HYBRID MUSIC RECOMMENDATION BASED ON DIFFERENT DIMENSIONS OF AUDIO CONTENT AND AN ENTROPY MEASURE

Zehra Çataltepe and Berna Altinel

Istanbul Technical University, Computer Engineering Dept, Maslak, Istanbul, Turkey

ABSTRACT

Our music recommendation system recommends a song to a user, at a certain time, based on the listening history of the user. Based on different sets of audio features (MFCC, MPITCH, BEAT, STFT) of all available songs, different clusterings of songs are obtained. Users are given recommendations from one of these clusterings. The right clustering for a user is determined based on the Shannon entropy of the distribution of songs the user listened in each clustering. Using this content based recommendation scheme, as opposed to a static set of features resulted in upto 60 percent increase in recommendation success.

In addition to the audio features (content) of songs user listened, the singers for the songs and also the most popular songs at the time of recommendation are also available. We introduce two recommendation algorithms that decide on the weight of content cluster, singer cluster and popularity adaptively for each user, based on the user history. Our experiments on user session data consisting of 2000 to 500 sessions and of length 5 to 15 indicate that these adaptive recommendation schemes give better recommendation results than using only content based recommendation.

1. INTRODUCTION

Widespread use of mp3 players and cell-phones and availability of music on these devices according to user demands increased the need for more accurate Music Information Retrieval (MIR) Systems. Music recommendation is one of the subtasks of MIR Systems and it involves finding music that suits a personal taste [1]. Audioscrobbler[1], iRate[2], MusicStrands[3], and inDiscover[4] are some of the music recommendation systems today [2]. Usually music recommendation systems follow a collaborative filtering or a content-based approach. Collaborative filtering is the approach used in Amazon [3], a new item is rated by some users and the item is recommended to other users based on the rating of the previous users [4,5]. The disadvantage of the collaborative approach is that when a new item arrives, it has to be rated by someone in order to be used for the other users. In the content-based approach, based on some form of distance between the items already rated by the user and a new item, the item is recommended or not [2, 6, 7, 8]. In order to compute similarities between music pieces different approaches have been suggested. In this paper, we use low level musical features extracted from the audio signals. In the past, two studies [9,10] have also considered collaborative and content based methods for music recommendation. In [9] a Bayesian network is used to include both rating and content data for the recommendation and the hybrid approach is shown to produce better recommendations than using collaborative or content-based approach alone. [10] also use a hybrid approach, where they evaluate CB (Content Based), COL (Collaborative Filtering) and STA (Statistical) methods and their combinations. We base some of our work on that of [10] and give more information about this work below.

In this paper, we also use hybrid approaches to music recommendation. Apart from previous studies, we use the observation that a person may base his/her choice for a song on certain aspects of the song, such as its rhythm or melody. We consider audio features of a song according to four different sets (MFCC, MPITCH, BEAT, STFT) obtained using the Marsyas software [opihi.cs.uvic.ca/marsyas] of Tzanetakis [14]. We do our recommendation computations based on the set of features that clusters the songs a user has listened before as compact as possible, according to an entropy criterion. When we recommend a certain number of songs, M, to the user at a certain time, we recommend a certain percentage of songs based on the content of the songs the user has listened to so far and the song and the user clustering in which the user is. We recommend the remaining songs based on the popular songs at the time of the recommendation and the user's past history. Unlike [10], instead of system-wide weights, we use adaptive weights for each user based on the user's listening history.

The rest of the paper is organized as follows. In Sections 2 and 3, we introduce the data set we used and the features we extracted from songs so that we can measure similarities between them. Section 4 contains descriptions of our recommendation systems and the recommendation success we obtained using each of them. Section 5 concludes the paper.

2. EXPERIMENT DATASET

The user session data is the most important component of a recommendation system. Although there have been recent attempts to produce publicly accessible audio databases [11], we are not aware of a publicly accessible music recommendation database that contains considerable amount of users, sessions and songs.

In the system we consider, cell-phone users request a song and people who call them hear the song they choose instead

[1] www.audioscrobbler.com
[2] irate.sourceforge.net
[3] www.musicstrands.com
[4] www.indiscover.net

of the regular ringback tone. We are provided with the identity of the songs and the times they are selected for each user. Although demographic and personality factors (such as age, origin, occupation, socio-economic background, personality factors, gender, musical education) have been shown to affect music preference [12,13], we did not have access to any user specific information in our dataset. There were sessions of variable length, however we concentrated only on sessions of length 5, 10 and 15 songs. There were a total of 11398, 1215 and 518 user sessions of length 5, 10 and 15. Due to time limitations, we used 2000, 1000 and 500 of these sessions respectively. There were a total of 730 songs whose audio features we obtained as described below.

The purpose of our system is to recommend a song to a user at a certain time. We assume that the user has selected some songs before. We use the songs the user listened to before, the groupings of users according to certain song features and the time of the request to recommend M songs. We obtain clusterings of songs according to different audio features. In order to get advantage of songs selected by users with a history of similar songs, we also obtain clusterings of users according to the songs they have listened to so far. We refer to these two different kinds of clusterings as *song-cluster* and *user-cluster* of a song below.

3. AUDIO FEATURES

Several feature extraction methods including low-level parameters such as zero-crossing rate, signal bandwidth, spectral centroid, root mean square level, band energy ratio, delta spectrum, psychoacoustic features, MFCC and Auditory filterbank temporal envelopes have been employed for audio classification [12]. In our experiments we have obtained the following content based audio features using Tzanetakis's Marsyas software, with default parameter settings.

3.1. Timbral Features

Timbral features are generally used for music-speech discrimination and speech recognition. They differentiate mixture of sounds with the same or similar rhythmic content. In order to extract the timbral features, audio signal is divided into small intervals that can be acceptable as stationary signal. The following timbral features are calculated for these small intervals: Spectral Centroid, Spectral Rolloff, Spectral Flux, Time Domain Zero Crossing, Low Energy, Mel-Frequency Cepstral Coefficients (MFCC).

Means and variances of the spectral centroid, spectral rolloff, spectral flux, zero crossing (8 features) and low energy (1 feature) results in 9 dimensional feature vector and represented in experimental results as STFT label [14]. Means and variances of the first five MFCC coefficients yield a 10 dimensional feature vector, which is represented as MFCC in the experiments.

3.2. Rhythmic Content Features

Rhythmic content features characterize the movement of music signals over time and contain such information as the regularity of the rhythm, the beat, the tempo, and the time signature [14,15]. The feature set for representing rhythm structure is based on detecting the most salient periodicities of the signal. Rhythmic content features are calculated by beat histogram calculation and yield a 6 dimensional feature vector which is represented using BEAT label.

3.3. Pitch Content Features

The melody and harmony information about the music signal is obtained by pitch detection techniques. Although musical genres by no means can be characterized fully by their pitch content, there are certain tendencies that can lead to useful feature vectors [14]. Pitch content features are calculated by pitch histogram calculation and yield a 5 dimensional feature vector which is represented as MPITCH in the experimental results.

The following is a list of audio features we use and their length:

- BEAT (6 features)
- STFT (9 features)
- MFCC (10 features)
- MPITCH (5 features)
- All (30 features)

4. MUSIC RECOMMENDATION

Let $s(i) = [s(i,1), s(i,2), ..., s(i, N_i)]$ represent the i'th user session containing N_i songs. $s(i, j)$ represents the j'th song of the i'th session. $t(i) = [t(i,1), t(i,2), ..., t(i, N_i)]$ is the vector of the times (measured in terms of the number of seconds since Jan 1, 1970) at which the songs in session *s(i)* were chosen. The recommendation task that we consider is the following: Given the portion of a session excluding the last song, $s(i)^- = s(i,1), s(i,2), ..., s(i, N_{i-1})$, recommend N=20 songs from among 730 songs at time $t(i, N_i)$.

Each song $s(i, j)$ is represented by means of the 30 dimensional audio feature vector, $x_{i,j} \in R^{30}$ consisting of

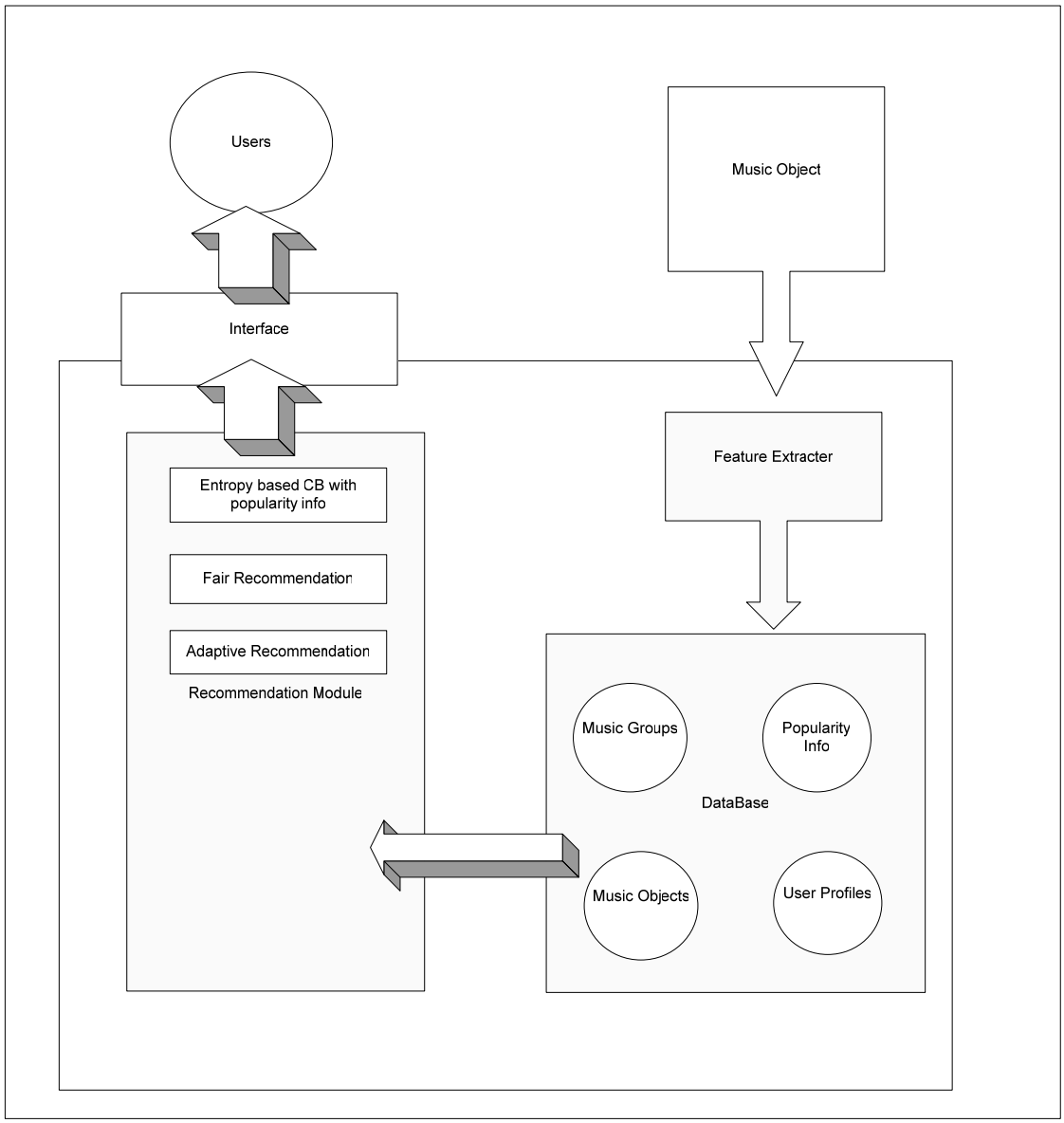

Figure 1. The components of our music recommendation system.

MFCC, MPITCH, STFT and BEAT features described above**.** First, using CLUTO [16] software and graph clustering option, we obtain 8 different clusterings of all the 730 songs in our database. The 8 clusterings are obtained using different combinations of MFCC, MPITCH, STFT, BEAT, MFCC and MPITCH, STFT and BEAT, MPITCH and BEAT features. We considered all 15 possible feature set combinations, but discarded combinations for which the clustering algorithm can not perform well (i.e. very non-homogenous clusters, many songs outside clusters, etc.).

Since we compare our work to that of [10], we give more details about their work here. In CB approach of [10] , first all the songs are clustered, then each cluster is given a weight based on whether a song the user listened before is in the cluster or not. The number of songs recommended from each cluster is chosen proportional to the weight of the cluster. The disadvantage of the CB based approach is the fact that the user is recommended songs only from the clusters he has listened to before. In COL approach, not only the clusters which have contributed to the songs the user listened to, but also clusters that contributed to other users are taken into account. Of course there could be clusters that contain songs not listened enough by anybody and those will be ignored. In STA approach, all the songs are divided into two groups, short term and long term. A certain number of songs is selected from the long term list and the remaining ones are selected from the short term list. STA behaves similar to the popularity in recommendation systems. Since [10] found out that CB was the least successful among the methods he experimented with, we concentrated on COL and STA. We implemented the COL approach as described in [10] and for STA, we used the time frame immediately t days before the time of the recommendation. We think this makes STA take better advantage of popular songs around

the time of the recommendation. Although [10] recommends using 50% from among the popular songs and 50% from among the others, we experiment with different ratios.

Our recommendation algorithms use three different measurements on a song:

Cluster similarity: Similarity of the song to the 8 clusterings of all songs that we produce using the CLUTO software. Similarity of a song to a cluster is measured according to the Euclidean distance between the song and the cluster centroid.

Singer similarity: Singer similarity value is calculated according to the 4 level hierarchy presented to the cellphone users: Turkish/Foreign song, genre and singer. If two songs share the same singer then their singer similarity is 4, if they do not share the same highest category, their singer similarity is 0.

Popularity: For any day of recommendation, we group songs into popular and non-popular. We compute the popularity ratio as the number of times a song is listened within the last t days divided by the number of times all songs are listened within the last t days. We compute the mean popularity ratio for all songs and group songs whose popularity ratio are below the average as unpopular and the rest as popular.

A recommendation consists of $N = N_c + N_s + N_p$ songs, where N_c, N_s, N_p represent the number of songs recommended according to cluster, singer and popularity components respectively. These songs are chosen according to the measurements made on songs $s(i, j), j < N_i$ that the user selected before he was about to make his selection at time $t(i, N_i)$.

In the following three sections we describe three different recommendation algorithms which differ in the way they choose N_c, N_s, N_p and we also present the experimental results for each algorithm. Figure 1. shows an overview of the recommendation system:

4.1. Content Based Recommendation Using Entropy of Clusterings:

This method based on the [10] content based recommendation algorithm. All songs are clustered via Cluto. Then observing the user history (i.e. songs $s(i, j), j < N_i$) we decide which among the 8 clusterings should be used for the user. In order to decide which feature set is the best for the user we use an entropy based measure. For each of the 8 clusterings we compute an entropy value as follows: we find the closest cluster centroid and assign song *s(i,j)* to that cluster. Let $p_k = n_k / (N_i - 1)$, where n_k is the number of songs *s(i,j)* assigned to the k'th cluster, $k \leq K$. The (Shannon) entropy value for this clustering is computed as the sum of $-p_k \log p_k$ for *k=1..K*. The clustering whose entropy is minimum is selected as the clustering to which the user belongs, because it is the clustering that can group the songs user has listened to in the best possible way. After we determine the clustering, we choose songs from each cluster in the clustering, proportional to the number of *s(i,j)* that belonged to the cluster. The songs are selected so that their average similarity to *s(i,j)* is maximum. As explained before and in [10], the disadvantage of this approach is the fact that we recommend songs that may be too close to the songs that the user has listened before.

In order to get advantage of the popular songs, we recommend a certain portion of the songs using this method and we fill up the remaining songs based on the popular songs at the time of the recommendation.

Table 1. shows the success of recommendation for varying ratio of recommendations from the popular songs. A recommendation is considered successful if the N_i'th song is among the recommended songs. 20 songs are recommended in all cases. The number of sessions considered are 2000, 1000 and 500 for sessions of length 15, 10 and 5 respectively. Percentage of songs recommended from among the popular songs at the time of recommendation are shown on the second column, and as more popular songs are recommended recommendation success increases. The remaining songs are recommended using content based method. Column 3 in the table shows the recommendation success when the entropy of clusterings of songs in user history are used to select the best clustering for the user among 8 different clusterings. Columns 4, 5 and 6 shows the recommendation success when only a static set of features and hence clustering (ALL, MPITCH+MFCC, BEAT+STFT) are used for clustering. The last column shows the percentage difference between the entropy based recommendation and the best of the static recommendation methods. The entropy based recommendation results in 10 to 62 percent better recommendation success. The entropy based measure results in better improvement as the session length increases, because feature set and hence the clustering valued by the user in selecting a song can be predicted more reliably. When CB recommendation is done based on only a static set of features, using ALL features results in better recommendation success.

Session Length	%Popular Recommended	%RecomSuccess Adptv Features	%RecomSuccess All Features	%RecomSuccess MFCC+MPTCH	%RecomSuccess STFT+BEAT	%Adaptive features better than best static
5	20	21	19	11	11	10
5	40	30	22	16	13	36
5	60	40	28	14	17	42
5	80	44	33	22	19	33
10	20	22	18	13	13	22
10	40	32	25	16	18	28
10	60	41	27	13	13	51
10	80	46	29	20	17	58
15	20	22	17	8	11	29
15	40	33	21	16	13	57
15	60	44	27	15	15	62
15	80	50	32	17	17	56

Table 1. Recommendation success when 8 clusterings and entropy measure vs. a static single clustering is used.

4.2. Fair Recommendation: In this method we use all three components (cluster similarity, singer similarity and the popularity) and learn the percentage values for each component. We do the learning as follows: For each

$s(i, j), j < N_i$, we try to find *s(i,j)* based on all remaining $N_i - 2$ songs in the session. We use cluster, singer and popularity components all by themselves and we increment the weight of a component if it finds the song *s(i,j)*. We choose N_c, N_s, N_p proportional to these weights.
The results of this recommendation scheme are shown in Table 2 at the 2nd column. As seen in the table, the percentage of success for Fair Recommendation is a lot higher than the Content Based Recommendation.

4.3. Adaptive Recommendation: In this recommendation scheme, we choose N_c, N_s, N_p from among a certain number (250) of different possible values. As we did in the previous recommendation algorithm, we evaluate each N_c, N_s, N_p combination's score based on how well they can predict each $s(i, j), j < N_i$. We choose the combination that gives the best success rate.
The recommendation success ratio for this approach is shown in Table 2, 5th column. The success ratio of adaptive recommendation seems to be smaller than that of Fair Recommendation. We think that this is due to the fact that Fair Recommendation uses a component (like singer for example) and ignores the other two (like content and popularity for example) when it makes its decision. Whereas, Adaptive Recommendation is able to evaluate contributions of all components at the same time. For all the experiments shown, the content based recommendation is made using the 8 clusterings and the entropy based approach described in Section 4.1.

Session Length	%RecomSuccess Fair Recommendation	%RecomSuccess Adaptive Recommendation
5	70	65
10	71	63
15	73	70

Table 2. Recommendation Results for Fair and Adaptive Recommendation.

5. CONCLUSIONS

In this study, we introduced a number of new ideas for music recommendation based on audio features. First of all, we introduced a framework that lets us use different sets (portions) of audio features for each user so that we can do content based recommendation to a user based on the most relevant audio feature dimensions for the user. We used the entropy measure to decide on which feature set to use for a particular user. We also introduced the Fair and Adaptive Recommendation algorithms, that learn the weights to give to content, singer similarity or popularity components, for each user, based on the user history. Among these algorithms, the Fair Recommendation algorithm resulted in the best success rate.

Acknowledgements
We thank Argela for providing the user session data, G. Tzanetakis for Marsyas software and G. Karypis for Cluto software. We thank Prof. Sule Gunduz of Istanbul Technical University for useful discussions.

REFERENCES

[1] R. Typke, F. Wiering, and R. C. Veltkamp, "A Survey of Music Information Retreival Systems," in *Proc. of the International Conference on Music Information Retrieval (ISMIR) 2005.*
[2] O. Celma, M. Ramirez, and P. Herrera, "Foafing the Music: A Music Recommendation System Based on RSS Feeds and User Preferences," in *Proc. of ISMIR 2005.*
[3] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, 4(1), 2003.
[4] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating "Word of Mouth", " *ACM CHI'95 Conference on Human Factors in Computing Systems*, 1995, pp. 210–217.
[5] W. Cohen and W. Fan, "Web-collaborative filtering: Recommending music by crawling the Web, " *Computer Networks*, vol. 33, no. 1–6, pp.685–698, 2000.
[6] K. Hoashi, K. Matsumoto, and N. Inoue, "Personalization of user profiles for content-based music retrieval based on relevance feedback," *ACM Multimedia*, 2003, pp.110–119.
[7] B. Logan, "Music recommendation from song sets, " *Proc. of ISMIR 2004.*
[8] P.K. Cano, et.al., "An Industrial-Strength Content-based Music Recommendation System," in *Proceedings of 28th Annual International ACM SIGIR Conference*, Salvador, Brazil, 2005.
[9] K. Yoshii, *et.al*., "Hybrid Collaborative and Content-based Music Recommendation Using Probabilistic Model with Latent User Preferences," in *Proc. of the International Conference on Music Information Retrieval (ISMIR)*, 2006.
[10] H-C. Chen and A.L.P. Chen, "A music recommendation system based on music and user grouping," *Journal of Intelligent Information Systems*, Volume 24, Numbers 2-3, 113-132, 2005.
[11] C. McKay, D. McEnnis, and I. Fujinaga,
"A Large Publicly Accessible Prototype Audio Database for Music Research," in *Proc. of the International Conference on Music Information Retrieval (ISMIR)*, 2006.
[12] A. Uitdenbogerd and R. van Schyndel, "A review of factors affecting music recommender success," in *Proc. of the International Conference on Music Information Retrieval (ISMIR)*, 2002.
[13] B. Yapriady and A.L. Uitdenbogerd, "Combining Demographic Data with Collaborative Filtering for Automatic Music Recommendation, " *Knowledge-Based Intelligent Information and Engineering Systems, LNCS* Volume 3684/2005.
[14] G. Tzanetakis, and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.
[15] T. Li, and G. Tzanetakis, "Factors in automatic musical genre classification of audio signals," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2003.
[16] G. Karypis, *Cluto A Clustering Toolkit Manual*, University of Minnesota, Department of Computer Science Technical Report, 2002.