Li et al. [12]. First, for each token, a feature vector representation has to be obtained. In the given scenario, for each token, its content (i.e., the actual string), orthographic properties, PoS information, gazetteer-based entity information, and identified person entities are considered. In a second scenario, in addition to these, also the output of the rule-based approach (more precisely, the name of the rule responsible for prediction of an entity) serves as an input feature. Ideally, this incorporates indicators of high relevance and allows for supervised selection of the manually generated rules for the final predictions. For each prediction task, the corresponding annotation type is also added to the features as target class.

To construct the feature vectors, the training corpus is scanned for all occurring values of any of the considered attributes (i.e., annotations). Then, each token is represented by a vector where each distinct annotation value corresponds to one dimension which is set to 1 if the token is annotated with the corresponding value. In addition, the context of each token (consisting of a window that includes the 5 preceding and the 5 subsequent tokens) is incorporated. This is achieved by creating an SVM input vector for each token that is a concatenation of the feature vectors of all tokens in the context window. To reflect the distance of the surrounding tokens to the actual token (i.e., the center of the window), a reciprocal weighting is applied, meaning that "the nonzero components of the feature vector corresponding to the j^{th} right or left neighboring word are set to be equal to $1/j$ in the combined input vector." [12]. In our experiments, this typically results in feature vectors with approximately 1.5 million dimensions.

In the SVM learning phase, the input vectors corresponding to every single token in all training documents serve as examples. According to the central idea of [12], two distinct SVM classifiers are trained for each concept of interest. The first classifier is trained to predict the beginning of an entity (i.e., to classify whether a token is the first token of an entity), the second to predict the end (i.e., whether a token is the last token of an entity). To deal with the unbalanced distribution of positive and negative training examples, a special form of SVMs is used, namely an SVM with uneven margins [14]. From the obtained predictions of start and end positions, actual entities, as well as corresponding confidence scores, are determined in a post-processing step. First, start tokens without matching end token, as well as end tokens without matching start token are removed. Second, entities with a length (in terms of the number of tokens) that does not match any training example's length are discarded. Third, a confidence score is calculated based on a probabilistic interpretation of the SVM output for all possible classes. More precisely, for each entity, the conjunction of the Sigmoid transformed SVM output probabilities of start and end token is calculated for each possible output class. Finally, the class (label) with the highest probability is predicted for the entity if its probability is greater than 0.25. The probability of the predicted class serves as a confidence score.

3.3 Entity Consolidation and Prediction

From the extraction step (either rule- or learning-based), for each processed text and each concept of interest, a list of potential entities is obtained. For each band, the lists from all texts associated with the band are joined and the occurrences of each entity as well as the number of texts an entity occurs in are counted (term and document frequency, respectively). The joined list usually contains a lot of noise and redundant data, calling for a filtering and merging step. First, all entities extracted by the learning-based method that have a confidence score below 0.5 are removed since they are more likely to not represent band members than representing band members according to the classification step. On the cleaned list, the same observations as described in [19] can be made. For instance, on the list of extracted band members, some members are referenced with different spellings (*Paavo Lötjönen* vs. *Paavo Lotjonen*), with abbreviated first names (*Phil Anselmo* vs. *Philip Anselmo*), with nicknames (*Darrell Lance Abbott* vs. *Dimebag Darrell* or just *Dimebag*), or only by their last name (*Iommi*). On the discography lists, release names are often followed by additional information such as release year or type of release. This is dealt with by introducing an approximate string matching function, namely the level-two Jaro-Winkler similarity, cf. [19].[9] For both entity types, this type of similarity function is suited well as it assigns higher matching scores to pairs of strings that start with the same sequence of characters. In the level-two variant, the two entities to compare are split into substrings and similarity is calculated as an aggregated similarity of pairwise comparison of the substrings. To reduce redundancies, two entities are considered synonymous and thus merged if their level-two Jaro-Winkler similarity is above 0.9. In addition, to deal with the occurrence of last names, an entity consisting of one token is considered a synonym of another entity if it matches the other entity's last token.

This consolidated list is usually still noisy, calling for additional filtering steps. To this end, two threshold parameters are introduced. The first threshold, $t_f \in \mathbb{N}^0$, determines the minimum number of occurrences of an entity (or its synonyms) in the band's set to get predicted. The second threshold, $t_{df} \in [0...1]$ controls the lower bound of the fraction of texts/documents associated with the band an entity has to occur in (document frequency in relation to the total number of documents per band). The impact of these two parameters is systematically evaluated in the following section.

4. EVALUATION

To assess the potential of the proposed approaches and to measure the impact of the parameters, systematic experiments are conducted. This section details the used test collections as well as the applied evaluation measures and reports on the results of the experiments.

[9]For calculation, the open-source Java toolkit *SecondString* (`http://secondstring.sourceforge.net`) is utilized.

4.1 Test Collections

For evaluation, two collections with different characteristics are used – the first a previously published collection used in [19], the second a larger scale test collection consisting of band biographies.

4.1.1 Metal Page Sets

The first collection is a set of Web pages introduced in [19]. This set consist of Google's 100 top-ranked Web pages retrieved using the query *"band name"music members* (cf. Section 2) for 51 Rock and Metal bands (resulting in a total of 5,028 Web pages). In [19], this query setting yielded best results and is therefore chosen as reference for the task of band-member extraction. As ground truth, the membership-relations that include former members are chosen (i.e., the M_f ground truth set of [19]). For this evaluation collection also the results obtained by applying the Hearst patterns proposed in [19] are available, allowing for a direct comparison of the approaches' band member extraction capabilities.

For the discography extraction evaluation, no reference data is available in the original set. Therefore – and since the discography of the contained bands has changed since the creation of the set – a new Web crawl has been conducted to retrieve recent (and more related) data. Since the aim of this new set is to extract released media, for each of the 51 bands in the metal set the query *"band name" discography* is sent to Google and the top 100 pages are downloaded (resulting in a total of 5,090 Web pages). To obtain a discography ground truth, titles of albums, EPs, and singles released by each band are downloaded from MusicBrainz.

To speed up processing of the collections, all Web pages with a file size over 100 kilobyte are discarded resulting in set sizes of 4,561 and 4,625 documents for the member set and the discography set, respectively. Evaluation of the supervised learning approach is performed as a 2-fold cross validation (by splitting the band set and separating the associated Web pages), where in each fold a random sample of 100 documents is drawn for training.

4.1.2 Biography Set

The second test collection is a larger scale collection consisting only of band biographies to be found on the Web. Biographies are investigated as they should contain both information on (past) band members and information on (important) released records.

Starting from a snapshot of the MusicBrainz database from December 2010, all artists marked as bands and all corresponding band members as well as albums, EPs, and singles are extracted. In addition, also band-membership information from Freebase[10] is retrieved and merged with the MusicBrainz information to make the ground truth data set more comprehensive. After this step, band-membership information is available for 34,238 bands. For each band name, the echonest API[11] is invoked to obtain related biographies. Using the echonest's Web service, related biographies (e.g., from Wikipedia, Last.fm, allmusic, or Aol Music[12]) can be conveniently retrieved in plain text format. Since among the provided biographies for a band, duplicates or near-duplicates, as well as only short snippets can be observed, (near-)duplicates as well as biographies consisting of less than 100 characters are filtered out. After filtering (near-)duplicates and snippets, for 23,386 bands (68%) at least one biography remains. In total, a set of 38,753 biographies is obtained. To keep processing times short, furthermore all documents that contain more than 10 megabyte of annotations after the initial processing step are filtered out.

For training of the supervised learner, a random subset of 100 biographies is chosen. All biographies by any artist that is part of the training set are removed from the test set, resulting in a final test set of 37,664 biographies by 23,030 distinct bands.

In comparison to the first test sets, i.e., the Metal page sets, the biography set contains more bands, more specific documents in a homogeneous format (i.e., biographies instead of semi-structured Web pages from various sources), but less associated documents (in average 1.63 documents per band, as opposed to an average of 90 documents per band for the Metal page set).

[10]http://www.freebase.com

[11]http://developer.echonest.com

[12]http://music.aol.com

4.2 Evaluation Metrics

For evaluation, *precision* and *recall* are calculated separately for each band and averaged over all bands to obtain a final score. The metrics are defined as follows:

$$precision = \begin{cases} \frac{|T \cap P|}{|P|} & \text{if } |P| > 0 \\ 1 & \text{otherwise} \end{cases} \quad (1)$$

$$recall = \frac{|T \cap P|}{|T|} \quad (2)$$

where P is the set of predicted entities and T the ground truth set of the band. To assess whether an extracted entity is correct, again the level-two Jaro-Winkler similarity (see Section 3.3) is applied. More precisely, if the Jaro-Winkler similarity between a predicted entity and an entity contained in the ground truth is greater than 0.9, the prediction is considered to be correct. Furthermore, if a predicted band member name consist of only one token, it is considered correct, if it matches with the last token of a member in the ground truth. These weakened definitions of matching allow for tolerating small spelling variations, name abbreviations, extracted last names, additional information of releases, as well as string encoding differences.

For comparison with the Hearst pattern approach for band member detection on the Metal page set, it has to be noted that in [19], calculation of precision and recall is done on the full set of bands and members (and their corresponding roles), yielding global precision and recall values, whereas here, the evaluation metrics are calculated separately for each band and are then averaged over all bands to remove the influence of a band's size. Using the global evaluation scheme, e.g., orchestras are given far more importance than, for instance, duos in the overall evaluation, although for a duo, the individual members are generally more important than for an orchestra. Therefore, in the following, the different approaches are compared based on macro-averaged evaluation metrics (calculated using the arithmetic mean of the individual results).

4.3 Evaluation Results

In the following, the proposed rule-patterns, the SVM approach, as well as the SVM approach that utilizes the out-

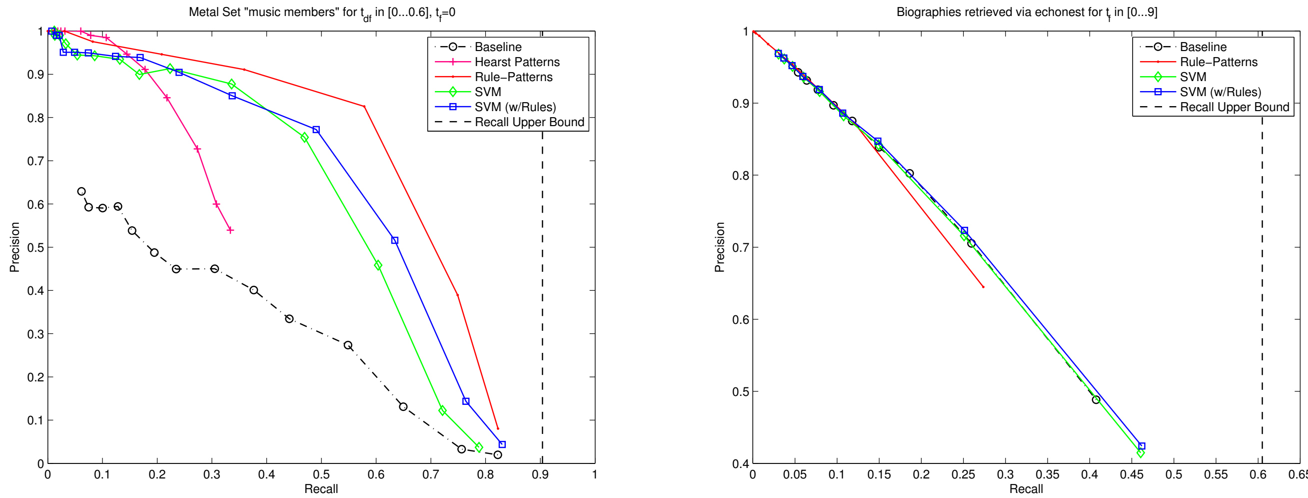


Figure 1: Precision-recall plots for band-member prediction on the Metal page set (left) and on the biography set (right). Curves are obtained by systematically varying threshold parameters (t_{df} and t_f for Metal page set and biography set, respectively). Precision and recall values macro-averaged over all bands in the corresponding test set.

put of the rule-patterns are compared for the tasks of band-member detection and discography extraction. For detecting band-members, a baseline reference consisting of the person entity prediction functionality of GATE is provided. On the Metal page set, band-member prediction is further compared to the Hearst pattern approach from [19]. For the task of discography extraction, no such reference is available. For all evaluations, an additional upper bound for the recall is calculated. This upper bound is implied by the underlying documents, since band members and releases that do not occur on any of the documents can not be predicted.

4.3.1 Band-Member Detection

The left part of Figure 1 shows precision-recall curves for the different band member detection approaches on the Metal page set. For a systematic comparison with the Hearst pattern approach, the t_{df}, i.e., the threshold that determines on which fraction of a band's total documents a band member has to appear on to be predicted, is varied. It can be seen that the rule-based approach clearly performs best. Also SVM and SVM using the rules output outperform the Hearst pattern approach. It becomes apparent that on the Metal set, rule patterns, the GATE person baseline, and the supervised approaches can yield recall values close to the upper bound, i.e., these approaches capture nearly all members contained in the documents at least once. For the Hearst patterns, recall remains low. However, when comparing the Hearst patterns, it has to be noted that this approach was initially designed to also detect the roles of the band members — a feature none of the other approaches is capable of.

Since on the biography set only 1.63 documents per band are available on average, variation of the t_{df} threshold is not as interesting as on the Metal page set. Therefore, the right part of Figure 1 depicts curves of the proposed approaches with varied values of t_f, i.e., the threshold that determines how often an entity has to be detected to be predicted as a band member. On this set, the supervised learning approaches tend to outperform the rule-based extraction approach slightly. However, there is basically no difference between the SVM approaches and the baseline with the only exception that the SVM approaches can yield higher recall values. Another observation is that the upper recall boundary on the biography set is rather low at about 0.6.

4.3.2 Discography Extraction

For discography extraction the situation is similar as can be seen from Figure 2. Also for this task the rule-based approach outperforms the SVM approaches (this time also on the biography set). Recall is also close to the upper bound using SVMs on the Metal page set while on the biography set, none of the approaches is capable of reaching the already low upper recall boundary at 0.36. Conversely, on the biography set, all proposed approaches yield rather high precision values. However, due to the lack of a baseline reference, it is difficult to draw final conclusions about the quality of these approaches for the task of discography extraction.

What can be seen from both the evaluations on discography and band-member extraction is that – despite all work required – rule-patterns are preferable over supervised learning methods. Another consistent finding so far is that SVMs that utilize the output of the rule-pattern classification process are superior to SVMs without this information, but still inferior to the predictions of the rule-patterns alone.

The most unexpected result can be observed for band-member extraction on the biography set. None of the proposed methods outperforms the standard person detection approach by GATE. A possible explanation could be that the baseline itself is already high. Since biographies typically follow a certain writing style and consist — in contrast to arbitrary Web pages — mostly of grammatically well-formed sentences, natural language processing techniques such as PoS tagging perform better on this type of input. Thus, the person detection approach just works better on the biography data than on the Metal page set.

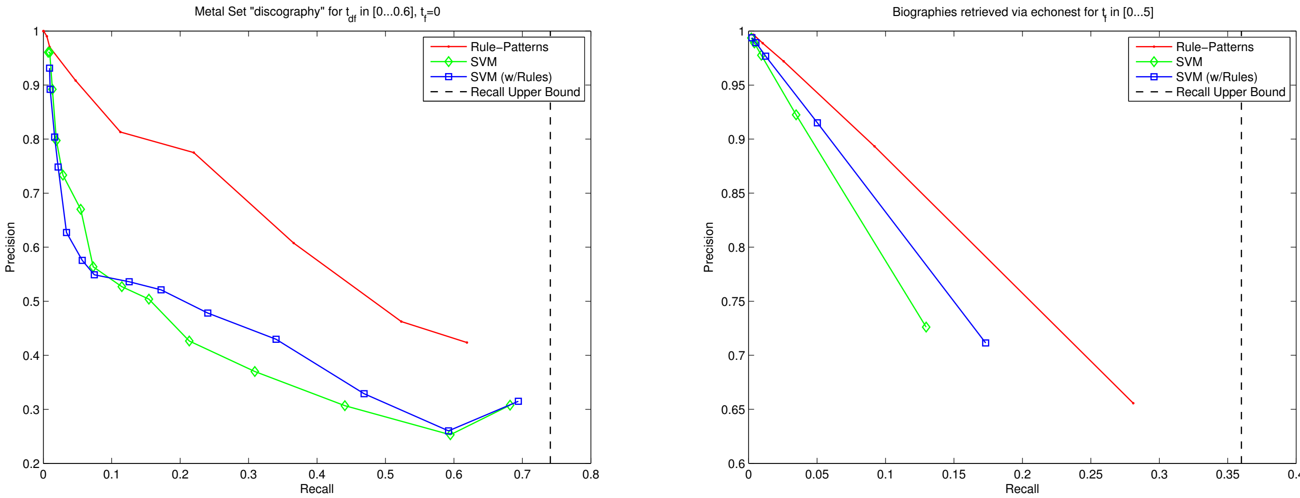


Figure 2: Precision-recall plots for discography detection on the Metal page set (left) and on the biography set (right). Settings as in Figure 1.

In terms of the different sources of data, i.e., the chosen test collections, it can be seen that using biographies, in general lower recall values (and higher precision values) should be expected. This can be seen also from the upper recall bounds that are rather low for both tasks. When using Web documents, more information can be accessed which results also in higher recall values. On the discography Metal set, a recall of 0.7 can be observed which is already close to the upper bound of 0.74. However, using Web documents requires considerations which documents to examine (e.g., by formulating an appropriate query to obtain many relevant pages) as well as dealing with a lot of noise in the data.

4.3.3 Relating Documents to Artists

In addition to the two main tasks of this paper, we also briefly investigate the applicability of the presented methods to identify the central artist or band in a text about music, which could be useful for future relation extraction tasks and tools such as music-focused Web crawling and indexing. To this end, we utilize the rule-patterns aiming at detecting occurrences of artists and train SVMs on occurrences of the name of the band a page belongs to. For prediction, the most frequently extracted entity with occurrences greater than a threshold t_f is selected. As a baseline, simple prediction of any sequence of capitalized tokens at the beginning of the text is chosen. The results can be seen in Figure 3. For this task, SVMs perform better than the rule-patterns. However, rather surprisingly, the highest recall value can be observed for the simple baseline.

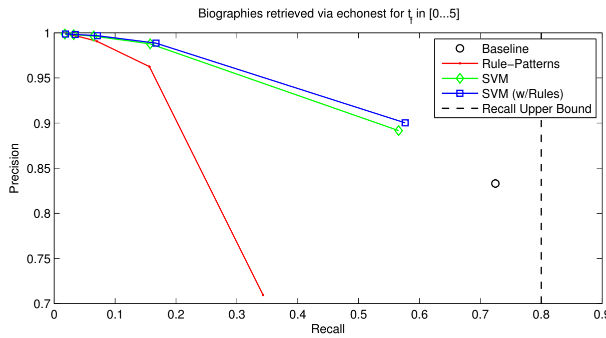


Figure 3: Precision-recall plots for discography detection on the biography set. Curves obtained by varying threshold parameter t_f. Precision and recall values averaged over all pages.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we presented first steps towards semantic Music Information Extraction. We focused on two specific tasks, namely determining the members of a music band and determining the discography of an artist (also explored on sets of bands). For both purposes, supervised learning approaches and rule-based methods were systematically evaluated on two different sets of documents. From the conducted evaluations, it became evident that manually generated rules yield superior results. Furthermore, it could be seen that careful selection of the underlying data source is crucial to achieve reliable results.

In general, the results obtained show great potential for these and also related tasks. By just focusing on biographies, even more highly relevant meta-information on music could be extracted. For instance, consider the following paragraph taken from the Wikipedia page of the *Alkaline Trio*:

“In September 2006, Patent Pending, the debut album by Matt Skiba’s side project Heavens was released. The band consisted of Skiba on guitar and vocals, and Josiah Steinbrick (of hardcore punk outfit F-Minus) on bass. On the album, the duo were joined by The Mars Volta’s Isaiah “Ikey” Owens on organ and Matthew Compton on drums and percussion.”[13]

This short paragraph contains band-membership and line-up information for the *Alkaline Trio*, for the band *Heavens*, for the band *F-Minus*, and for the band *The Mars*

[13] http://en.wikipedia.org/w/index.php?title=Alkaline_Trio&oldid=431587984

Volta. In addition, discographical information for *Heavens*, genre information for *F-Minus*, and a nickname/alias for *Isaiah Owens* can be inferred from this small piece of text. Furthermore, relations between the mentioned bands ("side-project") as well as the mentioned persons (collaborations) can be discovered. Using further information extraction methods, in future work, it should be possible to capture at least some of this semantic information and relations and to advance the current state-of-the-art in music retrieval and recommendation. However, for systematic experimentation and targeted development, the creation of a comprehensive and thoroughly (manually) annotated text corpus for music seems unavoidable.

6. ACKNOWLEDGMENTS

Thanks are due to Andreas Krenmair for conceiving the music-related JAPE patterns and sharing his implementation. This research is supported by the Austrian Research Fund (FWF) under grants L511-N15 and P22856-N23.

7. REFERENCES

[1] H. Alani, S. Kim, D.E. Millard, M.J. Weal, W. Hall, P.H. Lewis, and N.R. Shadbolt. Automatic Ontology-Based Knowledge Extraction from Web Documents. *IEEE Intelligent Systems*, 18(1):14–21, 2003.

[2] D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. Nymble: a High-Performance Learning Name-finder. In *Proc. 5th Conference on Applied Natural Language Processing*, 1997.

[3] E. Brill. A Simple Rule-Based Part of Speech Tagger. In *Proc. 3rd Conference on Applied Natural Language Processing*, 1992.

[4] J. Callan and T. Mitamura. Knowledge-Based Extraction of Named Entities. In *Proc. 11th International Conference on Information and Knowledge Management (CIKM)*, 2002.

[5] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proc. 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002.

[6] G. Geleijnse and J. Korst. Web-based artist categorization. In *Proc. 7th International Conference on Music Information Retrieval (ISMIR)*, 2006.

[7] E. Gómez and P. Herrera. The song remains the same: Identifying versions of the same piece using tonal descriptors. In *Proc. 7th International Conference on Music Information Retrieval (ISMIR)*, 2006.

[8] S. Govaerts and E. Duval. A Web-Based Approach to Determine the Origin of an Artist. In *Proc. 10th International Society for Music Information Retrieval Conference (ISMIR)*, 2009.

[9] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. 14th Conference on Computational Linguistics - Vol. 2*, 1992.

[10] P. Knees. *Text-Based Description of Music for Indexing, Retrieval, and Browsing.* PhD thesis, Johannes Kepler Universität, Linz, Austria, 2010.

[11] A. Krenmair. Musikspezifische Informationsextraktion aus Webdokumenten. Diplomarbeit, Johannes Kepler Universität, Linz, Austria, 2010.

[12] Y. Li, K. Bontcheva, and H. Cunningham. SVM Based Learning System for Information Extraction. In J. Winkler, M. Niranjan, and N. Lawrence, eds., *Deterministic and Statistical Methods in Machine Learning*, vol. 3635 of *LNCS*. Springer, 2005.

[13] Y. Li, K. Bontcheva, and H. Cunningham. Adapting SVM for Data Sparseness and Imbalance: A Case Study on Information Extraction. *Natural Language Engineering*, 15(2):241–271, 2009.

[14] Y. Li and J. Shawe-Taylor. The SVM with uneven margins and Chinese document categorization. In *Proc. 17th Pacific Asia Conference on Language, Information and Computation (PACLIC)*, 2003.

[15] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson. The Music Ontology. In *Proc. 8th International Conference on Music Information Retrieval (ISMIR)*, 2007.

[16] M. Schedl and P. Knees. Context-based Music Similarity Estimation. In *Proc. 3rd International Workshop on Learning the Semantics of Audio Signals (LSAS)*, 2009.

[17] M. Schedl, P. Knees, and G. Widmer. A Web-Based Approach to Assessing Artist Similarity using Co-Occurrences. In *Proc. 4th International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2005.

[18] M. Schedl, C. Schiketanz, and K. Seyerlehner. Country of Origin Determination via Web Mining Techniques. In *Proc. IEEE International Conference on Multimedia and Expo (ICME): 2nd International Workshop on Advances in Music Information Research (AdMIRe)*, 2010.

[19] M. Schedl and G. Widmer. Automatically Detecting Members and Instrumentation of Music Bands via Web Content Mining. In *Proc. 5th Workshop on Adaptive Multimedia Retrieval (AMR)*, 2007.

[20] S. Sekine. NYU: Description of the Japanese NE system used for MET-2. In *Proc. 7th Message Understanding Conference (MUC-7)*, 1998.

[21] Y. Shavitt and U. Weinsberg. Songs Clustering Using Peer-to-Peer Co-occurrences. In *Proc. IEEE International Symposium on Multimedia (ISM): International Workshop on Advances in Music Information Research (AdMIRe)*, 2009.

[22] M. Slaney and W. White. Similarity Based on Rating Data. In *Proc. 8th International Conference on Music Information Retrieval (ISMIR)*, 2007.

[23] M. Sordo, C. Laurier, and O. Celma. Annotating Music Collections: How Content-based Similarity Helps to Propagate Labels. In *Proc. 8th International Conference on Music Information Retrieval (ISMIR)*, 2007.

[24] D. Turnbull, L. Barrington, and G. Lanckriet. Five Approaches to Collecting Tags for Music. In *Proc. 9th International Conference on Music Information Retrieval (ISMIR)*, 2008.

[25] B. Whitman and S. Lawrence. Inferring Descriptions and Similarity for Music from Community Metadata. In *Proc. International Computer Music Conference (ICMC)*, 2002.

The Importance of Service and Genre in Recommendations for Online Radio and Television Programmes *

Ian Knopke†
British Broadcasting Corporation
201 Wood Lane, White City
London, UK
ian.knopke@gmail.com

ABSTRACT

The BBC iPlayer is an online delivery system for both radio and television content [1]. One of the unique features of the iPlayer is that programming is based around a seven day "catch-up" window. This paper documents some early investigations into features that may be used to produce quality recommendations for that system. The two features explored here, *services* and *genre*, are partly unique to BBC metadata, and are available for all programmes in the schedule. Services are roughly equivalent to channels or stations, while genres are editorially-assigned categorisations of media content. Results of genre / service-based diversity are presented, as well as some simple recommenders based on there, and additional discussion of the topic and results.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Recommendations

Keywords

Recommendations, Broadcasting, Collaborative Filtering

1. INTRODUCTION

The BBC iPlayer is an online delivery system for both radio and television content. Freely available for users within the geographical borders of the United Kingdom, it has been immensely successful and is used by millions of people each day. Unlike similar systems from commercial broadcasters, the BBC's system is provided without advertising.

*(Produces the permission block, and copyright CC 3.0 information). For use with SIG-ALTERNATE.CLS. Supported by ACM.
†

WOMRAD 2011 2nd Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2011 (Chicago, US)

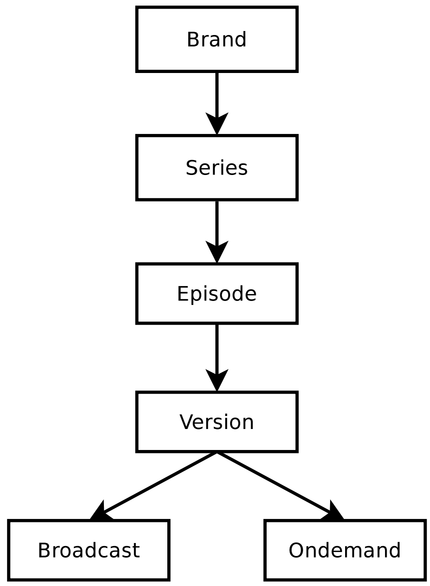


Figure 1: BBC Programme Hierarchy

One of the unique features of the iPlayer is that programming is based around a seven day "catch-up" window. Programming is first shown as a linear broadcast over normal transmission systems (radio and tv). Shortly thereafter the same content becomes available online, without charge, for a period of one week. The BBC maintains a near-perfect synchronicity between their linear and online broadcasting worlds, with over 95% of linear content available as "catch-up" internet television or radio on many different gaming consoles, integrated television platforms and mobile devices, as well as desktop and laptop computers. This synchronicity is completely integrated at both the metadata and transcoding levels, and across both radio and television.

A simplified diagram of the BBC programme metadata hierarchy is shown in Figure 1. The most important element for purposes of this paper is the episode. Episodes may be edited into different versions, and then sent out as transmitted broadcasts or made available online as ondemands. Episodes are grouped into series, under a particular brand. Brands are equivalent to what a user might find listed in a programme guide; common UK examples are EastEnders or Dr. Who (tv) or Desert Island Discs (radio).

This paper documents an investigation into features that may be used to produce quality recommendations. The two features explored here, services and genre, are partly unique to BBC metadata, but genres are also used in other music and media recommendation systems. While there is a large body of research into content-based features for music recommendation, it should be noted that the research presented here is entirely based on metadata, user histories, and the BBC programme hierarchy.

2. RECOMMENDATION SYSTEMS FOR THE BBC IPLAYER

2.1 Previous Issues and Possible Solutions

Most recommendation systems generate recommendations by identifying similar users based on their recorded product choices, and then identifying products popular with these users that a new, similar user has not yet chosen. This is often referred to as *collaborative filtering*. Amazon and last.fm are two examples of such systems [4], and there are many variants [7, 5].

These systems have proven to be effective in many commercial environments, leading to increased site traffic, sales, and an improved connection between individual users and the items that they are interested in. However, a system of this type was recently incorporated into the BBC iPlayer product and failed to produce similar behaviour, with a daily usage rate of approximately 4% of episode click-throughs. It is useful to examine some of the reasons why a technique that has been successful in other online contexts would perform so poorly in the case of the BBC. Particular issues with standard collaborative filtering systems include:

Dynamic Programme Schedule Most online stores have a collection of items, such as books or songs, that are largely unchanging. While new items are often added, the amount of new material in relation to the majority of the collection is small enough that one can consider it to be relatively static. In practice, the relatively small number of new items added can be handled through weekly or daily recalculations of recommendations across the entire product set / user histories. In contrast, the list of iPlayer ondemands is primarily limited to a seven day availability window. The composition of programs within this window changes dynamically, with new programmes being added at least every hour, and older ones expiring. The list of valid programmes, and effectively the viewer's history of programmes to recommend against only extends back a week. In effect, a completely new set of programmes is introduced every week, making it difficult to leverage the user's play history towards generating new recommendations.

Cold Start Problem In the classical collaborative filtering model, new items do not get recommended until enough users have discovered them through other means. This is really just another aspect of the sparse data problem, where there is not enough user history to make adequate recommendations [3]. New items are often introduced to users through mechanisms such as promotions, or through partial solutions such as artificially introducing non-personalised defaults based on average user ratings of all products [2]. In contrast, the short availability window of iPlayer programmes effectively meant that existing programmes never left this "build-up" phase of generating enough history with which to make effective recommendations. In most cases new programmes often weren't recommended until they were near the end of their availability windows. It is an extremely unfortunate situation to have the BBC place considerable effort into creating world-class content, and then not recommend it for the majority of that programme's availability, or perhaps not at all.

Eliminating Old Content In a typical collaborative filtering system, removing items requires recalculation of the mathematical relationships between all users and products (or just products to products). This is a computationally-expensive process, and consequently most online stores only remove products from their catalogues infrequently. If necessary, invalid results can be temporarily filtered until such time as a system-wide batch recalculation can be accomplished. In some cases removal of items can cause referential integrity (foreign key) issues, and many collaborative filtering systems apparently do not have mechanisms for removing content at all. This led to many programmes being recommended that were no longer available, and required the implementation of an expensive, secondary real-time filtering system to remove expired recommendations.

2.2 Possible Solutions

One obvious but partial solution to these problems would be to filter the output results to only produce recommendations within the current time window. While this would alleviate the problem of producing expired recommendations, it does not solve other issues such as the cold start problem.

Another approach, and the one explored here, is to instead find more general categorisations for programmes. If all programmes in the current schedule can be assigned to a set of static categories, these can then be used to record user histories against. The experiments in this paper explore the potential of two such features, services and genre, for use in storing cumulative user histories. These have the advantage of being assigned to all radio and television programmes in the BBC linear schedule and are readily available.

2.3 Services

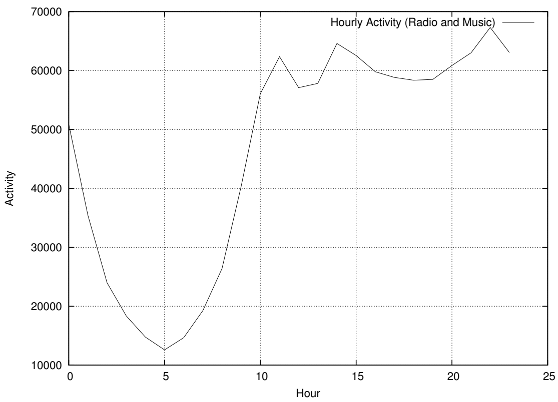
In linear broadcasting, a service is a particular station or channel such as "BBC One" or "6 Music". In the world of online "catch-up" broadcasting services tend to function more as an association of programmes that share some common heritage. The reasons for this are partly historical, but these divisions are also still valid from an audience perspective; the original channel structures were created to fulfill different audience requirements. For instance, "6 Music" tends to focus on very new music, while the "BBC Four" radio audience is more classically oriented. However, one of the advantages of online broadcasting is that audience members have the ability to switch between services more easily than ever before. When removed from the restraints of the linear schedule, one would expect to see users take advantage of this and new listening trends and patterns to be reflected in user play histories.

2.4 Genres

Table 1: Common BBC Services and Genres

Services	Genres
bbc_1xtra	childrens
bbc_6music	religion_and_ethics
bbc_7	entertainment
bbc_london	drama
bbc_radio_five_live	factual
bbc_radio_one	weather
bbc_radio_three	music
bbc_radio_two	sport
bbc_three	news
bbc_world_service	comedy

Figure 2: Accumulated Daily Online Radio Activity



Every BBC programme, both television and radio, has at least one genre assigned to it by an expert editorial staff member. These are used in a variety of marketing and promotional functions, as well as for programming, and are considered to be accurate in the broadcasting industry.

A list of some common BBC services and genres is given in Table 1. While the properties of services and genre in relation to the linear broadcast audience is well known, similar information about online usage is not as well evaluated. Both features, however, are thought to be influential in the online domain. The value of these for recommending online programming remains relatively unevaluated in an empirical way.

3. EXPERIMENT

We performed two kinds of experiments. First, the diversity of genres and services were tested. Based on this, four simple recommendation systems were evaluated for how close a match they were to a historical dataset.

A month of iPlayer play history was made available from May 28 to June 25, 2010, consisting of approximately 18 million instances of user selected ondemands, with most shows lasting a half or full hour. Of this, approximately 17 million are televised selections and 1 million are radio. Daily online radio and television usage patterns, averaged over the time period are given in Figures 2 and 3 respectively.

After some discussion and initial exploration, it was decided to test these factors based on the diversity of user play history. To test the diversity of both services and genres, a play history of 89,574 radio and 747,992 television users for the above time period were extracted, and the diversity of each user's individual history was calculated. While other more complex diversity evaluation systems are available [6], three common measures of diversity were used: Gini impurity, entropy (2), and a standard classification error using the maximum value (3).

Figure 3: Accumulated Daily Online Television Activity

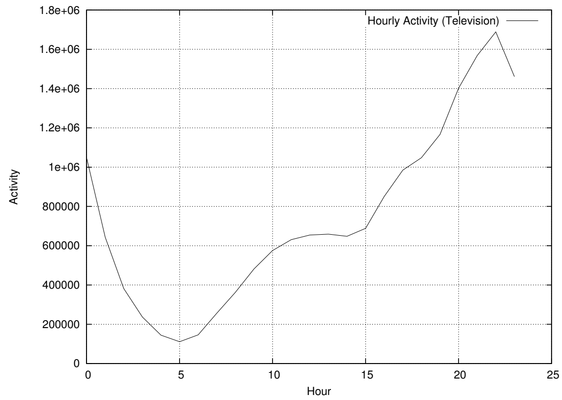


Table 2: Diversity of BBC Services

	Radio / Music	TV
Gini	0.03	0.25
Entropy	0.07	0.6
Classification Error	0.025	0.19

$$\text{gini}(t) = 1 - \sum_{i=0}^{c-1} p(i|t)^2 \tag{1}$$

$$\text{entropy}(t) = -\sum_{i=0}^{c-1} p(i|t) log_2 p(i|t) \tag{2}$$

$$\text{maxclasserror}(t) = 1 - max_i p(i|t) \tag{3}$$

Table 2 shows the averaged values for all users. For comparison purposes, similar figures were also calculated for the television users. These results clearly show that the majority of individual radio users concentrate around a very small number of services, with very little diversity. Television users, on the other hand, tend to have much more diverse service histories and do not appear to be as tied to particular services in the online world. Similar figures for genre are show in Table 3 and to a lesser degree exhibit the same trends.

Based on these results, four simple recommendation strategies were tested for recommending radio and music programmes. Recommendations were based on:

Table 3: Diversity of BBC Genres

	Radio / Music	TV
Gini	0.15	0.37
Entropy	0.32	0.93
Classification Error	0.14	0.31

Figure 4: Markov Chain built from genres using BBC 3

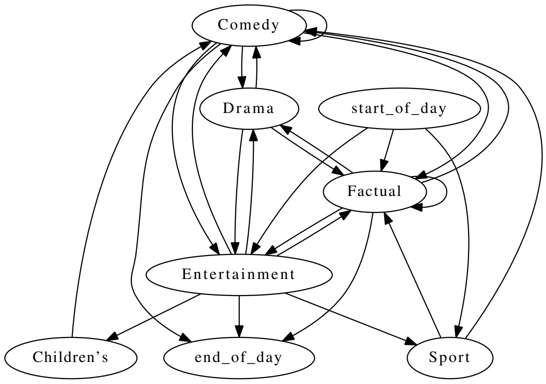


Table 4: Results of Simple Recommenders

Last programme	.06
Most common	.14
Markov	.28
Markov w/services	.34

- The genre of last programme
- The most common genre in the user's history
- A Markov chain of genres derived from all linear broadcast schedules
- Individual Markov chains of genres for each service

The inclusion of Markov chains requires some explanation. The order of programmes is traditionally an important factor in the scheduling of linear broadcasts, with the intention of sustaining audience interest for longer time periods. Consequently, a simple Markov chain based on successive genres was constructed using the linear schedules. Effectively this reduces to a probability distribution for each genre where the most likely genre was compared to that of the next item in the user's history. Note that start and end-of-day states were inserted to represent the 6 AM daily schedule changeover, as no connection is implied between days. In the case of the fourth recommender, individual Markov chains were built for each service and resolved using the service of the previous programme. As an example, Figure 4 shows a simple Markov chain built on successive genres for BBC 3.

Each recommender was then tested on each user's play histories in sequence and a tally of matches / failures kept. These were evaluated using the user's past histories as simple percentages, as shown in Table 4.

4. DISCUSSION

While none of the strategies tested could be considered a complete recommendation system, it is surprising that correct results can be obtained more than a third of the time using only these two simple features, and a knowledge of the programmes found in the linear schedule. One possible way to interpret this is that the online audience shares some of the behaviour of the linear scheduling audience, even when freed of the constraints of only having a single content choice at any one time.

Also, the use of the original service has more of an impact in a radio context than in a television context. To be sure, there are significant differences between television and radio as media formats, and in many ways are not comparable. Nevertheless, it is interesting to try. One possible interpretion is that television viewers have embraced the online experience to a greater extent than pure music or radio listeners. However, it may also be that radio users are more loyal in general to particular stations/brands than television users for other reasons besides just the music. For instance, online radio stations such as last.fm specialise in automatically generating curated collections of music. Disregarding any differences between their recommendations and those programmed by the human curators at the BBC, the main difference is the other elements such as presenters and news segments, and these may be what keeps listeners from changing services.

Genre is also useable for radio recommendations, but genre as a single feature appears to work better for recommending television programmes.

5. FUTURE DIRECTIONS

While this was more on the order of an initial exploration of the problem space, the work presented here suggests a number of additional areas of research. It seems clear that time of day is also probably an important factor. We would also like to do better comparisons between the linear and online audience behaviours, as it seems that there is probably a fair amount of common behaviour there. Also, the study should be expanded to include additional features.

6. REFERENCES

[1] BBC. iPlayer, 2011. `http://www.bbc.co.uk/iplayer/`.

[2] J. S. Breese, D. Heckerman, and C. M. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, Madison, WI, 1998. Morgan Kauffman.

[3] C.-N. Hsu, H.-H. Chung, and H.-S. Huang. Mining skewed and sparse transaction data for personalized shopping recommendation. *Machine Learning*, 57(1-2):35–59, 2004.

[4] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[5] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 285–95, 2001.

[6] M. Slaney and W. White. Measuring playlist diversity for recommendation systems. In *Proceedings of the ACM Workshop on Audio and Music Computing for Multimedia*, pages 22–32, Santa Barbara, CA, USA, 2006. ACM.

[7] X. Su and T. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:1–20, 2009.

Probabilistic Game Theoretic Algorithms for Group Recommender Systems

George Popescu
EPFL - HCI Group
IC IIF Station 14
1015 Lausanne, Switzerland
+41 21 693 1246
george.popescu@epfl.ch

Pearl Pu
EPFL - HCI Group
IC IIF Station 14
1015 Lausanne, Switzerland
+41 21 693 6081
pearl.pu@epfl.ch

ABSTRACT
Aggregating users' individual preference and recommending a common set of items for a group has become a challenging topic in group recommender systems and social websites. This issue is mainly concerned with the following three objectives: eliciting individual users' preferences, suggesting outcomes that maximize the overall satisfaction for all users and ensuring that the aggregation mechanism is resistant to individual users' manipulation. Firstly we show how our proposed probabilistic weighted-sum algorithm (PWS) works and emphasize on its advantages. Then we compare PWS with related approaches implemented in similar systems using the case of our music recommender, GroupFun. We describe an experiment design to study users' perceptions of the algorithms, their perceived fairness and incentives to manipulate the final recommendation outcome. We expect our results to show that PWS will be perceived as fair and diversity- and discovery-driven, thus enhancing the group's satisfaction. Our future work will focus on the actual evaluation of GroupFun using the experiment design presented here.

Categories and Subject Descriptors
H1.2 [**User/Machine Systems**]: human factors; H5.2 [**User Interfaces**]: evaluation/methodology, user-centered design.

General terms
Experimentation, Human factors.

Keywords
Quality measurement, usability evaluation, recommender systems, quality of user experience, post-study questionnaire.

1. INTRODUCTION
Group recommender systems use various aggregation strategies to suggest a common list of items to a group of users. These strategies aim at increasing the group's welfare and are based on users' votes on items. The social welfare is an aggregate of individual utilities of all group members. Most common used deterministic strategies are: plurality voting, utilitarian, approval voting, least misery, most pleasure, average without misery, fairness, most respected person, Borda count, Copeland rule or Kemeny scores (Masthoff, 2005). One can easily create other distinct strategies based on these. Social choice theory aims to offer an answer to "which strategy is most effective and will be most liked by a group of users?" (Hastie and Kameda, 2005). With the purpose of determining what strategy people actually use, Masthoff (2004) found that individuals use computationally simple strategies mentioned above, particularly the average strategy, the average without misery and the least misery strategy. However, there is no dominant strategy as people switch between them given a different context. Fairness plays an important role in decision making but members do not have a clear strategy for applying it.

Our main research question is to determine "which group satisfaction rule best satisfy users expectations". We propose 4 algorithms and investigate upon: "which algorithm is best suited to meet users' expectations" for our music recommender system, GroupFun and "how users perceive the algorithms' accuracy". Next we present related work, then considered algorithms together with our implementation and future experiment design.

2. BASELINE AND RELATED WORK
2.1 MusicFX
MusicFX is a music system offering best music match to employees working out in a fitness center (McCarthy and Anagnost, 1998). The algorithm aims at selecting the most preferred music genre that maximizes members' listening pleasure. For this it computes a group preference index and sums squared individual preferences. Then it lists the most popular categories. The system also saves events in its history such as: member entrance, member exit, individual preference update, system parameter adjustment and maximum station play time elapsed. Since some individual preference filters may not change, the system opts for a different music configuration according to two criteria: playing more the music which members like most and playing less the music which members like least. The weighted random selection operator is one strategy used to reduce the likelihood of starvation. Another strategy is limiting the period of time for one genre to be played – regardless of how popular it is – before the selection algorithm is invoked in order to select a new station. MusicFX has two important advantages: (1) it increases the variety of music and (2) it democratizes the music selection process. Thus it is adaptive to changing preferences of its users also proposing new songs for them. One drawback of the system is that it changes music stations abruptly in the middle of the songs.

2.2 PolyLens
PolyLens is a collaborative filtering recommender system which recommends movies to groups of people based on their individual preferences (O'Connor et al. 2001). It represents a group

WOMRAD 2011 2ndWorkshop on Music Recommendation and Discovery, colocated with ACM RecSys 2011 (Chicago, US)

extension of the MovieLens recommender system with over 80,000 users and their ratings for more than 3,500 movies (with a total of nearly 5 million votes). Users can create and manage groups, access individual and group recommendations and receive notification alerts for group invitation. The algorithm uses the least misery strategy given that the groups formed to watch a movie together tend to be small. As such the group is as happy as its least happy member. The authors mention that "the social value function and algorithm are unlikely to work well for large groups". They further note that "it is still an open research question to understand the types of social value functions that best satisfy large groups and to implement them algorithmically".

2.3 Voting strategies

An extensive study on a group of television viewers aiming at finding which strategy people use was realized by Masthoff (2005). In the experiment 10 deterministic group voting rules are compared: plurality voting, utilitarian strategy, Borda count, Copeland rule, approval voting, least misery strategy, most pleasure strategy, average without misery strategy, fairness strategy and most respected person strategy. The experiment shows that individuals do not use a clear dominant strategy, but average, average without misery, and least misery are all plausible candidates for implementation. In a different experiment addressing how people judge the recommendation results multiplicative utilitarian strategy is the most promising strategy, but the other strategies received close scores. In the study of television viewers the hypothesis that social status influences selection has no statistical dominance. Non-linear utility suits better users' expectations than a linear one. For instance quadratic rating scale is appropriate for implementation. Furthermore, there is strong evidence that human subjects use a series of simple strategies in different judgment contexts they face. For instance, if one takes the satisfaction of the group to be the average of the satisfaction of the individuals, then the average strategy performs best. Taking the minimum better corresponds to the predictions which are made by individuals assessing their own needs.

3. ALGORITHMS

In the music domain many users usually form many groups and listen to many songs. Given the fact that the length of one song is of 3 to 4 minutes users usually select a playlist containing several to lots of songs. This is not the case of movies selection when users need to agree on only one or few movie(s) they would like to consume given their limited time and the length of a movie: ~2h. Thus, the music domain presents both opportunities and challenges since the recommendation needs to focus on both diversity and accuracy.

We propose the following 4 algorithms for comparison:

- PS (Probabilistic Sum): select the common playlist' songs probabilistically, each of them having the same probability to be selected
- LM (Least Misery): select songs with the highest minimum individual ratings
- DWS (Deterministic Weighted Sum): deterministically select songs with the highest score
- PWS (Probabilistic Weighted Sum): compute weighted sum and select songs based on their score probabilities.

3.1 General framework

Let A be the set of all users and S the set of all possible outcomes that can be rated. In our group music recommendation setting, the outcomes are songs s_i that are selected in the common playlist. We let each user a_j submit a numerical vote $score(s_i, a_j)$ for each song s_i that reflects their preference for that song. These votes are given as ratings on a 5-point Likert scale and normalized so that the scores given by each user sum to 1:

$$score(s_i, a_j) = \frac{rating(s_i, a_j)}{\sum_i rating(s_i, a_j)} \quad (1)$$

We then assign a joint score to each song that is computed as the sum of the scores given by the individual users:

$$score(s_i) = \sum_{a_j \in A} score(s_i, a_j) \quad (2)$$

To choose the songs to be included in a playlist of length k, a deterministic method is to choose the k songs with the highest joint rating: weighted sum (DWS):

$$score(s_i) = \frac{score(s_i)}{\sum_{s_i \in S} score(s_i)} \quad (3)$$

The probabilistic weighted sum (PWS) iteratively selects each of the k songs randomly according to the probability distribution:

$$p(s_i) = \frac{score(s_i)}{\sum_{s_i \in S} score(s_i)} \quad (4)$$

By comparison, the probabilistic sum (PS) method chooses the k songs with equal probability:

$$p(s_i) = \frac{1}{|S|} \quad (5)$$

The least misery (LM) method takes into account the minimum of ratings for each user:

$$\min\left(score(s_i, a_j)\right), \forall a_j \in A \quad (6)$$

3.2 Example

To illustrate how each algorithm works, we consider the following example. In the next table, user1, user2, and user 3 represent group members. The score distribution normalized to 1 for each of the users is displayed in the respective row, and the joint scores are shown in the table below.

Table I. Item selection example using the 4 algorithms

User1	Song1: 0.1	Song2: 0.4	Song3: 0.4	Song4: 0.1
User2	Song1: __	Song2: 0.2	Song3: __	Song4: 0.8
User3	Song1: 0.4	Song2: 0.2	Song3: __	Song4: 0.4
Total	Song1: 0.5	Song2: 0.8	Song3: 0.4	Song4: 1.3

The least misery (LM) will choose song 2 and song 3 (each of them has the minimal rating 0.2). For all other songs the minimum score is 0.1. After normalizing the total scores by the sum of scores, we obtain the following probability distribution for the set of outcomes.

Table II. Probability distribution

P	Song1: 0.16	Song2: 0.26	Song3: 0.13	Song 4: 0.43

Considering the probability as the final score, the deterministic weighted sum (DWS) will chose songs 4, 2, 1 and 3. Probabilistic weighted sum (PWS) will choose one song after another using this probabilistic distribution. Compared to other social choice based algorithms, PWS is incentive compatible. That is, it is to the best interest of the individual to reveal his/her preferences truthfully. It is in fact equivalent to a random dictator method, where the dictator will choose a song randomly with the probabilities given by its degree of preference – a reasonable method since nobody wants to hear the same song over and over again. This is because the probability of a song s_i to be chosen can be written as:

$$p(s_i) = \frac{score(s_i)}{|A|} = \sum_{a_j \in A} \frac{score(s_i, a_j)}{|A|} \qquad (7)$$

or, in other words, the probability of choosing user a_j times the normalized score that user a_j has given to song s_i. Indeed, User3's preference for song 1 yields a significant probability that this song will be included in the playlist, relative to other songs.

3.3 Discussion

The contribution of the PWS algorithm in the paper stands out with respect to group satisfaction. We expect users to be more satisfied using PWS than other algorithms given their expectations to discover the music of other members.

Advantages of PWS compared with the other algorithms:

1. Users are free to choose the number of songs
2. Ratings are updated permanently
3. The algorithm is computationally simple
4. Users can negotiate their ratings and trade utility
5. Incentive-compatible truthful property is observed
6. The algorithm favors music diversity

The disadvantages of PWS are:

1. It is difficult to quantify rating differences between distinct users. The weights given by each user cannot be compared with the ones given by another since users have different estimations of their utility.
2. Self-selection effect: most popular songs will receive most votes - not ideal if long tail distribution is desired.

Since PWS can be interpreted as similar to the random scheme users have to test it in more recommendation rounds to understand its inner logic. PWS can be further developed to include the group dynamics. One solution is to consider trust and other members' comments on the songs rated by one user (e.g. "like"/"dislike").

The PWS algorithm stands out with respect to allowing users to engage in trustful individual preference elicitation and music discovery. By returning to the recommendation list the group will find a different playlist every-time they are would like to listen to group music. By considering the probabilistic distribution of ratings and an extensive music library the algorithm will mostly suggest songs strongly liked by most others. Sometimes it will recommend unexpected, rating-wise, serendipitous items facilitating music discovery and group enjoyment.

4. GROUPFUN

GroupFun is a web application that helps a group of friends to agree on a common music playlist for a given event they will attend, e.g. a birthday party or a graduation ceremony. Firstly, it is implemented as a Facebook plugin connecting users to their friends. Secondly, it is a music application that helps individuals to manage and share their favorite music with groups. In GroupFun users can listen to their own collection of songs as well as their friends' music. With the collective music database, the application integrates friends' music tastes and recommends a common playlists to them. Therefore, the application aims at satisfying music tastes of the whole group by aggregating individual preferences through the use of previously presented algorithms.

Figure 1. "Home" page of GroupFun

In the "Home" page users see 4 playlists: one from a recent event, one containing popular songs, one from a party and the last one from an older event. They can listen to each song in each of the playlists.

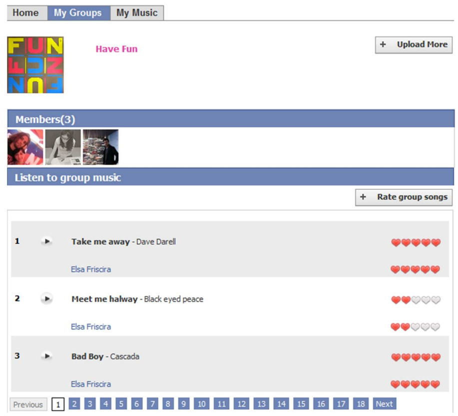


Figure 2. "My Groups" page of GroupFun

In the "My Group" page users can create groups, upload and rate their music, invite friends and hear the group's songs. Finally, in the "My Music" page users see their contribution to GroupFun: for each song is displayed the associated group, the rating and its name and artist. Users can also listen to their individual preferences in the same interface. One of the most important characteristics of GroupFun is that it combines music, friends and groups together. In other words, GroupFun serves as a platform allowing users to conveniently organize their individual music library, effectively communicate with friends and actively participate in social activities.

5. EXPERIMENT DESIGN

To compare how users perceive the 4 algorithms, we plan to carry on a between-groups user study. With the results of the experiment we will be able to make a judgment of the influence of both the algorithms and the design on users' satisfaction. We plan to collect solid user feedback regarding how an algorithm should allow group members to arrive at a common decision in a music recommendation setting.

Our hypotheses are that: (1) users will not reveal their preferences strategically as to influence the algorithm's outcome; (2) they will prefer more PWS than DWS given the increased diversity of the recommendations they receive and (3) the group will perceive more overall satisfaction but less diversity using the LM algorithm compared with PWS.

5.1 Procedure

First we recruit university students, friends who have Facebook accounts and other users on the Amazon Mechanical Turk platform. All of them will use their own computers in order to connect to the GroupFun application. We consider 4 algorithms implemented for 4 groups of 40 users together with 1 interface. Each of the algorithms displays a common list of 10 songs to all group members. Users are asked to contribute their music to only one of the groups and fill in an online post-study questionnaire assessing their satisfaction. The final music outcome is shown to all group members after they have finished their tasks. Users can interact with the system in diverse ways such as: upload more or less songs, change their ratings, see and hear to their friends' playlists, etc.

Table III. Evaluation of 4 algorithms using the same interface

Interface/Algorithm	**PWS**	**DWS**	**LM**	**PS**
Interface	40	40	40	40

5.2 Measurements

The first 40 users see the results of the probabilistic weighted sum algorithm, the next those of deterministic weighted sum and so on. Since individuals who upload more songs expect to see their song names more often in the final list they would prefer to know a priori the computation rule of the algorithm so that they would adjust the number of songs they upload. Given users' known self-ratings and group ratings computed by the algorithms we expected our subjects to identify some differences between the 4 approaches. Some of the questions from the post-study questionnaire are presented in the table below. They were extracted from a well-known user evaluation model, named ResQue [7], that our group has developed.

Table IV. Evaluation questions

Measurements	*Questions*
Perceived attractiveness	The layout of the system's interface is attractive.
Perceived satisfaction	The items recommended to me matched my interest.
Perceived helpfulness	I took into account the ratings given by my friends.
Outcome change intention	I was interested in changing the outcome of the algorithm.

6. CONCLUSIONS AND FUTURE WORK

In this paper we presented our research work on the algorithmic development and evaluation of our music recommender system and Facebook application, GroupFun. The major contribution of this paper is the demonstration of the applicability of the PWS algorithm for group recommendation strategies and negotiation. In this context, we analyzed different group recommendation approaches w.r.t. group satisfaction and discussed key satisfaction issues to be taken into account. The PWS algorithm we proposed calculates probabilities for songs to appear in groups' playlists favoring music diversity and discovery. Using PWS users state their preference truthfully. They align their decision to that of the group. Furthermore, our current development of GroupFun allows users to create groups, rate and share their music profiles with their friends.

To understand how users' perceive our algorithms and current interface, we plan to conduct an experiment to compare the 4 algorithms in a between-subjects study. As such we will evaluate user satisfaction for music group recommendations. Furthermore, to learn more about the perceived ease of use and perceived usefulness of our system we plan to invite more members and analyze user feedback in terms of design and functionality. We also intend to develop a new version of the algorithm which will better match users' behavior and expectations.

7. ACKNOWLEDGMENTS

This research was supported by EPFL and the Swiss National Science Foundation. We thank Elsa Friscira and Laurentiu Dascalu for their important contribution to the development of GroupFun and GroupFun users for their answers and involvement in the evaluation of our algorithms.

8. REFERENCES

[1] Hastie, R. and Kameda, T. 2005. The robust beauty of majority rules in group decisions. In *Psychological Review*, Vol. 112, no. 2, 494-508.

[2] Masthoff, J. 2004. Group modeling: Selecting a sequence of television items to suit a group of viewers. *User Modeling and User-Adapted Interaction*, Vol. 14, no. 1, 37-85.

[3] Masthoff, J. 2005. The pursuit of satisfaction: affective state in group recommender systems. In Computer Science, Vol. 3538, 297-306.

[4] McCarthy J.F. and Anagnost, T.D. 1998. MusicFX: an arbiter of group preferences for computer supported collaborative workouts. *Proc. ACM Computer Supported Cooperative Work*, 363-372.

[5] O'Connor, M., Cosley, D., Konstan, J.A., Riedl, J. 2001. PolyLens: A recommender system for groups of users. *Proc. ACM European Conference on Computer Supported Cooperative Work*.

[6] Popescu G. and Pu P. 2010. Group Recommender Systems as a Voting Problem. *EPFL Technical report*.

[7] Pu P. and Chen L. 2010. A User-Centric Evaluation Framework of Recommender Systems. In the *3rd ACM Conference on Recommender Systems*, Barcelona, Spain

Inferring the meaning of chord sequences via lyrics

Tom O'Hara
Computer Science Department
Texas State University
San Marcos, TX
to17@txstate.edu

ABSTRACT

This paper discusses how meanings associated with chord sequences can be inferred from word associations based on lyrics. The approach works by analyzing in-line chord annotations of lyrics to maintain co-occurrence statistics for chords and lyrics. This is analogous to the way parallel corpora are analyzed in order to infer translation lexicons. The result can benefit musical discovery systems by modeling how the chord structure complements the lyrics.

Categories and Subject Descriptors

H.5.5 [**Sound and Music Computing**]: Modeling

General Terms

Experimentation

Keywords

Music information retrieval, Natural language processing

1. INTRODUCTION

A key task for music recommendation systems is to determine whether an arbitrary song might match the mood of the listener. An approach commonly used is for a system to learn a classification model based on tagged data (i.e., supervised classification). For example, training data might be prepared by collecting a large variety of songs and then asking users to assign one or more mood categories to each song. Based on these annotations, a model can be developed to assign the most likely mood type for a song, given features derived from the audio and lyrics.

Such an approach works well for capturing the mood or other meaning aspects of entire songs, but it is less suitable for capturing similar aspects for segments of songs. The main problem is that human annotations are generally only done for entire songs. However, for complex songs this might lead to improper associations being learned (e.g., a sad introduction being tagged upbeat in a song that is otherwise upbeat). Although it would be possible for segments to be annotated as well, it would not be feasible. There would simply be too many segments to annotate. Furthermore, as the segments get smaller, the annotations would become more subjective (i.e., less consistent). However, by using lyrics in place of tagged data, learning could indeed be done at the song segment level.

Parallel text corpora were developed primarily to serve multilingual populations but have proved invaluable for inducing lexicons for machine translation [6]. Similarly, a type of resource intended for musicians can be exploited to associate meaning with music. Guitarists learning new songs often rely upon tablature notation ("tabs") provided by others to show the finger placement for a song measure by measure. Tabs often include lyrics, enabling note sequences to be associated with words. They also might indicate chords as an aid to learning the sequence (as is often done in scores for folk songs). In some cases, the chord annotations for lyrics are sufficient for playing certain songs, such as those with accompaniment provided primarily by guitar strumming.

There are several web sites with large collections of tabs and chord annotations for songs (e.g., about 250,000 via www.chordie.com). These build upon earlier Usenet-based guitar forums (e.g., alt.guitar.tabs). Such repositories provide a practical means to implement unsupervised learning of the meaning of chord sequences from lyrics. As these resources are willingly maintained by thousands of guitarists and other musicians, a system based on them can be readily kept current. This paper discusses how such resources can be utilized for associating meaning with chords.

2. BACKGROUND

There has been a variety of work in music information retrieval on learning the meaning of music. Most approaches have used supervised classification in which user tags serve as ground truth for machine learning algorithms. A few have inferred the labels based on existing resources. The approaches differ mainly on the types of features used. Whitman and Ellis [10] combine audio features based on signal processing with features based on significant terms extracted from reviews for the album in question, thus an unsupervised approach relying only upon metadata about songs (e.g., author and title). Turnbull et al. [9] use similar types of audio features, but they incorporate tagged data describing the song in terms of genre, instrumentality, mood, and other attributes. Hu et al. [2] combine word-level lyrics and audio features, using tags derived from social media, filtered based on degree of affect, and then revised by humans (i.e., partly

WOMRAD 2011 2nd Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2011 (Chicago, US)

1. Obtain large collection of lyrics with chord annotations
2. Extract lyrics proper with annotations from dataset
3. *Optional:* Map lyrics from words to meaning categories
   (a) Get tagged data on meaning categories for lyrics
   (b) Preprocess lyrics and untagged chord annotations
   (c) Train to categorize over words and hypernyms
   (d) Classify each lyric line from chord annotations
4. Fill contingency table with chord(s)/token associations
5. Determine significant chord(s)/token associations.

Figure 1: Process in learning meanings for chord sequences. The meaning *token* is either an individual word or a meaning category label; and, *chord(s)* can be a single chord or a four-chord sequence.

supervised). McKay at al. [5] combine class-level lyric features (e.g., part of speech frequencies and readability level) with ones extracted from user tags from social media (specifically Last.fm[1]) as well as with features derived from general term co-occurrence via web searches for the task of genre classification.

Parallel corpora are vital for machine translation. Fung and Church [1] induce translation lexicons by tabulating co-occurrence statistics over fixed-size blocks, from which contingency tables are produced to derive mutual information statistics. Melamed [6] improves upon similar approaches by using a heuristic to avoid redundant links.

3. PROCESS

The overall task of processing is as follows: starting with a large collection of lyrics with chord annotations, infer meaning category labels for the chord sequences that occur, based on word associations for the chords sequences. Several steps are required to achieve this in order to make the lyrics more tractable for processing and due to the option for including a lyrics classifier as a refinement of the main induction step. The latter allows meaning to be in terms of high-level mood categories rather than just words.

Figure 1 lists the steps involved. First the Internet is checked to find and download a large sample of lyrics with word annotations. The resulting data then is passed through a filter to remove extraneous text associated with the lyrics (e.g., transcriber notes). Next, there is an optional step to convert the lyrics into meaning categories (e.g., mood labels). This requires a separate set of lyrics that have been tagged with the corresponding labels. Annotations provided by UCSD's Computer Audition Laboratory[2] are used for this purpose, specifically the CAL500 data set [9]. The mapping process uses text categorization with word features and also semantic categories in the form of WordNet ancestors [7]. Prior to categorization, both the CAL500 training data and Usenet testing data are preprocessed to isolate punctuation. However, no stemming is done (for simplicity). The remaining steps are always done. The second-last step computes contingency tables for the co-occurrence of chords and target tokens. Then these are used in the final step to derive co-occurrence statistics, such as mutual information.

[1]See http://www.last.fm.
[2]See http://cosmal.ucsd.edu/cal/projects/AnnRet.

```
[C] They're gonna put me in the [F] movies
[C] They're gonna make a big star out of [G] me
We'll [C] make a film about a man that's sad
    and [F] lonely
And [G7] all I have to do is act [C] naturally
```

Figure 2: Chord annotation sample. Lyrics are from "Act Naturally" by Johnny Russell and Voni Morrison, with chord annotations for song as recorded by Buck Owens.

```
C   They're  gonna  put  me  in  the
F   movies  <endl>
C   They're  gonna  make  a  big  star  out  of
G   me  <endl>  We'll
C   make  a  film  about  a  man  that's  sad  and
F   lonely  <endl>  And
G7  all  I  have  to  do  is  act
C   naturally  <endl> <endp>
```

Figure 3: Sample chord annotations extracted from lyrics. Each chord instance in figure 2 has a separate line.

3.1 Lyric Chord Annotation Data

The most critical resource required is a large set of lyrics with chord annotation. These annotations are often specified in-line with lyrics using brackets to indicate when a new chord occurs. Figure 2 shows an example. The Usenet group *alt.guitar.tab* is used to obtain the data. This is done by issuing a query for "CRD", which is the name for this type of chord annotation. The result is 8,000+ hits, each of which is then downloaded. The chord annotation data is used as is (e.g., without normalization into key of C).

After the chord-annotated lyrics are downloaded, post-processing is needed to ensure that user commentary and other additional material are not included. This is based on a series of regular expressions. The lyrics are all converted into a format more amenable for computing the co-occurrence statistics, namely a tab-separated format with the current chord name along with words from the lyrics for which the chord applies. There will be a separate line for each chord change in the song. Figure 3 illustrates this format. This shows that special tokens are also included to indicate the end of the line and paragraph (i.e., verse).

3.2 Optional Mapping via Lyric Classifier

Rather than just using the words from lyrics as the meaning content, it is often better to use terms typically associated with songs and musical phrases. This would eliminate idiosyncratic associations between chords and words that just happen to occur in lyrics for certain types of songs. More importantly, it allows for better integration with music recommendation systems, such as by using the music labels employed by the latter.

A separate dataset of lyrics is used for lyric classification. Although the overall process is unsupervised, it incorporates a mapping from words to categories based on supervised lyric classification. The source of the tagged data

CAL500 Emotion Categories

Label	f	Label	f
Angry-Aggressive	31	Laid-back-Mellow	7
Arousing-Awakening	77	Light-Playful	1
Bizarre-Weird	7	Loving-Romantic	1
Calming-Soothing	91	Pleasant-Comfortable	3
Carefree-Lighthearted	28	Positive-Optimistic	0
Cheerful-Festive	9	Powerful-Strong	3
Emotional-Passionate	23	Sad	3
Exciting-Thrilling	2	Tender-Soft	2
Happy	6		

Table 1: Frequency of categories from CAL500 used during classification. This reflects the frequency (*f*) of the categories for which lyrics were obtained. Only one category was applied per song, using first tag above a given threshold.

```
movie#1, film#1, picture#6, moving picture#1, ...
  => show#3
       => social event#1
            => event#1
                 => ...
       => product#2, production#3
            => creation#2
                 => artifact#1, artefact#1
                      => whole#2, unit#6
                           => ...
```

Figure 4: WordNet hypernyms for 'movie'. This is based on version 2.1 of WordNet. The first entry omits four variants in the synonyms set (e.g., flick#3), and each branch omits three levels of ancestors (e.g., entity#1).

is CAL500 [9], which uses 135 distinct categories. Several of these are too specialized to be suitable for music categorization based on general meaning, such as those related to specific instruments or vocal characterization. Others are usage related and highly subjective (e.g., music for driving). Therefore, the categorization is based only on the emotion categories. Table 1 shows the categories labels used here. Although relatively small, CAL500 has the advantage of being much more reliable than tags derived from social media like Last.fm. For instance, CAL500 uses a voting scheme to filter tags with little agreement among the annotators.

Out of the 500 songs annotated in CAL500, only 300 are currently used due to problems resolving the proper naming convention for artist and song in Lyric Wiki[3]. In addition, CAL500 provides multiple annotations per file, but for simplicity only a single annotation is used here. The resulting frequencies for the categories are shown in table 1.

Categorization is performed using CMU's Rainbow [4]. Features are based both on words as well as on semantic classes akin to word senses. WordNet ancestors called "hypernyms" [7] are used to implement this. See figure 4 for an example. The use of these word classes is intended to get around data sparsity issues, especially since the training set is rather small. The idiosyncratic nature of lyrics compared to other types of text collections makes this problem more prominent.

As no part of speech tagging is applied as well as no sense

[3]See http://lyrics.wikia.com.

Contingency Table Cells

X \ Y	+	-
+	XY	$X\neg Y$
-	$\neg XY$	$\neg X\neg Y$

G versus 'film'

	+	-
+	1	2,213
-	0	17,522

Table 2: Contingency tables. The left shows the general case, and the right shows the data for chord G and 'film'.

tagging, the hypernyms are retrieved for all parts of speech and all senses. For example, for 'film', seven distinct senses would be used: five for the noun and two for the verb. In all, 43 distinct tokens would be introduced. Naturally, this introduces much noise, so TF/IDF filtering is used to select those hypernyms that tend to only occur with specific categories. (See [3] for other work using hypernyms in text categorization.)

Each line of the extracted chord annotations file (e.g., figure 3) is categorized as a mini-document, and the highest-ranking category label is used or N/A if none applicable. To allow for more context, all of the words from the verse for the line are included in the mini-document. The final result is a revised chord annotation file with one chord name and one category per line (e.g., figure 3 modified to have Light-Playful throughout on the right-hand side).

3.3 Chord Sequence Token Co-occurrence

Given the chord annotations involving either words or meaning categories, the next stage is to compute the co-occurrence statistics. This first tabulates the contingency table entry for each pair of chord and target token, as illustrated in table 2. (Alternatively, chord sequences can be of length four, as discussed later. These are tabulated using a sliding window over the chord annotations, as in n-gram analysis.) This table shows that the chord G co-occurred with the word 'film' once, out of the 2,213 instances for G. The word itself only had one occurrence, and there were 17,522 instances where neither occurred. Next, the *average mutual information* co-occurrence metric is derived as follows:

$$\sum_x \sum_y P(X=x,Y=y) \times log_2 \frac{P(X=x,Y=y)}{P(X=x) \times P(Y=y)}$$

4. ANALYSIS

At the very least, the system should be able to capture broad generalizations regarding chords. For example, in Western music, major chords are typically considered bright and happy, whereas the minor chords are typically considered somber and sad.[4] Table 3 suggests that the chord meaning induction process indeed does capture this generalization. By examining the frequency of the pairs, it can be seen that most cases shown fall under the major-as-happy versus minor-as-sad dichotomy. There are a few low-frequency exceptions, presumably since songs that are sad do not just restrict themselves to minor chords, as that might be too dissonant.

The exceptions shown in the table might also be due to the conventions of chord theory. In particular, chord progressions for a specific key should just contain chords based

[4]Strictly speaking, it is the difference in major versus minor key, but there is a close relation between keys and chords.[8]

avMI	Chord	Word	XY	$X\neg Y$	$\neg XY$
.00034	C	happy	7	1,923	13
.00005	G	happy	4	2,210	16
.00030	Dm	happy	3	341	17
.00008	Em	happy	2	548	18
.00176	F	bright	10	971	3
.00018	Am	bright	3	962	10
.00071	Bm	sad	3	197	4
.00032	Bb	sad	2	325	5
.00039	Em	sad	3	1,097	6
.00542	Dm	sorrow	2	342	5
.00068	C	sorrow	2	1,928	5

Table 3: Sample major versus minor chord associations. Within each group, the entries are sorted by joint frequency (XY). The $\neg X \neg Y$ frequency is omitted (around 17,500), along with a few singleton occurrences.

on the following formula, given the notes from the corresponding major scale:[8]

$$Maj(or), Min(or), Min, Maj, Maj, Min, Diminished$$

Therefore, for the key of C, proper chord sequences only contain the following chords:

$$C, Dm, Em, F, G, Am, Bm, Cdim$$

Likewise, the following are for the key of G:

$$G, Am, Bm, C, D, Em, Fm, F^{\sharp}dim$$

For example, both Dm and Em are among the preferred chords for the key of C major (hence reasonable for 'happy').

Of course, individual chords are limited in the meaning they can convey, given that there are relatively few that are used in practice, compared to the thousands of playable chords that are possible. For example, only 60 chords account for 90% of the occurrences in the sample from Usenet (from a total about 400 distinct chords). Therefore, the ultimate test is on how well chord sequences are being treated.

For simplicity, chord sequences were limited to length four. This was chosen given the correspondence to the number of quarter-note beats in a common time measure (i.e., 4/4 time). Over 4,000 distinct 4-chord sequences were found. As 2,500 of these account for 90% of the occurrences, there is much wider variety of usage than for individual chords.

Running the co-occurrence analysis over words runs into data sparsity issues, so instead results are shown over the mood categories inferred from the CAL500 tagged data. Table 4 shows the top sequences for which a semantic label has been inferred by the classifier (i.e., without guessing based on prior probability). For the most part, the meaning assignment seems reasonable, adding more support that the process described here can capture the meaning associated with chord sequences.

5. CONCLUSION

This paper has presented preliminary research illustrating that it is feasible to learn the meaning of chord sequences from lyrics annotated with chords. Thus, a large, untapped resource can now be exploited for use in music recommendation systems. An immediate area for future work is the

avMI	Chord Sequence	Category	XY	$X\neg Y$	$\neg XY$
.0027	$D7, D7, D7, D7$	Bizarre	30	36	1,358
.0037	$Em, G, G6, Em$	Carefree	18	6	594
.0032	$D, A, A, C^{\sharp}min$	Carefree	14	2	598
.0032	$C^{\sharp}min, D, A, A$	Carefree	14	2	598
.0032	$A, C^{\sharp}min, D, A$	Carefree	14	2	598
.0032	$A, A, C^{\sharp}min, D$	Carefree	14	2	598
.0012	$D7, G, C, G$	Bizarre	14	17	1,374
.0018	$C, D7, G, C$	Bizarre	14	19	1,374
.0022	D, A, A, D	Powerful	13	8	667
.0014	C, D, C, D	Happy	13	39	502

Table 4: Most frequent chord sequence associations. The entries are sorted by joint frequency (XY), and the $\neg X \neg Y$ frequency is omitted (around 18,700). The category names are shortened from table 1.

incorporation of objective measures for evaluation, which is complicated given that the interpretation of chord sequences can be highly subjective. Future work will also look into additional aspects of music as features for modeling meaning (e.g., tempo and note sequences). Lastly, as this approach could be used to suggest chord sequences that convey moods suitable for a particular set of lyrics, work will investigate its use as a songwriting aid.

6. ACKNOWLEDGMENTS

Dan Ponsford provided valuable feedback on the overall process, and Per Egil Kummervold offered useful suggestions. Douglas Turnbull granted access to CAL500, and Cory McKay provided scripts for downloading lyrics.

7. REFERENCES

[1] P. Fung and K. W. Church. K-vec: A new approach for aligning parallel texts. In *Proc. COLING*, 1994.

[2] X. Hu, J. S. Downie, and A. F. Ehman. Lyric text mining in music mood classification. In *Proc. ISMIR*, pages 411–6, 2009.

[3] T. Mansuy and R. Hilderman. Evaluating WordNet features in text classification models. In *Proc. FLAIRS*, 2006.

[4] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. www.cs.cmu.edu/∼mccallum/bow, 1996.

[5] C. McKay et al. Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features. In *Proc. ISMIR*, 2010.

[6] I. D. Melamed. Models of translational equivalence among words. *Computational Linguistics*, 26(2):221–49, 2000.

[7] G. Miller. Special issue on WordNet. *International Journal of Lexicography*, 3(4), 1990.

[8] C. Schmidt-Jones and R. Jones, editors. *Understanding Basic Music Theory*. Connexions, 2007. http://cnx.org/content/col10363/latest.

[9] D. Turnbull et al. Semantic annotation and retrieval of music and sound effects. *IEEE TASLP*, 16 (2), 2008.

[10] B. Whitman and D. Ellis. Automatic record reviews. In *Proc. ISMIR*, 2004.