Music Genre Detection And Recommendation System

submitted in partial fulfillment of the requirement
for the award of the Degree of

Bachelor of Technology
in
Computer Engineering

by

Shardul Mahindrakar
Aayush Mohite
Aakash Mokani

under the guidance of

Prof. Sunil Ghane

Department of Computer Engineering
Bharatiya Vidya Bhavan's
Sardar Patel Institute of Technology
(Autonomous Institute Affiliated to University of Mumbai)
Munshi Nagar, Andheri-West, Mumbai-400058
University of Mumbai
October 2020

Project Approval Certificate

This is to certify that the Project entitled "Music Genre Detection And Recommendation System" by Mr. Shardul Mahindrakar, Mr. Aayush Mohite and Mr. Aakash Mokani is found to be satisfactory and is partially approved for the award of Degree of Bachelor of Technology in Computer Engineering from University of Mumbai.

External Examiner **Internal Examiner**

(signature) **(signature)**

Name: **Name:**

Date: **Date:**

Seal of the Institute

Contents

List of Figures

List of Tables

List of Abbreviations

MFCC	Mel-Frequency Cepstral Coefficients
ML	Machine Learning
TFIDF	Term Frequency - Inverse Document Frequency
KNN	K-Nearest Neighbour
FFT	Fast Fourier Transform
API	Application Programming Interface
CBAR	Content Based Audio Retrieval
IEEE	Institute of Electrical and Electronics Engineers
WAV	Waveform Audio File Format
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets

Abstract

Web applications used for streaming of music and content showcasing have become common. These products often hamper user experience by usually focusing on the streaming aspect. Doing so, they don't stress enough on catering content with respect to the users' needs. So it would be advantageous to replace just a music streaming website with an optimized music recommendation system which could also cater to the needs of a naive music listener. Our project solves this issue by integrating 2 features in a single web application. Namely the 'Genre Detection' and 'Music Recommendation '. It focuses more on cross-user compatibility. So, be it a professional musician or a bathroom singer, each and every user gets value out of this website. It is especially helpful for the naive music listeners as one can easily get to know the accurate genre of a song which they like or can know about similar songs. Overall this improves User experience drastically. Adding to this, a chat-feature inspired from discord is also implemented . With this, the people around the world with similar music taste can interact with reach other by joining a particular chat room based on their interest. This is one of the features which modern music streaming platforms lack. It is all about harmonizing a community;people with shared music interest can debate over anything and everything! This project can offer high scalability as it can be integrated with an actual streaming platform to stream the songs which the Recommendation system recommends. Also, we could add some features to cater to the needs of professional musicians so that they can analyse a particular audio file and get insights about the same.

Chapter 1

Introduction

Many a times, we like a song but are unable to find/tell it's genre. This can be quite important if you want to listen to similar songs and enrich your music experience thus we felt the need to provide with genre detection for music as well as recommend similar songs. Additionally we have implemented a chatroom feature to help users interact with each other based on their liking and share their thoughts and personal recommendations.

So following are the features our website offers:

- Music genre detection.
- Music recommendation platform.
- A chatroom for music enthusiasts to share their views.

Implementing a Genre Detection system involves **mainly 3 aspects**. Firstly when a user uploads an audio file(.wav format), that audio file is scanned and then processed using an audio library in python called as **'Librosa'**. In this step, the distinct audio features like tempo, rhythm, instrumentation,timbre ,etc. These audio features form the basis of audio file creation. Analyzing these features using algorithms and signal processing technology requires a clean piece of code.The features can be extracted using methods like Fast Fourier transform(FFT), Mel-frequency cepstral coefficients (MFCC), or deep learning models trained on large music databases. In this project we will be using **MFCC** to extract audio features. The second aspect is the ML model training and scaling of audio features. Scaling is necessary before model training as the model being used is a **KNN** classifier which relies on Euclidean distance to determine the similarity between data points. The last aspect is the prediction of Genre using this trained model. the model is trained on a dataset containing 1000 songs(100 songs each from 10 genres). The **prediction accuracy** achieved is expected to be around 70 percent.

This report presents an idea to overcome the existing problem of **inaccuracy in genre detection** as seen in many products on the internet. Further this project aims towards providing a seamless and highly reliable recommendation system which can improve user experience drastically. Further the implementation of the **chatroom feature** in the same app will enable a user to share or discuss their thoughts with a powerful community of wordwide users.

1.1 Motivation

This work was motivated by the need to inculcate more innovation and customization from the user's perspective. The currently available products on the internet are not always the perfect form of a solution. So, by adding **personal customizations** a website can be created which uplifts the existing shortcomings. Further, the technical challenges faced are vast in the field of development. This project includes complex algorithms and data processing techniques in which there is always room for improvement and optimization. Some of the shortcomings are: cumbersome interfaces, limited customization options, or lack of interactivity in the website.

Further, the accuracy in prediction or music recommendation is not much optimized. This product could work towards achieving **more sustainable** and an **accurate model**. Apart from this, our **passion for music** and technology and the drive to integrate these two aspects is an altogether a strong motivation to carry forward with this project. Also, while scaling this project on a large level, we can possibly be able to cover the gaps or niches in the market. For example we can try to target specific user demographics , genres, music preferences that are not actually addressed by the competitors.

1.2 Objectives

- To study user authentication and backend integration.
- To study audio processing methods and implementing one method.
- To learn Machine Learning algorithms to train model based on existing data.
- To study feature-importance analysis using machine learning models; particularly RandomForestClassifier and DecisionTreeClassifier .
- To study the implementation of **Spotify API** for fetching song details.
- To learn **Streamlit UI** to effectively portray the recommendation results.
- Implementing a server based chatroom using **Socket.io.**
- Enhancing **User experience** by making the website user friendly.

1.3 Problem Statement

Addressing the Complexity of Music Genre Identification and Recommendation and Enhancing overall User Experience. Provide a chatroom feature for like-minded users to interact and share their personal favorites with each other.

Chapter 2

Literature Survey

While working on audio processing and analysis, we got to know a lot about content based audio retrieval techniques and through that we were able to decide to go with MFCC feature extraction. **Content Based Audio Retrieval (CBAR)** has been a growing field of research for the past decade. To be capable of classifying and accessing the audio files, relevant to user's concern, is fundamental for structuring multimedia web search engines. Although we implemented the MFCC technique, we also learnt about the sort-merge technique. There is a research paper based on this topic on the IEEE Xplore website which helped us a lot. Link-https://ieeexplore.ieee.org/abstract/document/6850234

With respect to the Music recommendation model, we learnt about various natural language processing tools. The techniques used finally in our project are : **TF-IDF (Term Frequency-Inverse Document Frequency)** vectorization and **cosine similarity** . These are used mainly for information retrieval tasks such as in document classification, search engines, and recommendation systems.

The data obtained after defining a dataset on which the model is to be trained in Music recommendation systems is the lyrics of each and every song in the data frame. Firstly, **stemming** is used to normalize the data. Stemming is a text processing technique used to reduce words to their root or base form, known as the "stem." We got to know about this as and when we were researching about how to start with the model training after cleaning of data. So, the purpose of stemming is to normalize variations of words(in this case lyrics) that have the same meaning but derived word forms. For example stemming might convert a group of words like 'beauty', 'beautiful', 'beautifully' into 'beauti'.

After vectorization using TF-IDF, each document is represented as a vector where each dimension corresponds to a term, and the value of each dimension is the TF-IDF score of the corresponding term in the document. Cosine similarity is a measure of similarity between two vectors in a multi-dimensional space, often used to calculate similarity between documents represented as vectors. In lay man's terms, it is related to calculating the cosine of the angle between the existing data and the data on which prediction is to be done.

.

Chapter 3

Design

We have a landing page wherein you have the options of detecting the genre by uploading a .wav file yourself, get recommended similar songs by selecting one from our sizable database or chat with people by selecting one of your favorite genres and share your thoughts on the music scene. The detection and recommendation is done by their respective ML models trained on relevant datasets and the chatroom feature uses socket.io.

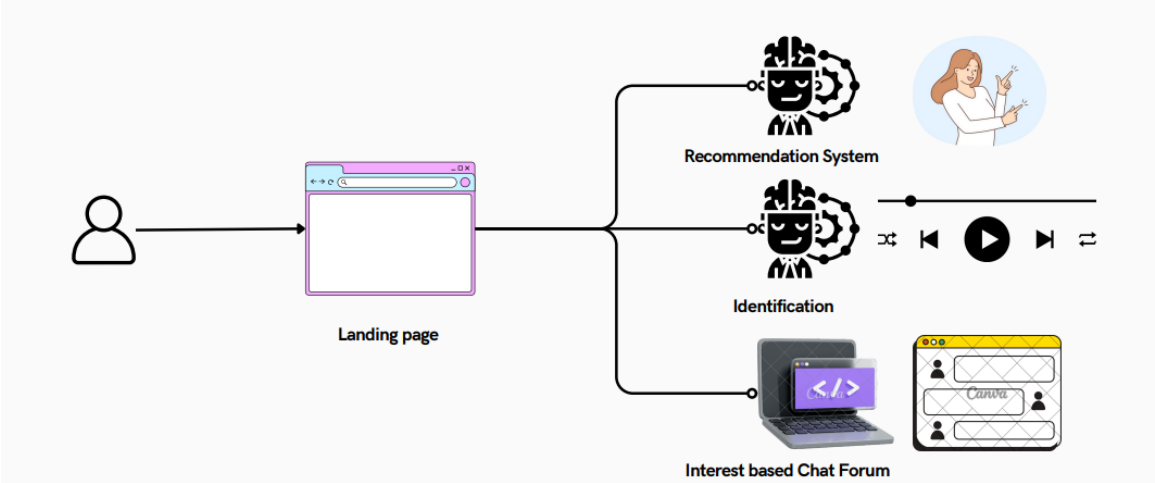

Figure 3.1: Block diagram of the actual functional website.

The website contains the following architecture:

- **Landing Page:**

  A 3-D interactive space UI made using 3 Js and React 3 Fiberwhich contains 3 options for the user to access the services.

- **Genre Detection Page:**

  A simple UI consisting of a dropbox to upload .wav file and another block to display the Genre(output).

- **Music Recommendation Page:**

A user can type or select a song from a list of songs and 5 songs will be recommended along with the album cover displayed with the help of Spotify API.

- **Community ChatRoom Feature:**

  An in-built chat community feature with a user login UI.

The Genre detection UI is implemented using the gradio API and the recommendation UI is implemented using Streamlit. The chatroom feature uses socket.io , Node.js, Express.js to make the server while the UI is made using HTML and CSS.

Chapter 4

Implementation

We have implemented the detection system using Python programming language and gradio. We have created and trained an ML model in Python employing **KNN (K-nearest neighbour)** method, which is a popular machine learning technique that is used for classification and regression tasks. Thus the model predicts the genre of the .wav file uploaded by the user and displays it through **Gradio**. **Gradio** is an open-source Python package allowing you to build a web application for ML models, APIs and more.

Similarly, the recommendation system is also equipped with an ML model trained on a sufficient dataset using Python. The user is prompted to type in a song or choose one from our collection. The resulting recommended songs are thus displayed with a poster and artist names along with the song name. The model employs **TFIDF (Term Frequency - Inverse Document Frequency)** which determines the importance of a term or word in a given text which in turn helps us in finding the important lines and verses in a song. On top of that, we use **Cosine Similarity** which aids in grouping similar words into a source word hence helping in finding similarities between different songs. This model has been deployed through **Streamlit** which is a popular open-source Python framework for machine learning used to deploy ML tools.

For the chatroom feature, we have used Socket.io which is a Real-time communication library used in chat features. We have implemented the web framework Express.js to create the chat server. The complete interface has 3 features namely:

1. The **joinRoom** event handler which allows users to join a specific chat room.
2. The **chatMessage** event handler which broadcasts messages to all users in the same room.
3. The **disconnect**— event handler that broadcasts a notification when a user leaves the room.

All the above mentioned features are connected together and implemented on a single landing page.

4.1 Preprocessing

Reading of data and creating a dataframe using **pandas**, a python library. When a user uploads an audio file, processing is done with the python library **librosa**. This processing includes retreiving the different variables of an audio file like beats, chroma stft, spectral centroid, spectral bandwidth, zero crossing rate, mfcc to name a few.A **RandomForestClassifier** can be used to highlight the importance of all these features in Model training.The **feature importance graph** is given below:

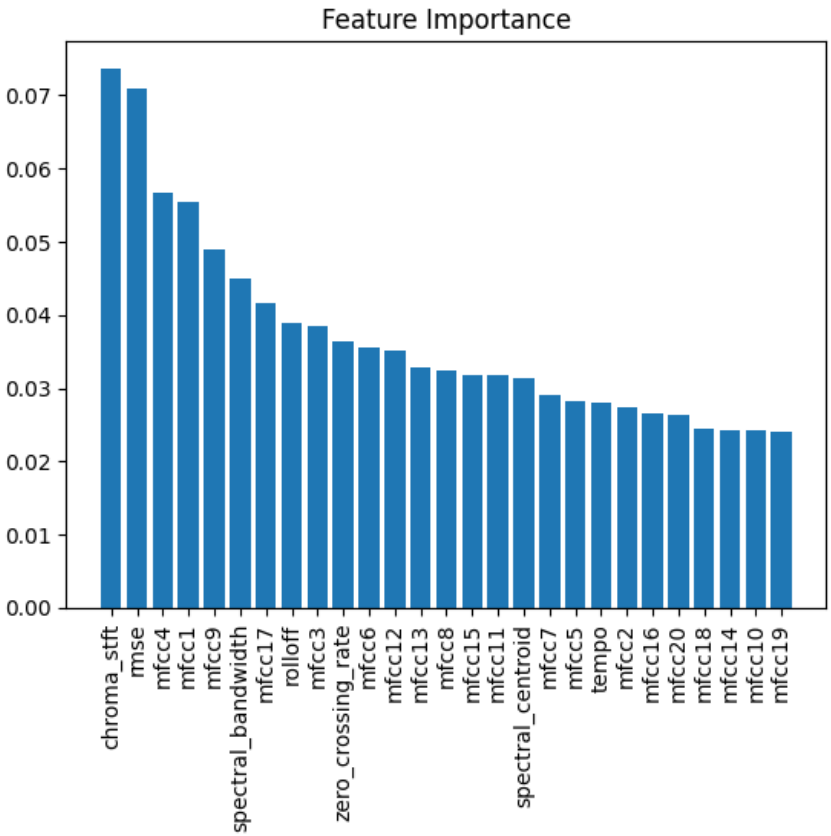

Figure 4.1: Feature Importance Graph

After fetching the audio features, splitting and scaling occurs which is done by the **Scikit-learn** module of librosa. Use of **MinMaxScaler** is done to scale the features.

4.2 Model Training

Once the features are scaled , the **KNN** (K Nearest Neighbours) model is trained using the co-ordinates of the scaled features with the help of **KNeighboursClassifier** module.

4.3 Displaying Results (API)

A function 'filetotext' is implemented which uses the knn model (.pkl file) to predict the genre and convert it to text. Once this is done, it sends the result to be displayed to the **Gradio** API. Gradio is a module in python used to create simple interfaces just like Streamlit. This is an alternative used in place of using Flask for backend so as to optimize the results and to increase the speed of the Backend processing. this will overall increase the user experience.

4.4 Frontend

The frontend of our project is made using **HTML, CSS** and **Javascript** to make the website lively and interactive.

Chapter 5

Results and Discussion

We have been able to get a decent **accuracy** (**72** on detection) on both models namely the detection and recommendation ones. Similarly, we have been able to implement the chat feature successfully after running the Express server on localhost and users on different ports.
The key takeaway from the project was how we were able to successfully train the model upto a decent level of accuracy using the dataset which may be inconsistent in some or the other form. Further, the challenges may arise in detection due to the quality of file being uploaded by the user .It may be sometimes difficult to accurately retrieve the features and thus may lead to wrong output being displayed.

Here are the results after running the projects:

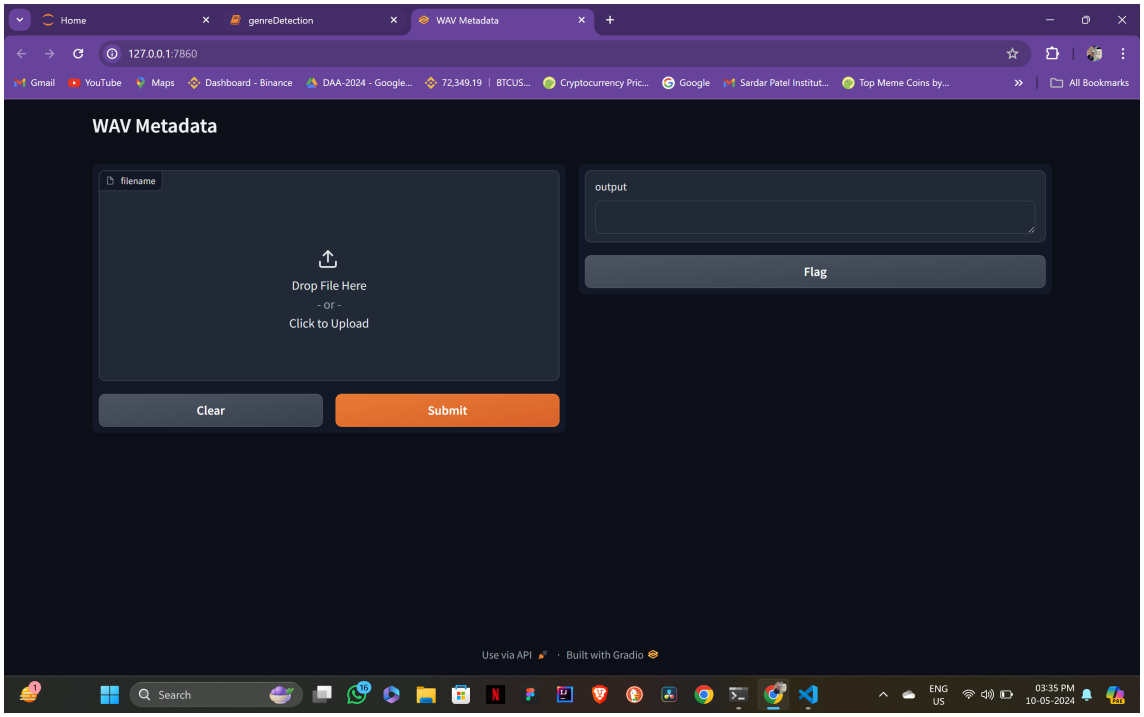

Figure 5.1: Genre Detection

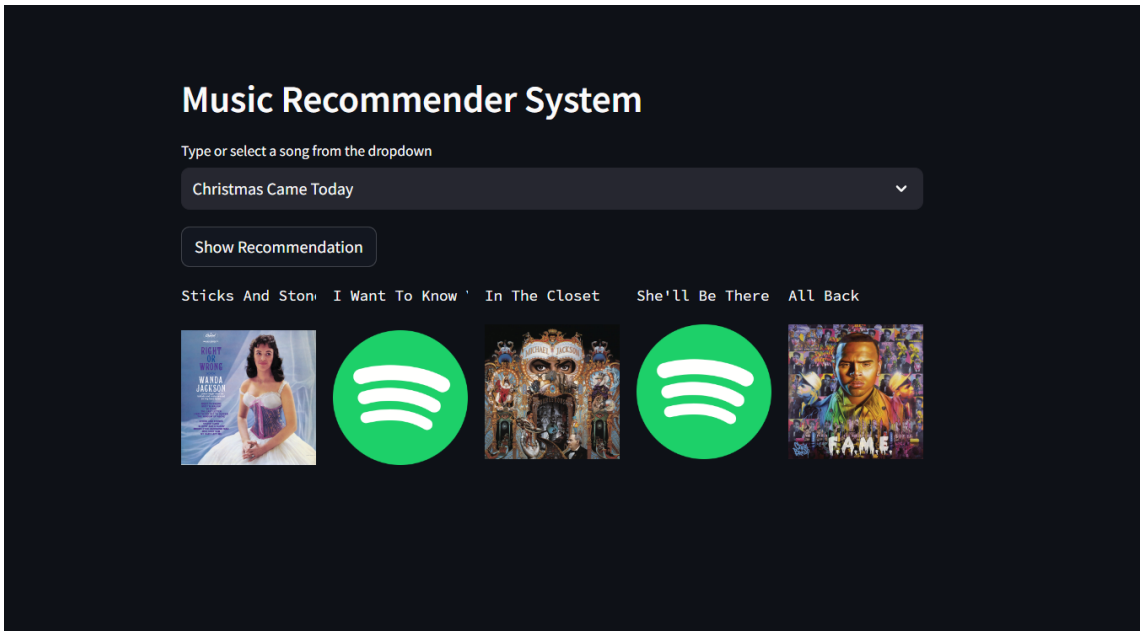

Figure 5.2: Music Recommendation

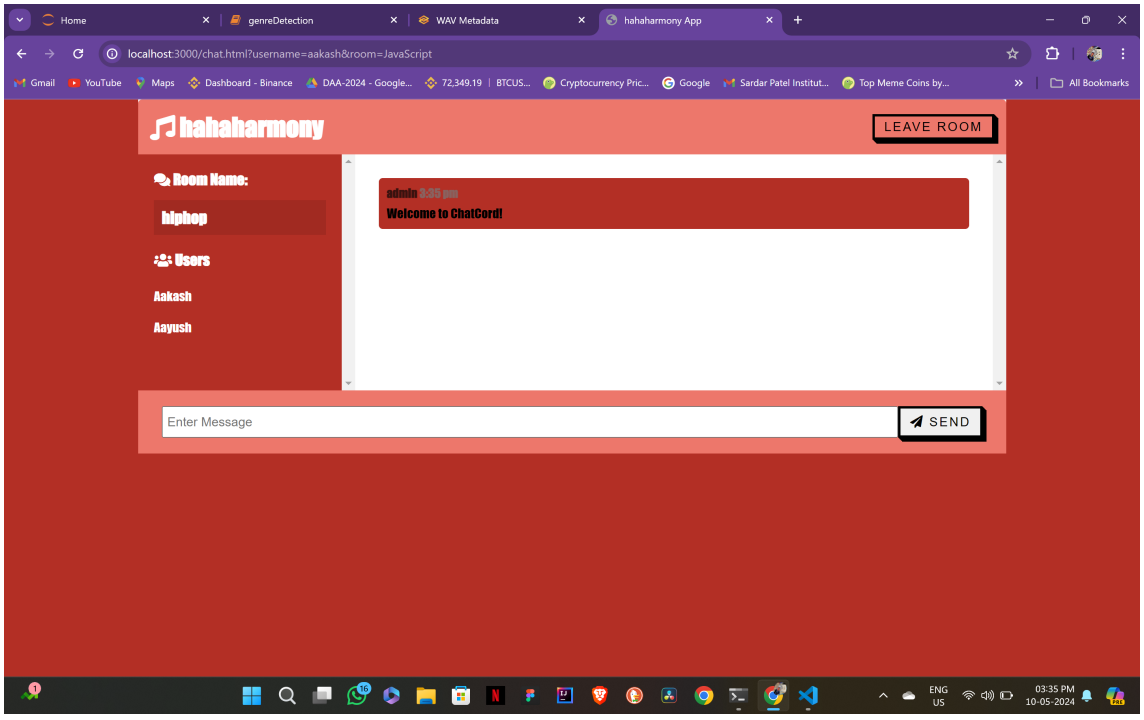

Figure 5.3: Chatroom Feature

Chapter 6

Conclusions

In conclusion, we have been able to predict the genre of the .wav file uploaded by the user done. This ML model has been trained a substantially bigger dataset and has been able to yield an acceptable accuracy in correctly determining the genre. The KNN(K-nearest neighbour) method employed has been able to yield decent accuracy. The processing time for the uploaded file is also quite low. The result is displayed on gradio for the user depicting the probable genre.

Similarly, the recommendation model has been able to, at most times, give close and relevant recommendations based on the file uploaded. We have used Spotify API for this and the model has been implemented on a dataset comprising of about 58,000 popular songs having a good mix of different music styles, countries and cultures. Since the accuracy of this feature cannot be quantitatively described we cannot talk about the probable accuracy in numbers for this feature. The resulting songs recommended are fetched as a list and displayed to the user withe the poster and artist names for so that the user can themself decide for the quality of recommendation.

Lastly, the chat feature has been implemented using socket.io that enables users who have similar taste in music to connect with each other and share their thoughts. The chatroom is allotted on the basis of the genre selected from our collection and the user is then added to the group-chat. No prior registration or sign up is required for the user to use this feature simply a name and genre shall be enough for access. Lastly, all these features have been deployed and can be reached by the user by a landing page created using HTML and CSS which has the option to access these features.

Chapter 7

Future Scope

The project is still quite naive and some features have been implemented using gradio and streamlit but not directly integrated into a web-page. Our future scope would be to create a dedicated website allowing these functionalities to be implemented independently into the website without external hosts. Additionally, we might add more functionalities wherein the user can listen to a snippet or the entire songs directly through our website. But since these goals are too ambitious without proper funding are not possible for us right now.

Furthermore, we can have users set up their accounts and store their liking, history, playlists, favorite genres, artists, etc. into a database accessible through a web /mobile application. Also the chat feature can be further developed to have users send and receive friend requests to connect with their friends and also have artists around connect and collaborate their work through this medium thus advancing the way music is produced and listened to on a large scale.

While some of these concepts are not possible right now we might be seeing a change in this domain and might just stand a chance to be a part of this ambitious advancement.

Bibliography

[1] Music Genre Classification using Machine Learning Techniques Hareesh Bahuleyan University of Waterloo, ON, Canada hpallika@uwaterloo.ca .

[2] **https://ieeexplore.ieee.org/abstract/document/9579813/Emotion based music recommendation system using LSTM-CNN architecture**

[3] Netflix Recommendation System based on TF-IDF and Cosine Similarity Algorithms Mohamed Chiny1 a , Marouane Chihab1 , Omar Bencharef2 b and Younes Chihab1 c 1Laboratory of Computer Sciences, Ibn Tofail University, Kenitra, Morocco 2Department of Computer Sciences, Cadi Ayyad University, Marrakesh, Morocco

[4] **https://ieeexplore.ieee.org/abstract/document/7849628/Design and implementation of web based real time chat interfacing server**