https://doi.org/10.69974/glskalp.02.03.32

GLS KALP
Journal of Multidisciplinary Studies
Volume 2; Issue 3;
July - September 2022

Recommendation System based on Artist and Music Embeddings

Akshat Surolia[1]

Abstract

In this paper, we present a personalized music recommendation system based on the embeddings of artists and music. The main peculiarity of our work is that the determining factors of a user's preferences for a developed music recommender system are the artists and the music that they listen to. The artist embeddings inform the network about the contextual representation of artists in a latent space, where similar artists are closer to each other. The music embeddings hold the information about the music. Both embeddings are then combined to form a new embedding, which is then used to predict the user's preferences. We use the Spotify's API to collect the data to train and evaluate the model. Two approaches of building a music recommender system are considered in this paper. Each approach significantly differs in the way the embeddings are learned.

Keywords: *Personalized Music, Artist, Spotify's API, Embeddings*

Introduction

Many music lovers have turned to listen to online music. The development in cloud technology has made it possible for music listeners to stream to all music they want. Spotify, Soundcloud, YouTube Music, Amazon Music and iTunes are some of the most popular music streaming services. People spend a lot of time carefully curating their music playlists, these playlists are made with love and care, something they would want to listen to and spend time with every day. This creates a personalized music experience that is tailored to their preferences.

Many users, however, find it difficult to make a playlist from a large collection of music. As a result, users are more likely to play the next song at random or based on a recommendation. As a result, a good, personalized music recommendation system has become crucial. There are various approaches to building a music recommendation system such as Collaborative Filtering(Shakirova, 2017), Content-based Filtering(Schedl, 2019), and Hybrid approaches(Dai et al., 2016; Hansen et al., 2020) which focuses on user's behavior and listening patterns, of course these approaches work well for large scale datasets. However, the main problem with these approaches is that they do not contain the contextual and semantic information about the music that the user listens to, which might not work that well for new artists, tracks and even when a new user's join since the system will not have enough information to recommend music to the user.

In this paper, we present a personalized music recommendation system based on the embeddings of artists and music. As we believe that artists and music are the determining factors of a user's preferences. This will solve the above problem of scalability and understanding the semantic and contextual meaning of music and user's preference by understanding the audio features and creating a latent space of artists, where similar artists will be closer to each other. We are using Spotify's API to gather similar artists, according to Spotify, Artist's resemblance is based mostly on two things. The first is "shared fans", which is particularly prevalent among artists who operate on the periphery. i.e., the more fans two artists share, and the bigger the percentage of each artist's total fans those shared fans represent, the more similar they are. The second one

[1] Associate Data Scientist, D S Matics, Pune, Maharashtra.

is "shared descriptions"(Alice Wang, Aasish Pappu, 2021) which entails using information from Spotify, blogs, magazines, and other areas where music is discussed for Analyzing online sites and integrating that information with information from other sources about artists to look for trends in descriptive terminology. The more exact a description is, the more likely it is that it will lead to a match between artists who share it. The embeddings are then combined to form a new embedding in multi-modal space, which is then used to predict the user's preferences.

Methodology

Data Gathering

The data gathering process consists of various steps, which are described below.

1. Collecting all the personally curated playlists from Spotify's API, to gather the information about your music taste.
2. Fetching the information about tracks and the artists that they contain.
3. Making the API calls to get the similar artists of each artist, this process is repeated until the point where each artist will have at least 20-25 similar artists and it will also increase the number of artists that we will have for our training dataset.
4. Now that we have all the artists, we need to collect all the tracks of every artist, To do that there can be two approaches,

a) We can use the Spotify API to get all the top tracks of every artist, we make sure that the number of tracks of each artist is at least enough for model to generalize well, but this will take a lot of time and will not guarantee a balanced distribution of tracks that we will have for our training dataset.

b) To overcome the above problem, we can use the Spotify API to get all the albums of every artist and iterate to all the albums to collect every track of those albums. This approach will make sure that we will have at least 100 tracks per artist, yet this process might take some time to complete depending on the number of albums per artists.

c) After gathering all the information about the artists and tracks, we can now collect the audio features of each track and store it in a .csv file. The audio features are the features that we will use to create music embeddings.

d) We have also used dataset "spotify_features" from Kaggle which includes multiple genres of all the tracks, this dataset is particularly used for a downstream task of genre classification to create audio embeddings.

e) Other small dataset from different resources like GitHub and Kaggle are also accumulated to increase the size of test dataset, these datasets were further processed get have a similar structure as our training dataset.

Data Analysis

The data collected for audio features shows the following metrics:

1. Popularity - The value will range from 0 to 100, with 100 being the most common. The popularity of an artist is determined by the popularity of all the artist's songs.

2. Acousticness - A confidence level of 0.0 to 1.0 indicates if the track is acoustic. 1.0 indicates a high degree of certainty that the track is acoustic.
3. Danceability - Based on a mixture of musical factors such as tempo, rhythm stability, bar strength, and overall regularity. A rating of 0.0 indicates that it is the least danceable, while 1.0 indicates that it is the most danceable.
4. Duration - The track's duration in milliseconds.
5. Energy - A scale from 0 to 1.0 that gives a perceptual assessment of intensity and activity. The dynamic range, perceived loudness, timbre, appearance speed, and general entropy are all perceptual qualities that contribute to this feature.
6. Instrumentality - Predict whether a track contains voices. The closer the instrumentalization rating approaches 1.0, the more likely the track lacks vocal material. Values larger than 0.5 represent instrumental cues, but as the value approaches 1.0, confidence grows.
7. Key - The track's tonality, Tones are assigned integer values using Standard Pitch Class notation. The value is -1 if no key is found.
8. Liveness - The presence of audiences in the recording is detected by the liveness feature. Higher liveness values indicate a greater likelihood that the track was performed live. A value greater than 0.8 indicates a high likelihood that the track will be operational.
9. Loudness - The overall volume of a clue measured in decibels (dB), ranges from -60 to 0 db. The sound values are averaged over the course of the track and can be used to compare the relative sound of the tracks.
10. Mode - The mode of a track denotes its modality (major or minor), or the type of scale from which its melodic content is derived. The greatest is represented by 1 and the smallest by 0.
11. Speechiness - The presence of spoken words, the closer the value of the attribute is to 1.0, the more exclusively spoken the recording is.
12. Tempo - A track's estimated global tempo in beats per minute (BPM). it is derived directly from the average duration of times.
13. Time signature - A time signature that is estimated specifies how many times each bar is repeated. The time signature ranges from 3 to 7, corresponding to time signatures ranging from "3/4" to "7/4."
14. Valence - A metric ranging from 0.0 to 1.0 that describes the musical positivity conveyed by a track. Tracks with a low valence sound more positive (for example, happy, cheerful, euphoric), whereas tracks with a high valence sound negative (for example, sad, depressed, angry).

According to this heat map shown below, the strongest correlation, is between loudness and energy (0.82). Furthermore, there are strong correlations between popularity and loudness (0.36), danceability (0.26), and energy (0.25). This is

comparable to valence, which has strong correlations with danceability (0.55), energy (0.44), and loudness (0.4). On the contrary, it is the least related to acousticness. This implies that songs popular on Spotify are likely to be danceable, loud, and energetic. Hip-hop, electronica, or dance music are all possibilities. However, it is possible that it does not accurately reflect the popularity of the music genre in general. Consider the fact that many people still enjoy classic, rock, or blues music.

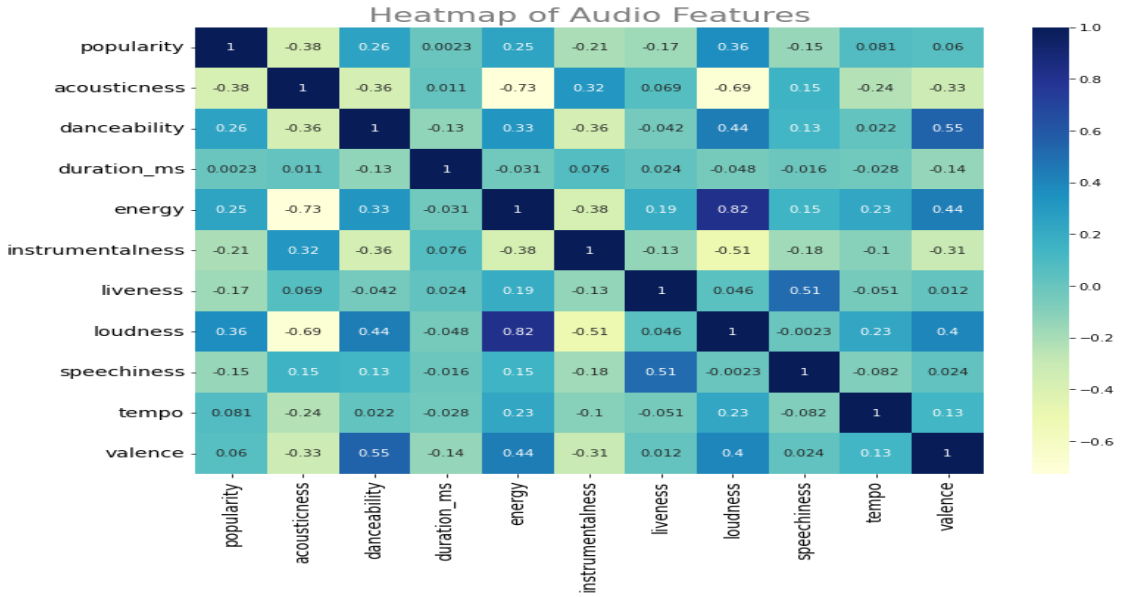

***Figure 1**: Correlation matrix heat map of audio features*

Data Processing

As it seems there are certain features which are linearly dependent or being derived from other features, this might create some issues while training the network so, for our use-case we dropped features like intrumentalness, duration, popularity etc., to remove the redundancy between linearly independent features.

By computing the necessary statistics on the samples in the training set, each feature is individually centered and scaled. All the numeric values are being normalized by removing the mean and scaling by unit variance.

Architecture Proposal

Artist Embeddings:

The notion is that individuals with similar preferences would listen to comparable musicians, therefore similar artists will frequently appear in the same environment. We would have a decent representation of artists that would be beneficial in music recommendations if we could construct vectorial representations of musicians that could communicate these common contexts.

Given a corpus of sentences, models like Word2Vec(Mikolov et al., 2013) learn to represent words that appear in similar verbal situations. If we treat the ids of artists who were co-listened to by users as a sentence, Word2Vec(Mikolov et al., 2013) will learn to give artists who are often listened to concurrently a close embedding. This method captures unconscious tendencies of decision-making. It accomplishes so without considering any records about artists, and as a result, it is privacy-preserving by design.

Word2Vec uses a skip-gram technique with negative sampling to learn if a word belongs in the context of a target word. Each artist's input data consists of a

statement in which the tokens are the ids of other musicians to whom the user has listened: [artist_id 1, artist_id 2,...,artist_id n].
All phrases are randomly shuffled after each pass of the model on the data to provide a new sampling of the possibilities. We utilize a new iterator method to do this while still using the gensim Word2Vec class without change. We count 5 passes on the data for each epoch to slow down the alpha learning rate (which is controlled by the gensim Word2Vec class). We utilize the skip-gram technique with a window size of 5 to learn embedding size of 100 during 50 training epochs.

Spotify, given an artist_id provides a list of similar artists based on the number of times they were co-listened to by users. We use this information to construct a corpus of sentences to train Word2Vec(Mikolov et al., 2013) model to learn embeddings of artists.

Music Embeddings:
Given the input data as audio features provided by Spotify(Saravanou et al., 2021), A simple network consisting of a single hidden layer, is used to learn the embeddings of different tracks. There are two different approaches to train this model differing in downstream tasks.

Artist Prediction
The model is trained to predict the artist_id of a track based on the audio features(Saravanou et al., 2021), The input is (1x8) dimensional feature vector with continuous, normalized values representing Acousticness, Liveness, Danceability, Energy, Loudness, Speechiness, Tempo, and Valence respectively. The output is a one hot encoded vector of length equals to number of artists in the dataset, resulting with the probabilities of the track being of a particular artist. In our case the number of artists which the model was trained on was 70,000, so the dimension was (1,70000). Since a single track can have multiple artists, we use the SoftMax function to normalize the probabilities and used a Binary Cross Entropy loss function. It turns out using a cross entropy loss function is not a good choice for this task, since it's a multi-class, multi-label classification problem
The size of hidden dimension can vary depending upon the number of artists in the training set, but as per our empirical study, the size of hidden dimension was found to be around 512.

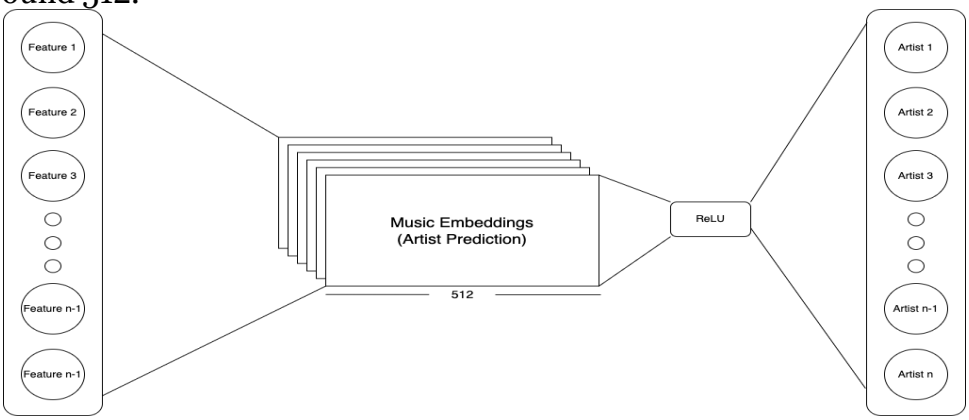

Figure 2: *Model Architecture for Music Embedding with Artist Prediction*

Genre Classification
The model is trained to predict the genre of a track based on the audio features(Saravanou et al., 2021; Sturm, 2013). The input is the same (1x8) dimensional feature vector. There are a total of 12 genres in the dataset, namely

Indie, Jazz, Pop, Electronic, Folk, Hip-Hop, Rock, Alternative, Classical, Rap, World, Soul, Blues, R&B, Reggaeton, Reggae, Dance and Country. So, the output is a one hot encoded vector of length 12. A single track can have multiple genres; therefore, we used the SoftMax function to normalize the probabilities and used a Binary Cross Entropy loss function. The size of hidden dimension was found to be around 128.

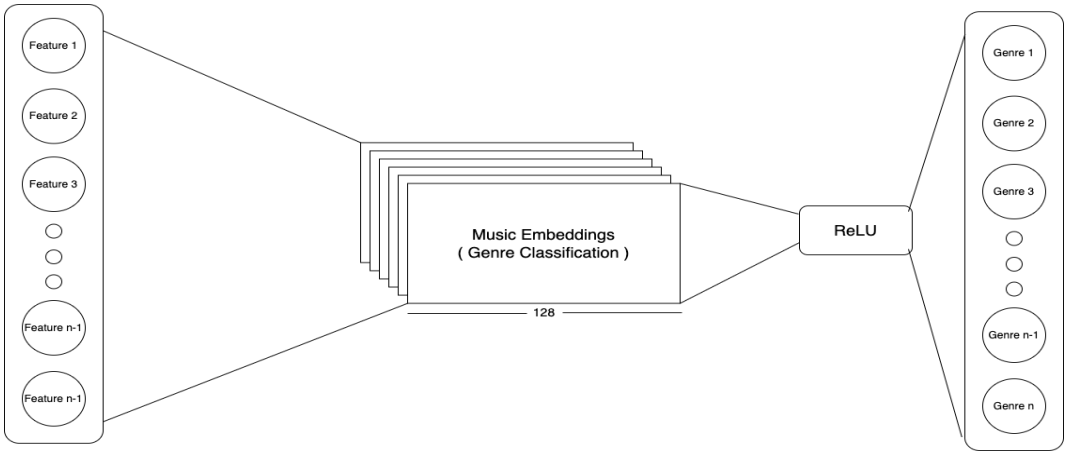

***Figure 3**: Model Architecture for Music Embedding with Genre Classification*

Recommendation System

Based on the personal preference, A dataset was created with the above features and artist ids as input and a categorical target column indicating is the tracks is preferred by the user or not. The dataset was then split into training and test sets.

We experimented with employing pre-trained embeddings in a neural network via embeddings layers, which allows us to learn more complex preferences patterns than simple Euclidean distance without the time-consuming data pre-processing. In actuality, a PyTorch neural net was built using two independent, immutable embeddings that were initially filled with zeros before being filled with the pre-trained embeddings when possible. Their concatenation is followed by three Linear layers: Linear (32, ReLU) and the final Linear (1, sigmoid), using the Adam optimizer with cross entropy loss.

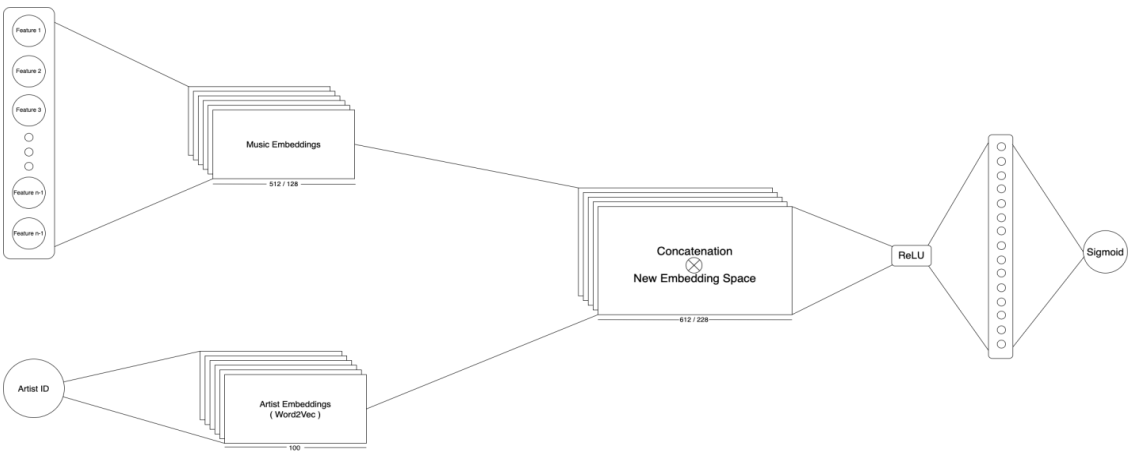

***Figure 4**: Model Architecture for Recommendation System*

Results

The models were trained for variable number of epochs with early stopping, and the best model was selected based on the validation loss. The test set was used to evaluate

the model. We used learning rate schedulers to reduce the learning rate of the model as the training progresses.
We use accuracy and loss as our evaluation metrics: accuracy is the fraction of the predictions that are correct. The loss is the binary cross entropy loss between the predictions and the target.

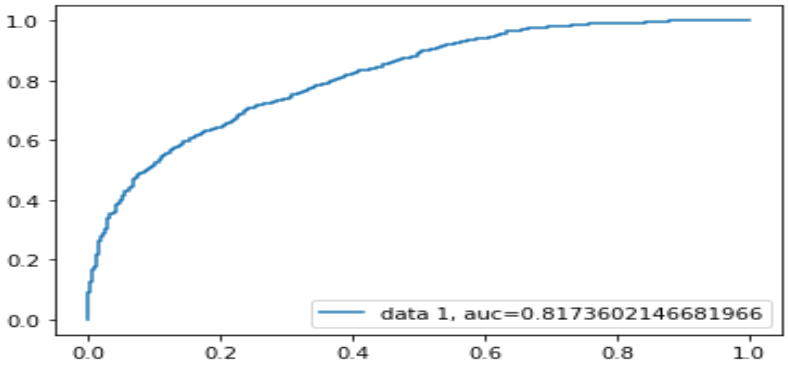

Figure 5: *ROC_AUC curve showing accuracy*

	Validation Loss	**Validation Accuracy**
Genre Classification	20.62%	53.34%
Artist Prediction	37.43%	31.51%
Recommendation System	12.70%	**83.34%**

Discussion

As per the evaluation metrics, the genre classification model performed better than the artist prediction model, this may be due to the fact that the artist prediction model has a more complex preference pattern, also since our input are linear structured features there is not much information for model to learn to distinguish between the different artists. This may be overcome with changing the input to MEL Spectrograms(Stoller et al., 2019) of the real audio segments of the tracks. The genre classification task is also a good task in learning the musical embeddings, we might also use valance prediction as a downstream task to learn the embeddings, since valance signifies the mood of the track, and concatenating this with learned embeddings of the lyrics it will create good embeddings for recommendations system. Yet the idea behind the task of artists prediction is use both the output and the embeddings to learn the preferences of the user. The predicted id can be used as an input for artist embeddings (word2vec) and the embeddings can be used as an input for the recommendation system. The good thing about this approach is that even though we might not have all the artist in our dataset, the network can still predict the id of someone who has similar style of music, which is in our dataset, therefore this approach is scalable and robust. Interestingly, we found that though the accuracy of genre classification model and artist prediction model is low but still the embeddings are performing quite well for the recommendation system.

Conclusion

There is no question that different ways of encoding data may alter the efficiency and performance of algorithms. Although many of the approaches have been tried, the

best performing one is the one that uses a neural network to learn embeddings, yet most of the approaches have focused on the task of learning and understanding the music itself, which can be bit tricky since human mind can be biased towards certain genres and artists. Maybe it's due to the fact that they connect with the artist or relate to the story. In this paper, we attempted to fill this need by delivering combined representation of music and artists in a multi-model embedding space to learn the user's preferences. We demonstrated that our models could acquire significant musical structure. Furthermore, we believe that more effort might be put into building a better assessment measure; only then will we be able to train models that can properly identify and learn human preferences.

References

Alice Wang, Aasish Pappu, H. C. (2021). Representation of Music Creators on Wikipedia, Differences in Gender and Genre | Proceedings of the International AAAI Conference on Web and social media. https://ojs.aaai.org/index.php/ICWSM/article/view/18101

Dai, H., Wang, Y., Trivedi, R., & Song, L. (2016). Deep Coevolutionary Network: Embedding User and Item Features for Recommendation. http://arxiv.org/abs/1609.03675

Hansen, C., Hansen, C., Maystre, L., Mehrotra, R., Brost, B., Tomasi, F., & Lalmas, M. (2020). Contextual and Sequential User Embeddings for Large-Scale Music Recommendation. RecSys 2020 - 14th ACM Conference on Recommender Systems, 53–62. https://doi.org/10.1145/3383313.3412248

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013, January 16). Efficient estimation of word representations in vector space. 1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings. https://arxiv.org/abs/1301.3781v3

Saravanou, A., Tomasi, F., Mehrotra, R., & Lalmas, M. (2021). Multi-Task Learning of Graph-Based Inductive Representations of Music Content. Proceedings of the 22th International Society for Music Information Retrieval Conference, ISMIR 2021. https://doi.org/10.5281/ZENODO.5624379

Schedl, M. (2019). Deep Learning in Music Recommendation Systems. In Frontiers in Applied Mathematics and Statistics (Vol. 5, p. 44). Frontiers Media S.A. https://doi.org/10.3389/fams.2019.00044

Shakirova, E. (2017). Collaborative filtering for music recommender system. Proceedings of the 2017 IEEE Russia Section Young Researchers in Electrical and Electronic Engineering Conference, ElConRus 2017, 548–550. https://doi.org/10.1109/EIConRus.2017.7910613

Stoller, D., Ewert, S., & Dixon, S. (2019). Training Generative Adversarial Networks from Incomplete Observations using Factorised Discriminators. https://github.com/f90/FactorGAN

Sturm, B. L. (2013). Classification accuracy is not enough: On the evaluation of music genre recognition systems. Journal of Intelligent Information Systems, 41(3), 371–406. https://doi.org/10.1007/s10844-013-0250-y