Association Rule Mining and Audio Signal Processing for Music Discovery and Recommendation

Md. Mahfuzur Rahman Siddiquee, Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh
Md. Saifur Rahman, Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh
Shahnewaz Ul Islam Chowdhury, Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh
Rashedur M. Rahman, Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh

ABSTRACT

In this research, the authors propose an intelligent system that can recommend songs to user according to his choice. They predict the next song a user might prefer to listen based on their previous listening patterns, currently played songs and similar music based on music data. To calculate music similarity the authors used a Matlab toolbox that considers audio signals. They used association rule mining to find users’ listening patterns and predict the next song the user might prefer. As they propose a music discovery service as well, the authors use the information of music listening pattern and music data similarity to recommend a new song. Later in result section, they replaced the audio based similarity with last.fm api for similar song listing and analyzed the behaviour of their system with the new list of songs.

KEYWORDS
Association Rule Mining, Data Mining, Music Discovery System, Music Recommender System

1. INTRODUCTION

Recommending good music can be a challenging task. The song a user might prefer to listen can be based on numerous factors like mood, location, etc. In our paper, we built a system for personalized music recommendation based on their mood which can be determined from their music listening patterns. In this research, we do not predict explicitly in which mood a user is at the moment, rather we try to predict which song might be liked next based on currently played songs. This requires implicit
analysis of users’ mood. To analyze the users’ mood, we consider the songs which are currently being played by the user. Then we analyze previous playlists those were played by users. We try to figure out which songs the user might prefer to listen. We do this analysis using association rule mining. We also propose a music discovery service as users might also like to listen new music that resembles with their current mood. Therefore, we measure similarity among the music. An audio data file may have many components like frequency, tempo, rhyme, bass, bitrate, etc. But how do we determine which songs are similar? We used fluctuation pattern and MFCCs of audio files for measuring distance among them. Thus, high distance value means less similar, and low distance value means highly similar. After using association rule mining we find which song a user might prefer to listen that are among the currently available songs, and we take the support/confidence value of each song. Now we pick each song from this list and based on the similarity of new songs and the confidence value, we recommend the next song.

We used Matlab and Rapidminer environment for our computations. For data processing, we used Python, Bash and C++. To measure audio similarity, we used a Matlab toolbox created by Elias Pampalk which was introduced in one of his works (Pampalk, 2004) and it has dependency to the open source tool Netlab (Ncrg.aston.ac.uk, 2015). Although the toolbox by Elias Pampalk is not suited for large scale processing, it can serve as a reference implementation and a good medium to study effects of parameters.

2. RELATED WORKS

The authors in (Kim, Lee, Yoon & Lee, 2008) designed a music recommendation system based on personal preference analysis. First they built a music model using Hidden Markov Models (HMM) with Mel Frequency Cepstral Coefficients (MFCC). They calculated HMM for each song, representing one model for each song. They check the similarity between models and calculate and store vectors for similarity. They also use an improved K-Means algorithm which limits the radius of a cluster. For analyzing preference and assign weights to clusters, they set higher weights to clusters which are further away from each other and fewer weights to clusters which are near. Their recommendation system uses these weights to model similarity to recommend music to users. Their system proves to provide very accurate recommendations for some particular genres.

Mandel and Ellis, in their paper (Mandel & Ellis, 2006), showed that using support vector machine (SVM) classifier with song level feature are better than using only SVM classifier. They also showed that use of Kullback-Leibler (KL) divergence for measuring distance is better than Mahalanobis distance over Gaussian model. They used 20 –coefficient MFCCs as suggested by Aucouturier and Pachet (Pachet & Aucouturier, 2004) and a “bag of frames” model for modeling the MFCCs. After extracting MFCCs, the mean and covariance of them are described in a Gaussian model. For distance measurement, they used both Mahalonobis and closed form of KL (Penny, 2001) divergence. They conducted experiments with the combinations of single Gaussian, GMM with KL divergence. They also compared artist-level versus song-level feature and SVM versus non-SVM classifiers. They showed that the use of single Gaussian is an effective way of capturing and comparing song-level feature (MFCC).

The authors in (Aucouturier & Pachet, 2002) used an interesting way of comparing music. They built timbre descriptor for the music and computed them for similarity measure. Timbre can be called as “tone color”. Musical instruments create different frequencies for the same note. Timbre is used to identify these instrumental differences. They chopped the song in 2048 pieces and calculated the MFCCs. As for huge data set, it would not be possible to store all the MFCC values, they used Gaussian Mixed Model (GMM) to store the mean and covariance. They considered two ways of distance measurement between models – Likelihood and Sampling. Likelihood seems to be heavier to compute, so they used sampling. The process was to choose a sample from one GMM and compute the
LOCALE: Collaborative Localization Estimation for Sparse Mobile Sensor Networks
www.igi-global.com/chapter/locale-collaborative-localization-estimation-sparse/66476?camid=4v1a

Katana: Towards Patching as a Runtime Part of the Compiler-Linker-Loader Toolchain
www.igi-global.com/article/katana-towards-patching-runtime-part/46149?camid=4v1a