Similar Song Prediction based on Neural Networks

Deepthi Male Neeharika Kondipati Gayathri Shivakumar
{dmale, vkondipati, gshivakumar}@cs.stonybrook.edu

Abstract

The problem statement is to predict a similar song based on the playlist of a user. Existing approaches for this task are mainly based on Collaborative Filtering and Markov Chains. We proposed a novel way of using neural networks to complete this task. The main inspiration was drawn from the fact that neural networks have been successfully applied to sequential prediction problems such as language model and click through rate prediction.We tried to analyse the song patterns of a user and tried to predict how similar the predicted song is. We could achieve a song similarity of 53% given the playlist of the user.

Introduction

Neural networks have found profound success in the area of pattern recognition.With the explosion of digital music in recent years, the application of pattern recognition technology to digital audio has become increasingly interesting.It has been shown that combining the sequential pattern of song-listening behavior and the general taste of the user is more helpful to system performance than solely utilizing the sequential pattern or the general preference of the user. This might be considered an equivalent to a hybrid of a conventional content-based (CB) recommendation system and a collaborative filtering (CF) recommendation system.However, a CB system only captures the similarity between items but not the sequential listening pattern of a specific user and pure collaborative filtering is content-agnostic : they do not use any kind of information about the songs that are being recommended, except for the listening patterns associated with them.

Also, there is a lot of work done in this domain including rule-based audio classification method, the pattern match method, Hidden Markov Models (HMM), but they have own shortcomings.For example, rule-based audio classification can only be applied to identify the audio genres with simple characteristics, such as mute, so it is very difficult to meet the requirements of complex and multi-feature music. Pattern matching method needs to establish a standard mode for each audio type, so the calculation amount is large and classification accuracy is low.HMM method has poor classification decision ability and needs priori statistical knowledge.

Neural networks, is thus, a fresh approach to this issue where we resorted to the Bag of Words model to predict a similar song that can be recommended to a listener. By repeatedly showing a neural network inputs classified into groups, the network can be trained to discern the criteria used to classify, and it can do so in a generalized manner allowing successful classification of new inputs not used during training. The advent of neural networks helps in creating a model which continuously learns from the listening patterns of a user and tries to figure out a song similar to the listener's trends which might actually interest him/her.

Dataset Collection and Pre-Processing

In our experiments, we have used the following datasets to get the data required for the project.

1. *The Echo Nest Taste Profile Subset*

The user data is provided as (user, song, play count) triplets, and each line looks like this:

b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOAKIMP12A8C130995	1
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOAPDEY12A81C210A9	1
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBBMDR12A8C13253B	2
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBFNSP12AF72A0E22	1
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBFOVM12A58A7D494	5

2. *The musiXmatch Dataset*

The musiXmatch dataset provides a large collection of song lyrics in bag-of-words format, for academic research.

3. *The Last.fm Dataset*

The last.fm dataset provides huge collection of data for song-level tags.

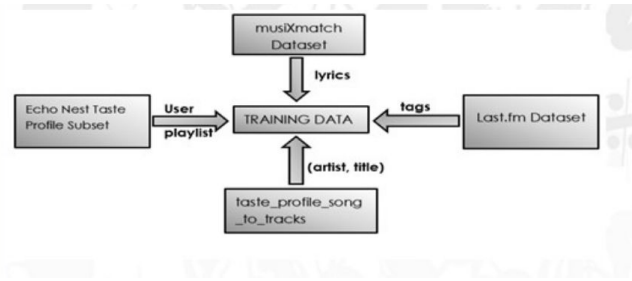
4. *The taste_profile_song_to_tracks.txt*

This file provides the artist and title for a given track. This serves important when converting song to track so as to get the lyrics by trackid.

Pre-Processing :

For each triplet in user data, we got the lyrics of the song from musiXmatch Dataset and the corresponding song tags from Last.fm dataset and generated our training data. The structure of training data is as follows -

<Userid, song_id, artist, title, lyrics, play_count tags>



Data Collection and Pre-Processing

Approach

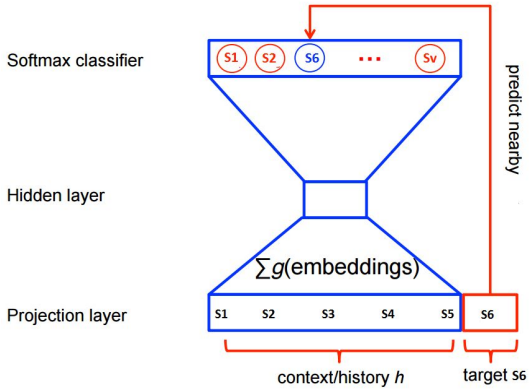
Neural Network Framework

The task is to build a recommendation system to predict a similar song for a user based on the songs previously listened by him/her. Each user has a listening record, which can be expressed as = (s_1, s_2, …) where s_i the index of the song listened by the user. The challenge is to train the predictor model with the song features of the songs in the playlist to predict a song similar to the taste of the user.

We used the word embedding technique to gain low-dimensional feature representation and reduced time/space complexity. The embedded feature has the merit of utilizing the similarity between each other to reduce data sparseness for later processing. It is because once a system has been trained with an embedded feature, it is equivalently trained with similar embedded features.

Formalizing Neural Network - CBOW

CBOW model performs the task of predicting the word given its context. We can extend the model to predict a song based on other songs in the playlist which can be considered as the context. Here each song is a vector of 28 features(explained in next section) and in turn each feature is a word embedding which is learned by the neural network in accordance with the main task of modelling similar songs.



In the above figure input to the neural network is the embeddings of songs a user listens to(S1...S5), S6 is the predicted song embedding in the CBOW model.

Features from Dataset

The data we collected has the following information about the song. We generated vocabulary by parsing through the lyrics and tags of the song. We have split the lyrics and tags into words and represented each song as a vector of word embeddings corresponding to artist, title, lyrics, play_count, tags. Inorder to account for the importance of artist, title and play_count, we have given weights to these features.

Implementation

Lyrical Feature Engineering

We have seen from the previous section on how the song is represented as a vector of word embeddings. We have performed the following enhancements on the lyrical features and observed the similarity score of the predicted songs in each case.

I. BOW Model:

The baseline model is where we considered the song tags and words from the lyrics of the song as features representing the songs. By considering all the words we have increased the model time/space complexity, which is why is not an efficient method.

II. Tokenizing/Stemming

We have reduced the dimensionality of the song features by tokenizing and stemming the lyrics. In this way we have reduced the dimensionality of the song features from 1496 to 1056 for 200 songs.

III. Clustering

Further improvement on the dimensionality reduction was achieved by clustering the vocabulary from the songs. By performing clustering with 500 clusters on the song features, we were able to reduce the dimensionality to 500.

Below is the table which shows the achieved similarity of the predicted song with the above mentioned techniques.

LYRICAL FEATURES	SIMILARITY
Bag of Words	47.352
Tokenize/Stem	44.891
Cluster	53.278

***Table**: Accuracy corresponding to Lyrical Features*

.

CBOW Model Representation

We need to provide the neural network with inputs (context songs) and labels (similar songs) which can be generated by various combinations of songs in user playlist.

We have generated the input-label data by taking the following combinations of songs in the user playlist:

I. Assuming last song similar to all songs in playlist

Consider a particular user playlist as follows-
(S1, S2, S3, S4, S5). We develop the train labels and train inputs in the following way.

Input Song(Context Song)	Similar Songs(Input Labels)
S1	S5
S2	S5
S3	S5
S4	S5

This method fails to give the sufficient data for neural network as we did not consider the similarity between other songs. We considered last song here, however this can be done with any other song in the playlist.

II. Half of the songs similar

Consider a particular user playlist as follows-
(S1, S2, S3, S4). We develop the train labels and train inputs in the following way.

Input Song(Context Song)	Similar Songs(Input Labels)
S1	S3
S1	S4
S2	S3
S2	S4

This happened to be better than previous consideration as we have considered more pairs.

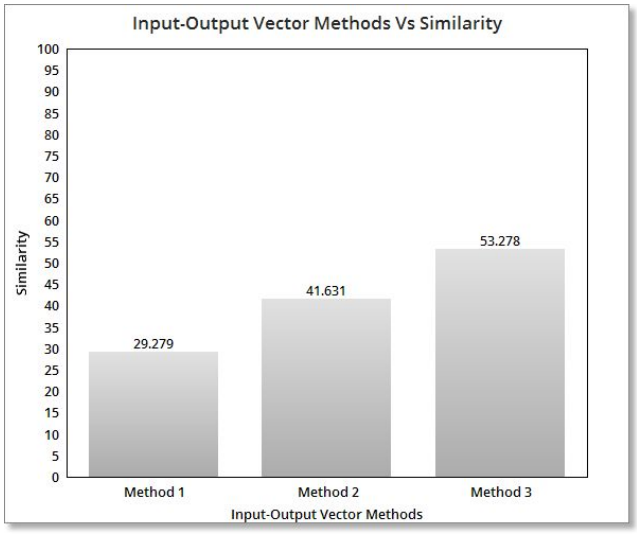
III. All songs similarity

Here, we considered that every song in the user playlist is similar to every other song assuming that user will listen to songs based on his interests and would not like to experiment by listening to new genres each time.
Suppose (S1,S2,S3) are the songs in user playlist. We develop the train labels and train inputs in the following way.

Input Song(Context song)	Similar Songs(Input Labels)
S1 S1	S2 S3
S2 S2	S1 S3
S3 S3	S1 S2

For the above mentioned 3 representations the predicted song similarity for 200 users is plotted in the figure below.

The similarity is calculated using the similarity with user playlist method as described in the next section.



Similar Song Prediction

For all the songs in our training data, we generate/learn song embeddings for each song from the neural network

model based on the playlist. Now, we use these song embeddings to calculate the similarity between every two songs in the database. Given a particular user's playlist, we find a song closer to all the songs in the playlist.

Evaluation

Similarity of Predicted Song

I. Similarity with user Playlist

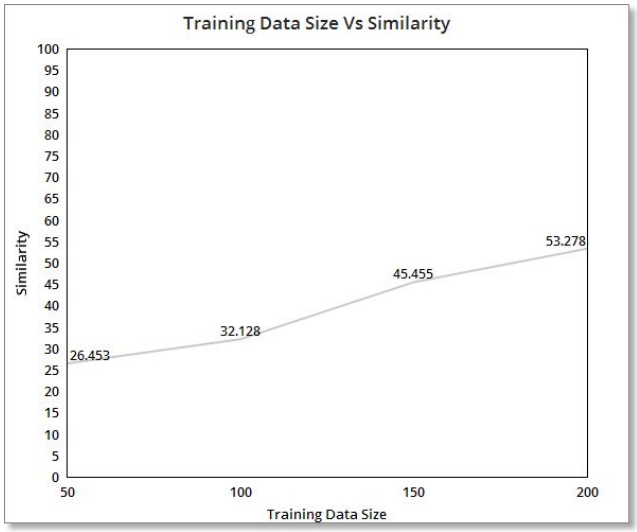
We considered the entire playlist and predicted a similar song to all the songs in the playlist. Then, we found the similarity of the predicted song to all of the songs in the playlist. This evaluation made more sensible as the playlist does not provide any information about the sequence in which the songs are listened. It just gives the set of songs listened.

II. Similarity with left-out song from Playlist

We removed one song(left-out song) from the playlist and predicted the similar song to other songs in the playlist. Then, we found how similar is the predicted song to the left-out song.

Increasing the User-Song data:

Below are the results for the model where we generated training data using all-songs similarity and evaluated the model using similarity with user playlist. We have increased the training data in each experiment and observed the improvement in the prediction similarity.



Future Scope

Certain features like genre are supposed to be represented in our system.. We intend to use weights for the same. Also, the lyrical features considered now do not give a complete picture as we have used bag of words and clustering whereas other syntactic and semantic features can be captured to give a better representation. We plan on integrating this with our existing system in the future.

References

https://arxiv.org/ftp/arxiv/papers/1606/1606.07722.pdf

http://benanne.github.io/2014/08/05/spotify-cnns.html#scalingup

https://labrosa.ee.columbia.edu/millionsong/lastfm

https://developer.spotify.com/spotify-echo-nest-api/

https://www.kaggle.com/c/msdchallenge/data

https://nlp.stanford.edu/courses/cs224n/2006/fp/sadovsky-x1n9-1-224n_final_report.pdf