Content-Based Music Recommendation

Akshay Naik
aunaik@iu.edu
Indiana University

Rishab Nagaraj
risnaga@iu.edu
Indiana University

Vaishnavi Srinivasan
vsriniv@iu.edu
Indiana University

ABSTRACT

This paper describes the creation of a music recommendation system that is based on audio signal data. We chose this problem because the current state of the art method for music recommendation is to use collaborative filtering or recommending songs based on their metadata. This system has been employed by, for example, Spotify, to create tailored playlists for their users. However, this system is not perfect. It has issues like, the cold start problem and the popularity bias problem. The system described in this paper aims to tackle this problem by removing metadata from the equation and giving users recommended playlists created only by using the audio signal data.

KEYWORDS

music recommendation, content based, mfcc, cnn, lstm

1 INTRODUCTION

The tailored playlists generated by multiple streaming platforms over the last few years have some significant drawbacks. Products like Spotify, Google Music, Deezer, Apple Music, etc employ some form of collaborative filtering methods for generating "recommended" playlists based on user history or user similarity. For example, Spotify[3] has built its recommendation system that incorporates a user-based collaborative filtering task, Natural Language Processing task for identifying terms that determine music similarity and a task that analyzes a raw sound signal to identify audio signals features such as time signature, key, mode, tempo, and loudness.

In these cases, the recommendation systems run into two major problems - cold start problem[1] and popularity bias[6]. In the case of the cold start problem any new artists with a minimal discography are almost never recommended. This is partly due to lack of metadata for the artist or their songs in the collaborative filtering model that can be used by the algorithm to perform a recommendation. In the case of popularity bias, the biggest artists with large discographies will always be highly recommended due to predominant user behavior (i.e) any artist that has a popularity vote on say Billboard Hot 100 or Billboard Top Pop songs would be recommended to users more.

In this project we work towards solving this problem by recommending songs purely on the basis of the audio signal data. We first extract each song's spectrum information. This is trained using a Convolutional Neural Network (CNN) and a Long Short Term Memory (LSTM) network to transform the data from high dimensional space (spectrogram) to low dimensional space (genre space). We perform this transformation to avoid the "Curse of dimensionality", wherein more data is required for every combination of feature for the machine learning classifier to make an informed prediction. The low dimensional data representation using genre contains the probability distribution of a song belonging to a class of genres. We had better accuracy on training dataset when compared with testing and validation as the dataset is unbalanced. We added weights to the models based on genre class distribution to counter this effect.

Once the probability distribution of a song is derived, we then employ a cosine similarity distance metric evaluator to build a recommendation playlist. This playlist contains the five songs that have the closest probability distribution to that of the input song. As a pop song can contain elements of rock, and a rock song can contain elements of metal, our recommendation system also uses the probability distribution to identify songs that have similar genres.

2 LITERATURE REVIEW

We researched current state of the art recommendation systems used in industry. We read Sophia Ciocca's[3] article on Spotify's recommendation system and its employment of audio, text and collaborative filtering methods for user specific playlist generation. We found that the system uses quite a few audio features, but doesn't look at any structural information about the song. This article inspired us to incorporate the structural information provided by the song, its MFCC, and the genre that it belongs to for playlist generation.

We referred to the work of Sander Dieleman[4], who used CNN for content-based music recommendation based on the results from collaborative filtering. She trained a CNN to transform the data for each signal to a 40-dimensional latent feature vector. Inspired by this work we decided to train a CNN to transform data from the time-frequency domain to genre space. Having access to the huge data corpus of Spotify as well as the output from the collaborative filtering technique run on the Spotify dataset, Dieleman was able to achieve some good results and we wanted to do something similar with the FMA dataset.

We also referred to the works of Albert Jimenez[5] and Ruoho Ruotsi[7] to understand how CNN and LSTM can be implemented together to understand the overall structure of the song. They used the GTZAN dataset for genre classification. The LSTM network has been implemented to extract the complete structure of the song and its dependencies across short periods of time to predict the probability distribution of the song across different genres.

For generating our recommended playlist, we referenced the work of Sylvester Card[2]. He has also transformed the data from a higher dimensional space to lower dimensional genre space to generate a a playlist. He has used Euclidean distance for similarity prediction. We evaluated our model for various distance metrics and finally decided to use Cosine similarity for song recommendation. The songs generated by using Cosine similarity were more similar to each other and generated a higher mean score of similarity than other metrics.

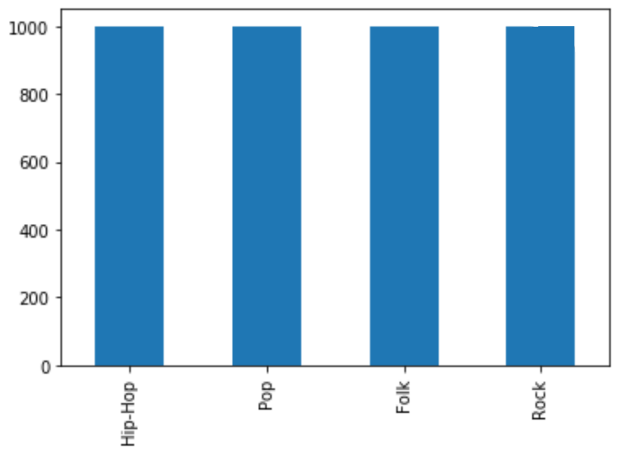

Figure 1: Small dataset Data Distribution

3 DATA

Since we are transforming the song from spectrogram dimensionality to genre dimensionality, we decided to use the data from the Free Music Archive (FMA) which has genre labels for the songs. It has 917 GB creative commons-licensed music and has been extensively used for Music Information Retrieval. We have used two of FMAs data subsets - small and medium.

(1) Small dataset: has approximately 4000 tracks with four balanced genres - Folk, Hip-Hop, Pop and Rock
(2) Medium dataset: has approximately 13500 tracks with eight unbalanced genres - Blues, Classical, Country, Folk, Hip-Hop, Jazz, Pop and Rock

3.1 Data Pre-processing

The songs were converted to a visual spectrum of frequencies that vary with time. These frequencies were extracted in the mel-scale, called Mel-frequency Cepstral Coefficients (MFCC), to help represent a humans perception of various sound frequencies. The spectrograms represent the Short Time Fourier Transforms (STFT) of the sound signal with a window of 2048 long samples (i.e) 10ms song period. The songs are extracted with a fixed sampling rate of 44100 Hz and are cropped at 29.95 seconds to have consistency between all the music signals.

The data distribution of the balanced small dataset is given in Figure 1. The data distribution of the unbalanced medium dataset is given in Figure 2. The dataset has a lot of Rock songs but very few Blues and Country songs. The conversion of the song to STFT and then to MFCC is displayed in Figure 3.

4 EXPERIMENTS

For content-based music recommendation, we decided to find the similarity between songs purely based on their music signal. However, finding similarity at the time domain space or even at the time-frequency domain space is complex due to the large number of dimensions. This also introduces the "Curse of dimensionality".

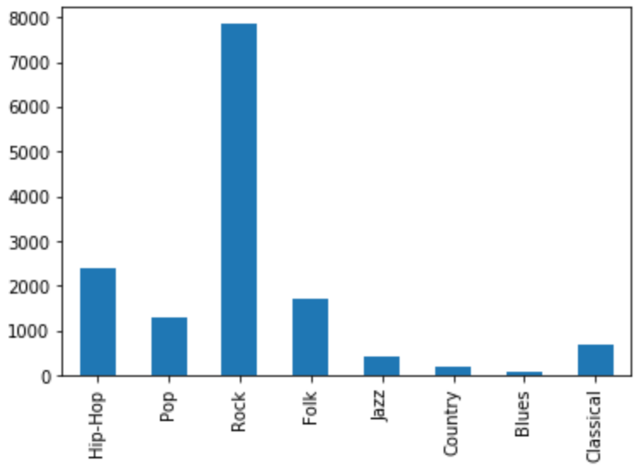

Figure 2: Medium dataset Data Distribution

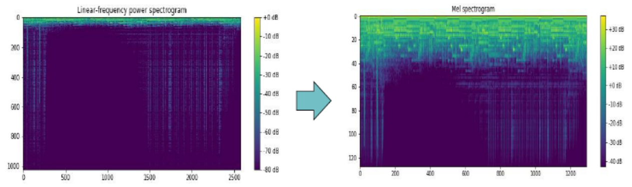

Figure 3: Power Spectrogram to Mel-frequency Cepstral Coefficients (MFCC)

To counter this, we decided to transform the features from the time domain space to the genre space, wherein each song is represented as a probability distribution over the genre classes under consideration. This genre space is used to recommend similar songs for the listener.

To transform our data from the time-frequency domain to the genre space, we have built the CNN and LSTM classification models to predict the genre probability distribution of a song in the output layer.

4.1 Convolutional Neural Network Model

Recently, Convolutional Neural Networks (CNN) have shown some promising results in signal processing tasks. So, we experimented with a CNN in our project considering the mel-spectrogram as a one channel image and we built a seven layered Convolutional Neural Network from scratch. Figure 4 depicts the architecture of our CNN.

We arrived at this architecture after a fair bit of experimentation. We started from a three layered network and increased the number of layers and units in each layer based on the classification performance. We also tried different dropout rates at the dense layer in order to generalize the model. We used these generalization techniques as our model was over-fitting the data (i.e) we had a train accuracy of approximately 92% and test accuracy of around 40%.

After a bit of experimentation, we decided to build the network with four convolution layers and three dense layers. The network

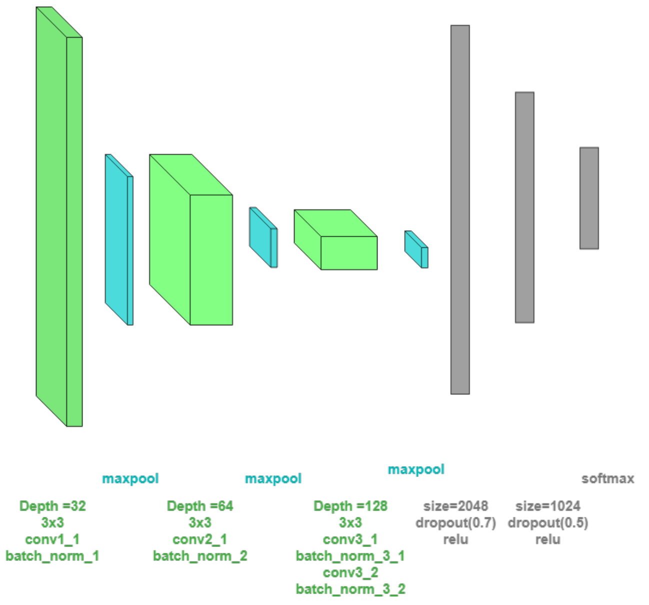

Figure 4: Convolutionl Neural Network

has batch-normalization at convolution layers and dropouts at dense layers for generalizing the model. We used a dropout rate of about 0.7 for the first dense layer and 0.5 for the second dense layer. Finally, with the updated model we were able to achieve a train accuracy of around 86% and test accuracy of around 65%.

4.2 Long Short Term Memory Model

The mel-spectrogram has a sequence wherein every timestamp strongly relies on the immediate predecessors and long term structure of the whole song. So, we decided that the Long Short Term Memory (LSTM) network would be the next obvious choice to build a classification model to predict the probability distribution of a song across genre classes. We constructed an LSTM network to detect the long term structure of the song and its dependencies across a short period of time.

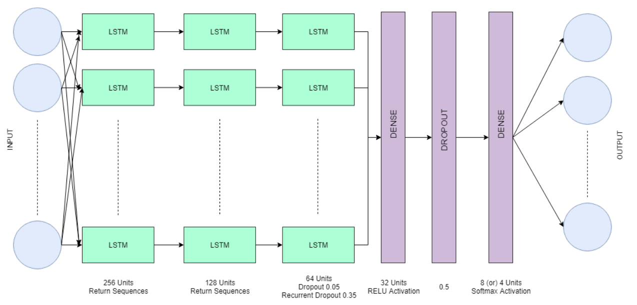

Figure 5: Long Short Term Memory Neural Network

Similar to the CNN, we performed some experimentation with the LSTM network. We started with a three layered network having two LSTM layers and one dense layer. We also added dropouts at the LSTM and the fully connected layer to regularize the model. We kept changing the dropout rates and increased the number of layers and units in each layer based on the classification performance. Again, we modified the structure and dropout rates as the model was overfitting the data (i.e) we had a train accuracy of approximately 97% and test accuracy of around 35%.

After a bit of experimentation, we decided to build the network with four LSTM layers and two dense layers. The network has dropouts at the LSTM and dense layers and a recurrent dropout at the LSTM layer for generalizing the model. We used a recurrent dropout rate of around 35% for the first LSTM layer with a dropout rate of 5%. For the dense layer we added a dropout rate of 50%. Finally, with the updated model we were able to achieve a train accuracy of around 74% and test accuracy of around 57%. Figure 5 depicts the architecture of our LSTM network.

4.3 Playlist generation

After the neural network has been trained, we obtain the genre probability distribution for each song. The cosine similarity of each song in the test set is found with respect to each song in the training set. Cosine Similarity is a distance metric where the results range from 0 to 1 where 0 implies orthogonality and 1 implies that the values are exactly the same. Cosine similarity between the vectors A and B is defined as $similarity = 1 - \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$

Figure 6 below shows the probability distribution of genre classes for the Rock song - 'All the Way to Holloway'. The song belongs to the Rock genre with 55.156% certainty, so it is not surprising that it is labelled as a Rock song. However, two other significant percentages stand out, namely, Pop and Country at 14.249% and 15.604% respectively. This distribution leads us to believe that while 'All the Way to Holloway' is a Rock song, we might see some Pop and/or Country songs appearing in the playlist.

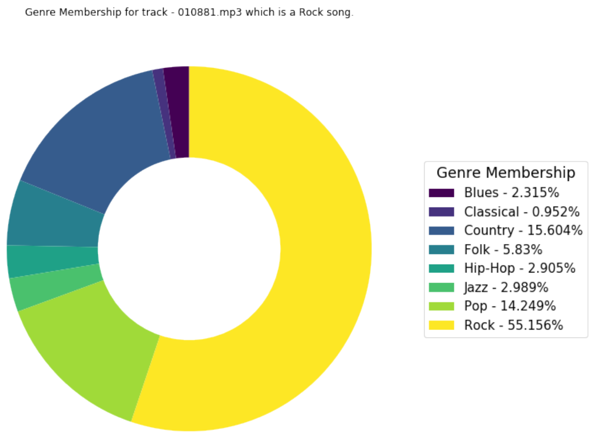

Figure 6: Genre Membership for All the Way to Holloway (A Rock song)

After the cosine similarity is found, the five most similar songs are picked and a playlist is created for each song in the training set. Recommendation is very subjective and so we cannot have an objective score for it. However, since we trained our neural network to reduce the dimension space from the high dimensional spectrogram to the low dimensional genre space, we assume a genre related metric, so each playlist is assigned a score based on the number of songs that match the genre of the training set song. This score is not an objective measure of the playlist's quality. It is purely a measure of the number of genre matches in that playlist.

Table 1: Most Similar Songs Playlist

Song	Artist	Genre
Cheap Cherry Wine	Junk Boys	Rock
Safe on Your Mountaintop	Fat Spirit	Rock
Black Case	Dlina Volny	Rock
Godspeed (Lyndon Scarfe Remix)	Maxwell Powers	Pop
The Words Don't Work	Orange Peels	Pop

Considering that we are looking at the genre space, the basic assumption would be that we would get a playlist where all 5 songs are Rock songs. However, the system found 2 pop songs that were adjudged to be more similar to 'All the Way to Holloway' than all other Rock songs in the training set. This shows that despite the fact that we have introduced genre as a metric, it does not bias the results to create a playlist with only one genre of songs in it.

5 RESULTS

For the purpose of evaluating the models, the hypothesis was that a generated playlist for any song would have songs belonging to the same genre, so we assigned each playlist a generated score ranging from 0 to 1 based on the number of songs in the playlist that matched the test/current song's genre.

For example, the above playlist would have a score of 0.6 because it has 3 matches (Rock songs) and 2 mismatches (Pop songs) out of the 5 songs on the playlist. The score for each playlist is calculated as follows -

$$Score = \frac{\sum_{i=1}^{n} x_i}{n} \tag{1}$$

where n is the number of songs in the playlist (5 for this paper) and

$$x_i = \begin{cases} 1, & \text{if genre is a match.} \\ 0, & \text{if genre is a mismatch.} \end{cases} \tag{2}$$

Table 2: Neural Network Performance

Neural Network	Data Subset	Epochs	Evaluation Score
CNN	Small	100	0.69
CNN	Medium	100	0.57
RNN	Small	50	0.59
RNN	Medium	60	0.48

The evaluation score seen in the table is obtained by taking the mean of the scores of all generated playlists.

6 CONCLUSION AND FUTURE SCOPE

This project demonstrates how we can apply deep learning techniques to music recommendation in order to solve the cold-start and popularity bias problems.

We believe that the system can be integrated with results from collaborative filtering in order to create an enhanced hybrid recommendation system, So, as an extension/improvement of this project, we would like to use the results from collaborative filtering to learn a low dimensional embedding from signal data and find some latent feature representation similar to the genre space for playlist generation using signal data.

We would also like to try the method suggested by Professor Minje Kim, namely, to learn an embedding using user-generated playlists and then test the system by carrying out A/B testing using the newly created models and the state of the art (i.e collaborative filtering).

These methods would certainly eliminate the cold-start and popularity bias issues and would have been the ideal approach to the problem. However, due to constraints in time, we chose to reduce the dimension space by using the genres.

ACKNOWLEDGMENTS

We would like to thank Professor Minje Kim for giving us the freedom to explore different project ideas and guiding us throughout the project.

REFERENCES

[1] Jesus Bobadilla, Fernando Ortega, Antonio Hernando, and Jesus Bernal. 2012. *A collaborative filtering approach to mitigate the new user cold start problem.* Elsevier - Knowledge Based Systems Vol. 26. Universidad Politecnica de Madrid, Spain.

[2] Sylvester Card. 2018. How I taught a neural network to understand similarities in music audio. https://medium.com/@silvercloud438/how-i-taught-a-neural-network-to-understand-similarities-in-music-audio-d4fca54c1aed

[3] Sophia Ciocca. 2017. How Does Spotify Know You So Well? https://medium.com/s/story/spotifys-discover-weekly-how-machine-learning-finds-your-new-music-19a41ab76efe

[4] Sander Dieleman. 2014. Recommending music on Spotify with deep learning. http://benanne.github.io/2014/08/05/spotify-cnns.html

[5] Albert Jimenez. 2016. Music Genre Classification with Deep Learning. https://github.com/jsalbert/Music-Genre-Classification-with-Deep-Learning

[6] Azadeh Nematzadeh, Giovanni Luca Campaglia, Filippo Menczer, and Alessandro Flammini. 2017. *How algorithmic popularity bias hinders or promotes quality.* arXiv:1707.00574v2. Indiana University Bloomington, Bloomington, IN.

[7] Ruoho Ruotsi. 2017. LSTM Music Genre Classification. https://github.com/ruohoruotsi/LSTM-Music-Genre-Classification