Music Recommendation System Using Audio

Rhythm Patel
rhythmkumar18083@iiitd.ac.in
IIIT Delhi
New Delhi, India

Mohnish Agrawal
mohnish18053@iiitd.ac.in
IIIT Delhi
New Delhi, India

Abhinav Suresh
abhinav18003@iiitd.ac.in
IIIT Delhi
New Delhi, India

Shaunak Pal
shaunak18098@iiitd.ac.in
IIIT Delhi
New Delhi, India

Ankit Mishra
ankit18019@iiitd.ac.in
IIIT Delhi
New Delhi, India

Rajat Prakash
rajat18078@iiitd.ac.in
IIIT Delhi
New Delhi, India

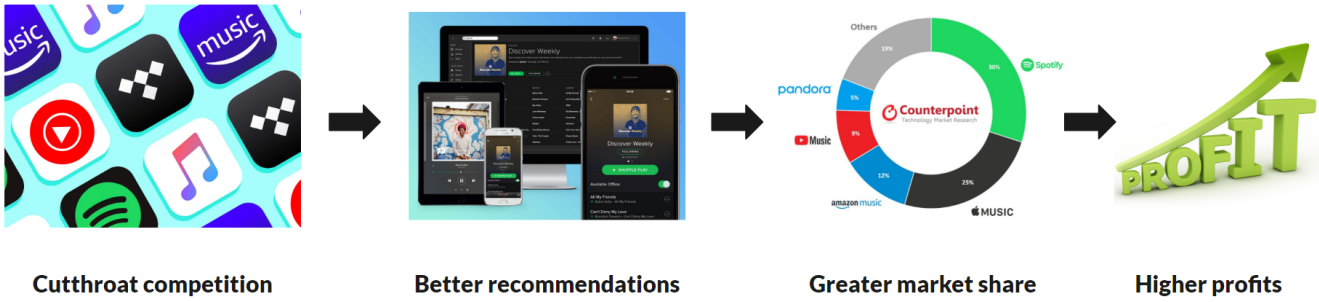

Figure 1: Problem Statement

KEYWORDS

information retrieval, music recommendation, deep learning, spectrogram

ACM Reference Format:
Rhythm Patel, Mohnish Agrawal, Abhinav Suresh, Shaunak Pal, Ankit Mishra, and Rajat Prakash. 2018. Music Recommendation System Using Audio. In *Proceedings of (Information Retrieval).* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 ABSTRACT

A recommendation system is a task to produce a list of recommendations for a user. The most common systems are hybrid models, which leverage the use of songs and user data to recommend songs to users. The former uses a content-based filtering mechanism, while collaborative-based models do the latter. Here, we make our recommendation algorithm using a content-based mechanism, wherein each song is converted to embeddings using a CNN model. The embedding is then passed through KD-Tree to retrieve the best possible recommendation for the user. This is important because since there is a cutthroat competition between music streaming platforms, better recommendations enable them to get a greater market share which drives profits for these companies. We have also implemented a search algorithm to search songs by artist, title as well as lyrics of the song.

GitHub link for the project: https://github.com/AbhinavSE/IR2022_Project_8

Information Retrieval, Winter 2022, IIIT Delhi

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

2 INTRODUCTION & PROBLEM STATEMENT

With the rapid growth of numerous digital music-streaming platforms like Spotify, Apple Music, and Amazon Music, the music content on the internet is greater than ever before, and so is the competition between them. For example, Apple music subscribers grew a whooping 420% in just 5 years (there were 15 million subs in June 2016, which grew to 78 million subs in June 2021). [2]. It is impossible for any human to manually sort all this music and recommend the specific songs to users.

This is where Music Recommendation Systems come in. With various algorithms and a clean UI, these systems make it easier for customers to browse through millions of songs by music recommendations based on artist, genre, instrument, and reviews. The most popular type is genre: pop, rock, hip-hop, R&B, instrumental, indie, jazz etc.

We all know that there is a cutthroat competition between these music streaming platforms (Fig. 1). A better experience for the user enables them to get a greater market share. Consequently, one of the main features for the user is the recommendation system which recommends songs that a given user likes. Thus, such features drive

higher profits for these for-profit companies. This is why music recommendation systems are important.

However, the perfect music recommendation system does not exist that has 100% accuracy. Another thing to note is that these are billion dollar companies, if not, trillion dollars, and have spent hundreds of millions of dollars on making a very very effective system. As a course project, we cannot have a better system than them. However, we can think outside the box and support creative & alternative ways of solving the problem.
The difference is: we will be recommending music using the audio content of the songs using spectrograms, along with/instead of other metadata which is the norm. This is described in the next sections. We have created a fully fledged web application using Dash. We have also implemented IR search algorithms which can search by song name, artist, and lyrics. Also, one more thing to note is that this is an engineering project.

3 CURRENT SYSTEMS & LITERATURE REVIEW

"A Survey of Music Recommendation Systems with a Proposed Music Recommendation System" [4] is a review paper and provides a survey on different recommendation systems currently in use. They include content-based, collaborative, emotion-based, and other techniques. The paper also gives an in depth comparison of these systems, by exploring the strengths and weaknesses of each of them. Finally the paper also gives an overview of an improved hybrid recommendation system that claims to solve many of the problems that current systems face.

"Automatic Music Recommendation Systems: Do Demographic, Profiling, and Contextual Features Improve Their Performance?" [7] introduces a dataset of listening histories along with the listener's demographic informations and some other features that characterize aspects of listening behavior. This dataset is then used for the evaluation of accuracy of a music recommendation model. The paper's results show that using features like the listeners' age, country, and gender improve the recommendation accuracy by around 8 percent, and also when a new feature called "exploratoryness" was used, the accuracy increased by around 12 percent.

"Recommending music on Spotify with deep learning" [1] is written by Sander Dieleman who was interning at spotify at the time.he explains his work at spotify and some preliminary results. The article gives a brief overview of collaborative filtering, its virtues and its flaws. The article also describes content-based recommendation for when no usage data is available, and about predicting listening preferences with deep learning, which is music recommendation based on audio signals. Finally the article talks about what the covnets learn about music and some potential applications of his work.

"Why Spotify's music recommendations always seem so spot on" [3] describes about information about a user's listening history, music preferences, duration of play for certain songs, and how they respond to recommendations is fed into an algorithm (are they liking them, skipping them, replaying them, saving them). Spotify is attempting to model user behavior on the app by determining ways for projecting in-app actions into human features and emotion, as well as tying music experiences to mood and situational settings such as time of day, week, or season. CoSeRNN, a neural network that weighs specific variables such as prior listening history and current context to generate song choices that are appropriate for the moment, has been put to the test. Spotify groups music and users together based on shared features or qualities using a machine learning tool called the approximate nearest-neighbor search algorithm. MUSIG that learns "meaningful representations of tracks and users" based on individual features of songs (like genre, acoustics, danceability, the wordiness of lyrics) and how they relate to one another (like if they appear on the same playlist)

"How Spotify recommender system works" [5] shows how advanced NLP can also be executed on actual lyrics of the songs to indicate specific topics around which songs are composed. This might also be used for recommending songs with the same topics, but, possibly, from different genres. This feature helps with the cold-start problem. Raw audio has been processed, the CNN assigns some characteristics to each song. It can detect some of the key characteristics like mode, rhythm, loudness and sometimes even genre of song. Using these metrics CNN can then categorize songs in groups that have similar characteristics.

A paper published in 2012 [6] was one of the first papers to use Convolutional Neural Networks (CNN) to implement a music recommendation system suggesting the first content based system.

This paper from NUS Singapore scientists [8] improved upon Sander's method and proposed a novel algorithm on deep belief networks and probabilistic graphical methods that outperformed Collaborative Filtering in warm and cold start stages. Finally they combined Collaborative Filtering and their own model to outperform baseline Collaborative Filtering models with Root mean squared (RMS) reduced to 0.34 for warm start and 0.47 for cold start.

4 METHODOLOGY - DATASET

4.1 Free Music Archive Dataset

Total tracks : 106574
Total genres : 163

Free Music Archive (FMA) is an open and easily available dataset which is capable of evaluating several tasks in Music Information Retrieval (MIR). MIR is a field concerned with browsing, searching, and organizing large music collections.

The full dataset size is 879 GB. But due to constraints, we are using the smaller version of this, which is 7.2 GB and has 8 genres (Pop, Rock, Instrumental, Hip-Hop, International, Electronic, Folk, Experimental)

4.2 CSV files with description

(1) **tracks.csv** - It contains metadata for all the tracks. There are 106574 tracks/rows and 6 columns : ID, title, artist, genres, tags and play counts.

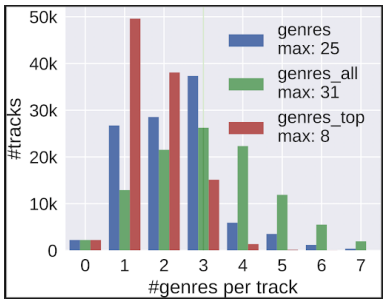

Figure 2: Genres per track

(2) **genres.csv** - It consists of names and parents of all the genres (163 genres) discussed. It is important for study on the basis of Hierarchy of genre and find the root genre.

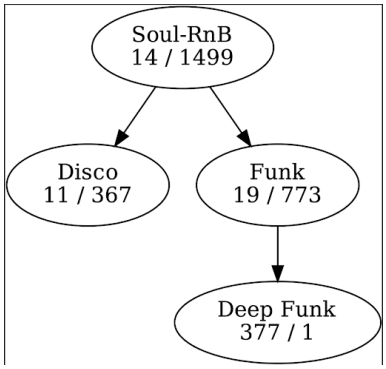

Figure 3: Genres per track

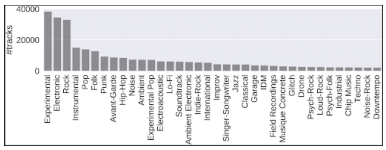

Figure 4: Genres per track

(3) **features.csv** - Some common features extracted using librosa.

(4) **echonest.csv** - Audio features by Spotify (previously known as Echonest) for 13129 tracks out of total tracks.

4.3 Multiple sizes of MP3-encoded audio data

(1) **fma_small.zip** - 8,000 tracks of 30s, 8 balanced genres (7.2 GB)

(2) **fma_medium.zip** - 25,000 tracks of 30s, 16 unbalanced genres (22 GB)

(3) **fma_large.zip** - 106,574 tracks of 30s, 161 unbalanced genres (93 GB)

(4) **fma_full.zip** - 106,574 untrimmed tracks, 161 unbalanced genres (879 GB)

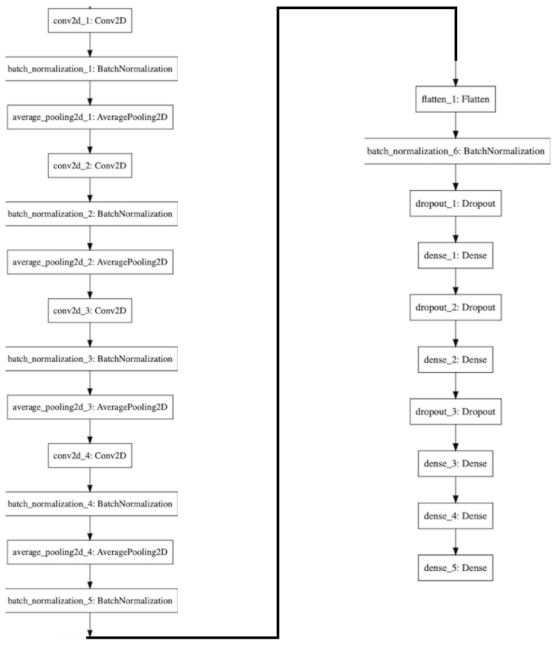

Figure 5: Baseline model structure

5 METHODOLOGY

5.1 Baseline

Our baseline model is shown here in Figure 5. We finally got a training accuracy of 78% and validation accuracy of 67% which is pretty good for a dataset of just 8000 samples. The model was trained in 20 epochs with a learning rate of 0.0001. The graphs representing the training history is also shown here in Figures 6, 7, 8, 9.

Using this model we can get a latent representation of any song by taking a 30 second sample and then use it to find similar songs using cosine similarity.

Drive link for the Keras weights: https://drive.google.com/drive/folders/1ch8F0XOf50H9PRuX3q0w2YcLaVnm9R_A

GitHub link for the project: https://github.com/AbhinavSE/IR2022_Project_8

5.2 Recommendation algorithm

The recommendation model was first trained using data from the Free Music Archive (FMA) with around 8000 song samples. Each song is a 30 second clip which was then converted to its spectrogram using the librosa library. This spectrogram is used as the

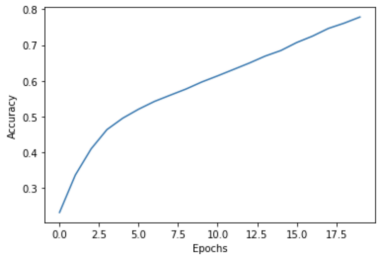

Figure 6: Accuracy v epochs

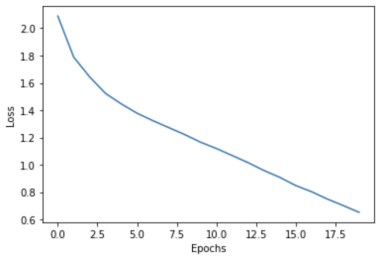

Figure 7: Loss v epochs

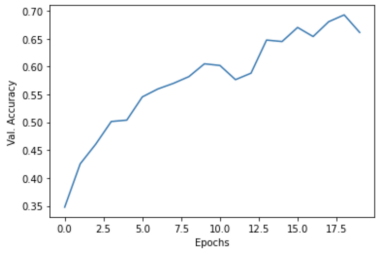

Figure 8: Validation Accuracy v epochs

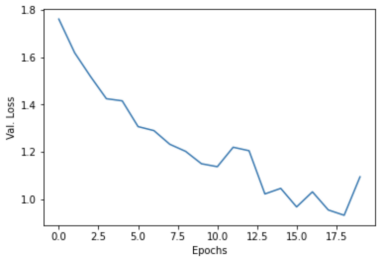

Figure 9: Validation Loss v epochs

input to the model and the output is the genre. After the model was trained, we dropped the final layer and used the model as an encoder for a feature vector of size 32.

Our final database consists of around 400 mainstream songs. These songs are converted beforehand to their feature vectors by passing their spectrograms into the model and then saved into an embeddings file. These feature vectors are then passed into a KD

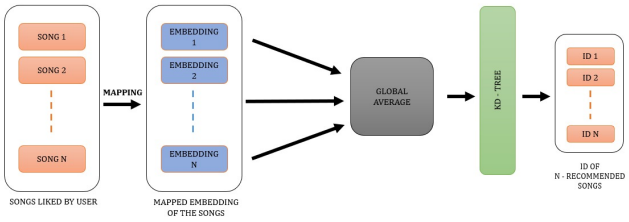

Figure 10: Recommendation system

tree so as to process the queries faster and in an optimized fashion.

To get the recommendations of the user, we first get the songs which the user liked from the front end as soon as they click on the like button. We collect the feature vectors from the embeddings file which we mentioned above. We then average all these vectors to get a feature vector representing the type of songs the user likes. This user vector is then passed into the KD tree to get the nearest neighbors of this user vector. These neighbors basically are the songs the user will most probably like based on the content of the previously heard songs which he/she liked. These songs are then shown to the user on the front end.

The diagram for the recommendation system is here. 10.

5.3 Search Algorithm

We have incorporated different retrieval techniques in our search functionality based on the filter provided to our search engine. The two different type of techniques used are:

1. n-gram character based search algorithm for Artist, Song Album

We use n-gram character based search algorithms to search for songs, filtering by artists, by the title or by the album. We use uni, bi, and tri-gram to make a dictionary mapping of k-grams to the artist/album/title. We choose songs with the specified k-gram while dynamically entering songs into the search box, and then use regex to search linearly over the shortlisted songs.

We use this technique owing to its high latency and low memory requirements over other techniques.

Using a Trie based method would be fast but would require more memory to store. The tradeoff between latency and memory is a lot rendering it useless for our use case.

2. TF-IDF based Lyrics search

For lyrics based search engine, we used TF-IDF to find songs which may contain the lyrics provided by the user. We use TF-IDF owing to its speed, simplicity & surprising accuracy.

We build the index beforehand to ensure that the system is efficient. We process the query and search on the as soon as the search

button is clicked.

6 HANDLING NEW/UNSEEN DATA

We have also incorporated a function to add new songs to the library if required. For n-gram character based techniques, we do not have to build the index from the beginning. We just process the song and add it directly to the index.

For lyrics based search engine, the index is rebuilt from scratch after adding the token from the lyrics into the vocabulary. Since the dataset is not very large, we do not have to worry about the efficiency of the system. An alternative would be to add all the metadata in a temporary folder and rebuild the index regularly from scratch in set intervals of time (1-2 days/1-2 weeks)

7 THE FRONTEND

The app contains 2 views which we can navigate to with the help of the navbar. The Music view contains the following:

- The top of the app shows different album covers in a carousel.
- There's a search bar to search through the existing songs based on Title, Artist, Lyrics and Liked songs.
- The songs are contained in cards which have a like button and a play button to play the song.
- The song player is fixed at the bottom showing the song an artist name and other player controls.
- There's an Add view with the help of which we can add new songs to the database and then later search and play it in the music view.

8 CONCLUSION - ANALYSIS & EVALUATION

Our training dataset just consisted of 8000 samples (7.2 GB). We could have added more but the preprocessing (converting to spectrograms slices) was time consuming. Preprocessing these 8000 samples took around 6-8 hours.

The Free Music Archive also doesn't have any popular mainstream music. Thus using a paid dataset with better and more samples can result in better results and this is our plan in the future.

Deploying the current prototype has a lot of challenges since there isn't enough space to store the final database of songs on the free hosting platforms like Heroku. We are planning to host this on a paid hosting platform.

9 CONTRIBUTIONS

Alphabetical order

- **Abhinav** - Recommendation Algorithm, Literature Survey
- **Ankit** - Search Algorithm, Literature Survey
- **Mohnish** - Search Algorithm, Metadata extraction, Dataset collection
- **Rajat** - Literature Survey
- **Rhythm** - Recommendation Algorithm, Literature Survey, Report & Presentation
- **Shaunak** - Frontend, Literature Survey

10 WORK TIMELINE

Week 1 - 5

Chose the exact problem statement and conducted existing literature review to get a sense of the problems faced in the current systems Chose the dataset that will be used for building the model

Week 6

The team was divided into sub-teams - A team that built the search and recommendations model themselves, and building the frontend UI/UX. The teams started working on their respective micro-problem statements.

Week 7

For building the model, the focus was shifted towards learning about various features that could be extracted from audio signals and then worked on extracting that information from the dataset. Built the baseline model. The type of system, the architecture and the inner working of the system was decided.

Week 8 - 9

Built upon the baseline to make other models, evaluated them, and chose the best one. Started working towards finalizing the recommendation model.

Week 10-12

Everyone shifted their focus and started working on the frontend and then integrating it with the backend.

Week 13-14

Added finishing touches to the frontend including music player. Went over the methodology followed, found and fixed faults within the system. Implemented functionality when new/unseen songs are searched for. Wrote up the report for the final presentation.

ACKNOWLEDGMENTS

We thank Dr Rajiv Ratn Shah, the professor of Information Retrieval course, for providing his constant help and support in the past, present, and the future. We also thank the Teaching Assistants for their guidance as well. We couldn't have done this project without them.

REFERENCES

[1] Sander Dieleman. 2014. Recommending music on Spotify with deep learning. https://benanne.github.io/2014/08/05/spotify-cnns.html
[2] Marie Charlotte Götting. 2022. Apple Music subscribers 2018 | Statistic. https://www.statista.com/statistics/604959/number-of-apple-music-subscribers/
[3] Charlotte Hu. 2021. Why Spotify's music recommendations always seem so spot on. https://www.popsci.com/technology/spotify-audio-recommendation-research/
[4] Dip Paul and Subhradeep Kundu. 2020. A Survey of Music Recommendation Systems with a Proposed Music Recommendation System, Jyotsna Kumar Mandal and Debika Bhattacharya (Eds.). Springer Singapore, Singapore, 279–285.
[5] Daniel Roy. 2020. How Spotify recommender system works. https://www.linkedin.com/pulse/how-spotify-recommender-system-works-daniel-roy-cfa/

[6] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2013/file/b3ba8f1bee1238a2f37603d90b58898d-Paper.pdf

[7] Gabriel Vigliensoni and Ichiro Fujinaga. 2016. Automatic Music Recommendation Systems: Do Demographic, Profiling, and Contextual Features Improve Their Performance?. In *ISMIR*.

[8] Xinxi Wang and Ye Wang. 2014. Improving Content-Based and Hybrid Music Recommendation Using Deep Learning. In *Proceedings of the 22nd ACM International Conference on Multimedia* (Orlando, Florida, USA) *(MM '14)*. Association for Computing Machinery, New York, NY, USA, 627–636. https://doi.org/10.1145/2647868.2654940