Music Genre Classification System

Submitted by

Arindam Das	(13000117122)
Anurag Ganguly	(13000117125)
Aniket Das	(13000117130)
Ananya Paul	(13000117131)

Under the guidance of Prof. Nairanjana Chowdhury

Submitted for the partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering

Techno Main Salt Lake,
EM 4/1, Salt Lake, Sector V, Kolkata – 700 091

CERTIFICATE

This is to certify that the project entitled "Music Genre Classification System" prepared by Arindam Das (13000117122), Anurag Ganguly (13000117125), Aniket Das (13000117130), Ananya Paul (13000117131) of B.Tech (Computer Science & Engineering), Final Year, has been done according to the regulations of the Degree of Bachelor of Technology in Computer Science & Engineering. The candidates have fulfilled the requirements for the submission of the project report.

It is to be understood that, the undersigned does not necessarily endorse any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

-- --

(Signature of the Internal Guide) (Signature of the HOD)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our mentor in the Department of Computer Science and Engineering, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he/she took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least, we convey our gratitude to all the teachers for providing us the technical skills that will always remain as our asset and to all non-teaching staffs for the gracious hospitality they offered us.

Place: Techno India, Salt Lake

Date: 09.07.2021

(Arindam Das)

(Anurag Ganguly)

(Aniket Das)

(Ananya Paul)

Contents

1 Introduction

1.1 Abstract

We aim to create a Music Genre Classification System that will predict the genre for any given musical track. The purpose of this work is to predict the genres of songs by using machine learning techniques. The system aims to find out the corresponding artist, recommending more of his or her songs. For this purpose, feature extraction is done by using signal processing techniques, then machine learning algorithms are applied with those features to do a multiclass classification for music genres. This project targets a high market demand owing to its high accuracy score and shorter segmentation period for the musical tracks. The expected project benefits are: creating a platform to access a personal choice of music, user friendly experience, and understanding the importance of teamwork.

1.2 Problem Domain

Our project pertains to a multi classification problem and a recommendation approach which we intend to design applying multiple Machine Learning algorithms, also including classes of Neural Network. Feature engineering comprises data pre-processing, data wrangling, and data visualization. Training and Testing a machine using the CNN Algorithm. Audio processing in which MFCC or Mel-Spectrogram is employed for Music Information Retrieval. The implementation of Deep Learning algorithms combined with Convolution Neural Network classes increases the computational cost by a significant margin. Many models based on similar domain are already floating in the market, posing a difficult challenge to establish our model

1.3 Related Studies

Owing to our project at hand, we have been looking through various publications from deemed universities as well as certain study reports. We have come to know that machine learning techniques have been used for music genre classification for decades now.
In July 2002, G. Tzanetakis and P. Cook used both the mixture of Gaussians model and k-nearest neighbors along with three sets of carefully hand-extracted features representing timbral texture, rhythmic content, and pitch content. They achieved 61% accuracy. As a benchmark, human accuracy averages around 70% for this kind of genre classification work. Tzanetakis and Cook used MFCCs, a close cousin of Mel-spectrograms, and essentially all work has followed in their footsteps in transforming their data in this manner.

In April 2003, methods such as support vector machines were also applied to this task, such as in 2003 when C. Xu et al. used multiple layers of SVMs to achieve over 90% accuracy on a dataset containing only four genres.
In the past 5-10 years, however, convolutional neural networks have been shown to be incredibly accurate music genre classifiers with excellent results reflecting both the

complexity provided by having multiple layers and the ability of convolutional layers to effectively identify patterns within images (which is essentially what mel-spectrograms and MFCCs are). These results have far exceeded human capacity for genre classification, with our research finding that current state-of-the-art models perform with an accuracy of around 91% when using the full 30s track length. Many of the papers that implemented CNNs compared their models to other ML techniques, including k-NN, mixture of Gaussians, and SVMs, and CNNs performed favorably in all cases. Therefore, we decided to focus our efforts on implementing a high-accuracy CNN, with other models used as a baseline.

1.4 Glossary

Acronyms	Expansion
AI	Artificial Intelligence
CNN	Convolution Neural Network
MFCC	Mel-Frequency Spectrograms
MIR	Music Information Retrieval
ML	Machine Learning
RBF	Radial Basis Function
EI	External Inputs
EO	External Output
EQ	External Inquiries
ILF	Internal Files
EIF	External Interface Files
UFT	Unadjusted Function Point
PC	Product Complexity adjustment
CAV	Complexity Adjustment Value
FP	Total adjusted Function Point

2 Problem Definition

2.1 Scope

Study Scope	**Application in our project**
Machine Learning	An application of **artificial intelligence** (AI) that provides systems the ability to automatically learn and improve from experience without being **explicitly** programmed.
Deep Learning	A subset of **ML** which will be used to learn, and can be **supervised, semi supervised** or **unsupervised.**
Neural Network	We will employ **spectrograms** as inputs of the network to train the model for the **multi classification** approach.
Feature Engineering	Processing of raw dataset implementing **data wrangling** and **data visualization** for training our model.
Data Wrangling	**Simplification** and cleaning of the messy and **complex** dataset.
Fast F ourier Transform	Algorithm that will compute the discrete **Fourier transform** of a sequence by converting a signal from its original domain**(time/space)** to a representation in the **frequency** domain and vice versa.

2.2 Exclusions

We have only considered .wav music files as our input. Any other file format is excluded from the input range.

2.3 Assumptions

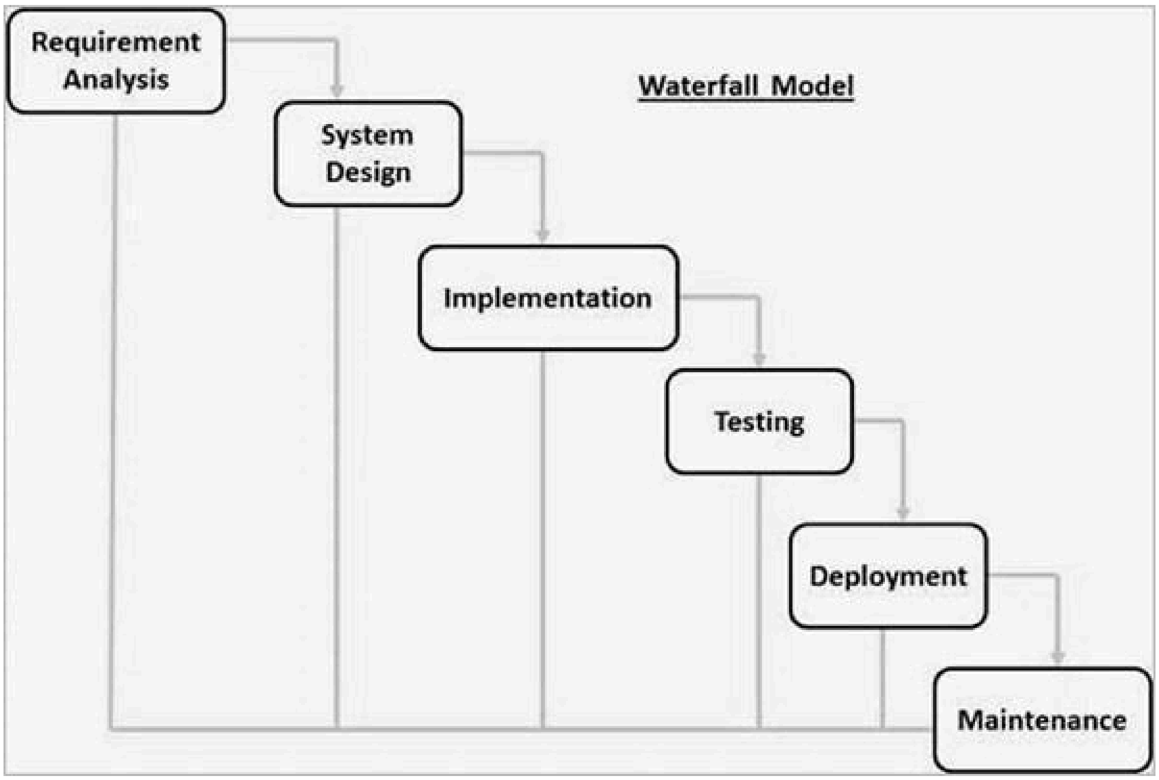
i. The length of the music file has been assumed to be at least 30 seconds for audio segmentation of data.
ii. Only one musical file format is considered (.wav).
iii. Only a set of predefined genres has been considered for our project.

3 Project Planning

For **Project Planning/schedule**, please paste suitably from MS Project. A Gantt chart should be shown for major phases with highlighted milestones.

3.1 Software Life Cycle Model

We have chosen the Waterfall model as the software life cycle model of our project.

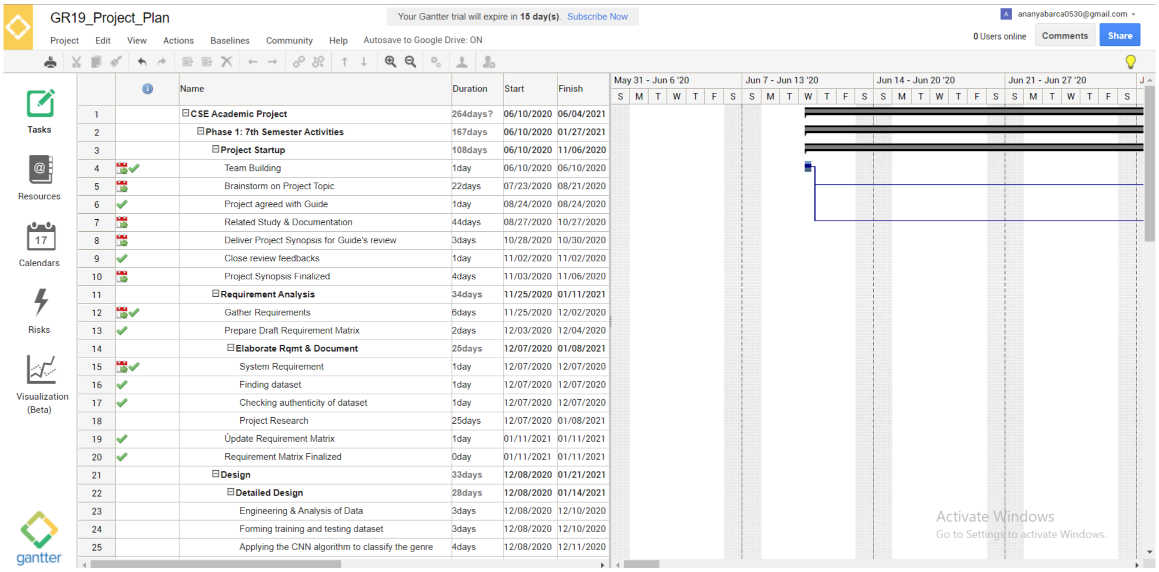


The main advantages of this model are as mentioned below:

i. Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
ii. Phases are processed and completed one at a time, and it's easier to arrange the tasks.

It has clearly defined stages, and the process and results are well documented

3.2 Scheduling



3.3 Cost Analysis

For the Cost Analysis part of our project, we have used Functional Point Analysis.

A cost analysis (also known as a benefit cost analysis) is a process by which organizations can analyze decisions, systems, or projects, or determine the value for intangibles. The model is built by identifying the benefits of an action as well as the associated costs, and subtracting the costs from the benefits. When completed, a cost benefit analysis will yield concrete results that can be used to develop reasonable conclusions around the feasibility.

Why Use Cost Analysis?

Organizations rely on cost analysis to support decision making because it provides an agnostic, evidence-based view of the issue being evaluated, without the influences of opinion, politics, or bias. By providing an unclouded view of the consequences of a decision, cost benefit analysis is an invaluable tool in developing business strategy, evaluating a new hire, or making resource allocation or purchase decisions.

Counting Function Point

Function units	Low	Average	High	Selection
EI	3	4	6	4
EO	4	5	7	5

EQ	3	4	6	4
ILF	7	10	15	10
EIF	5	7	10	7

Unadjusted function point

Function units	Count	Low	Average	High	Selection	Total (Count*selection)
EI	5	3	4	6	4	20
EO	5	4	5	7	5	25
EQ	3	3	4	6	4	12
ILF	8	7	10	15	10	80
EIF	2	5	7	10	7	14

UFT = 20+25+12+80+14
= 151

Complexity adjustment value

Scale

0 - No Influence

1 - Incidental

2 - Moderate

3 - Average

4 - Significant

5 - Essential

Serial No.	**Factors**	**Scale**
1	Does the system require reliable backup and recovery?	4
2	Are data communications required?	4
3	Are there distributed processing functions?	3
4	Is performance critical?	4
5	Will the system run in an existing, heavily utilized operational environment?	4
6	Does the system require on-line data entry?	2
7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	2
8	Are the master files updated on-line?	2
9	Are the inputs, outputs, files, or inquiries complex?	3
10	Is the internal processing complex?	4
11	Is the code designed to be reusable?	4
12	Are conversion and installation included in the design?	4
13	Is the system designed for multiple installations in different organizations?	5
14	Is the application designed to facilitate change and ease of use by the user?	4
	Total complexity adjustment value =	49

PC = 0.65 +0.0.1*CAV
= 0.65 + 0.01*49
= 1.14

FP = PC*UFP

= 1.14*151

= 172.14

4 Requirement Analysis

4.1 Requirement Matrix

A snippet of the requirement matrix is given below:

Grp19_Music_Genre_Classification_RM_Requirements Matrix .XLSX

	Rqmt ID	Requirement Item	Requirement Status	Design Module	Design Reference (section# under project Report)	Test Case Number	Technical Platform of Implementation	Prototype prepared ?	Name of Program / Component	Test Results Reference	Additional Com included in previ
11	FE-2.4	Processing of spectral images	Completed	FE	6.3.2.4	T-FE-2.4	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
12	TRTS-3	Forming training and testing dataset	Completed	TRTS	6.3.3	T-TRTS-3	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
13	TRTS-3.1	Splitting the preprocessed data into training and testing dataset	Completed	TRTS	6.3.3.1	T-TRTS-3.1	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
14	TRTS-3.2	Manipulation of audio data	Completed	TRTS	6.3.3.2	T-TRTS-3.2	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
15	TRTS-3.3	Applying the fundamental ML algorithms to compare the accuracies of the model	Completed	TRTS	6.3.3.3	T-TRTS-3.3	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
16	TRTS-3.4	Noting down the highest accuracy before the application of KNN and CNN algorithr	Completed	TRTS	6.3.3.4	T-TRTS-3.4	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
17	CNN-4	Applying the CNN algorithm to classify the genre	Completed	CNN	6.3.4	T-CNN-4	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
18	CNN-4.1	Starting with an input image, apply filters to create a feature map	Completed	CNN	6.3.4.1	T-CNN-4.1	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
19	CNN-4.2	Apply ReLU function to increase non-linearity	Completed	CNN	6.3.4.2	T-CNN-4.2	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
20	CNN-4.3	Apply a pooling layer to each feature map	Completed	CNN	6.3.4.3	T-CNN-4.3	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
21	CNN-4.4	Flatten the pooled images into one long vector, input the vectors into CNN	Completed	CNN	6.3.4.4	T-CNN-4.4	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
22	CNN-4.5	Processes the feature through the network	Completed	CNN	6.3.4.5	T-CNN-4.5	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
23	CNN-4.6	Trains through forward propagation and backpropagation for many epochs	Completed	CNN	6.3.4.6	T-CNN-4.6	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
24	SEQ-5	Applying Sequential Model algorithm on our dataset	Completed	SEQ	6.3.5	T-SEQ-5	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
25	SEQ-5.1	Importing TensorFlow	Completed	SEQ	6.3.5.1	T-SEQ-5.1	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
26	SEQ-5.2	Designing the multiple layers of the Sequential Model	Completed	SEQ	6.3.5.2	T-SEQ-5.2	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
27	SEQ-5.3	Compiling the model with the desired loss function and optimizer	Completed	SEQ	6.3.5.3	T-SEQ-5.3	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
28	SEQ-5.4	Training and testing the model	Completed	SEQ	6.3.5.4	T-SEQ-5.4	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
29	CMP-6	Comparison B/W models and finding the best algorithm for our classification system	Completed	CMP	6.3.6	T-CMP-6	Google Colab	Yes	MusicModel_v3.ipynb	MusicModel_v3.ipynb	
30	UI-7	Developing User Interface platform using Flask	Completed	UI	7	T-UI-7	Visual Studio Code	Yes	flaskapp.py	flaskapp.py	
31	M-8	Incorporating the user interface and backend algorithms into the platform	Completed	M	8	T-M-8	Visual Studio Code	Yes	prediction.html	prediction.html	

4.2 Requirement Elaboration

4.2.1 Preprocessing the music dataset:

i. Verify the authenticity of the music file format (.wav)
ii. Loading of raw dataset
iii. Cleaning, structuring, and enriching raw data into a desired format
iv. Grouping of similar data based on the chosen parameter

4.2.2 Engineering and Analysis of music data:

i. Decomposition of pre-processed data into its constituent frequencies
ii. Spectral Analysis of the frequencies
iii. Construction of MFCCs (Mel-Frequency Spectrograms)
iv. Processing of spectral images

4.2.3 Forming training and testing dataset:

i. Splitting the preprocessed data into training and testing dataset
ii. Manipulation of audio data
iii. Applying the fundamental ML algorithms to compare the accuracies of the model
iv. Noting down the highest accuracy before the application of Sequential and CNN algorithms

4.2.4 Applying CNN model to classify the genre:

i. Starting with an input image, apply filters to create a feature map
ii. Apply ReLU function to increase non-linearity
iii. Apply a pooling layer to each feature map
iv. Flatten the pooled images into one long vector, input the vectors into CNN
v. Processes the feature through the network
vi. Trains through forward propagation and backpropagation for many epochs
vii. Running the algorithm and finding out the desired output

4.2.5 Applying Sequential classification algorithm on our dataset:

i. Importing TensorFlow package
ii. Compiling and fitting the model with the dataset
iii. Running the algorithm to find out the desired output

4.2.6 Comparison between models and finding the best algorithm for our classification system

We have selected the sequential model for our platform as it gives out an accuracy of 68%. The CNN 1-D model gives us an accuracy of 60%, and hence it was discarded for the final scratch.

5 Design

5.1 Technical Environment

The minimum hardware requirements for our project are mentioned below:

- ✔ RAM – 8GB
- ✔ Cache memory – 4MB
- ✔ Clock frequency – 3.4 GHz
- ✔ Graphics memory – 2GB
- ✔ SSD storage – 500GB

The software requirements for our project have been mentioned below:

- ✔ OS – Windows 10/ Ubuntu 18.04
- ✔ Programming Language: Python 3, HTML, CSS, JavaScript
- ✔ Tools: Google Colab
- ✔ Dataset: Microsoft Excel(.csv)

5.2 Detailed Design

From the Flowchart given above, the explanation of the design is divided into several modules, which will be required to implement the music genre classification.

5.3.1 Preprocessing the music dataset

Data Preprocessing is a technique that is used to convert the raw data into a clean dataset. In other words, whenever the data is gathered from different sources, it is collected in raw format, which is not feasible for the analysis. In this module, we are verifying the authenticity of the music file format (.wav), loading the raw dataset, cleaning, structuring, and enriching raw data into a desired format, and grouping up of similar data based on chosen parameters.

5.3.2 Engineering and Analysis of music data

In this module, we will be implementing Feature engineering, which is extracting relevant features from the dataset. So, the first step is decomposition of pre-processed data into its constituent frequencies, which is then followed by spectral analysis of the frequencies. Then the process of construction of MFCCs (Mel-Frequency Cepstral Coefficients) is followed by processing of spectral images.

5.3.3 Forming training and testing dataset

The splitting of the dataset into training and testing dataset is done, which will be used in incorporating fundamental ML algorithms and checking their accuracies before proceeding to the Sequential Model and the Deep Learning Algorithms -> CNN.

5.3.4 Applying CNN classification algorithm to our genre

One of the two algorithms is the CNN (Convolutional Neural Network). In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics. Starting with an input image, apply filters to create a feature map. Apply ReLU function to increase non-linearity. Apply a pooling layer to each feature map and flatten the pooled images into one long vector, input the vectors into CNN model and process the feature through the network. Then the model is trained through forward propagation and backpropagation for many epochs.

5.3.5 Applying Sequential classification algorithm to our genre

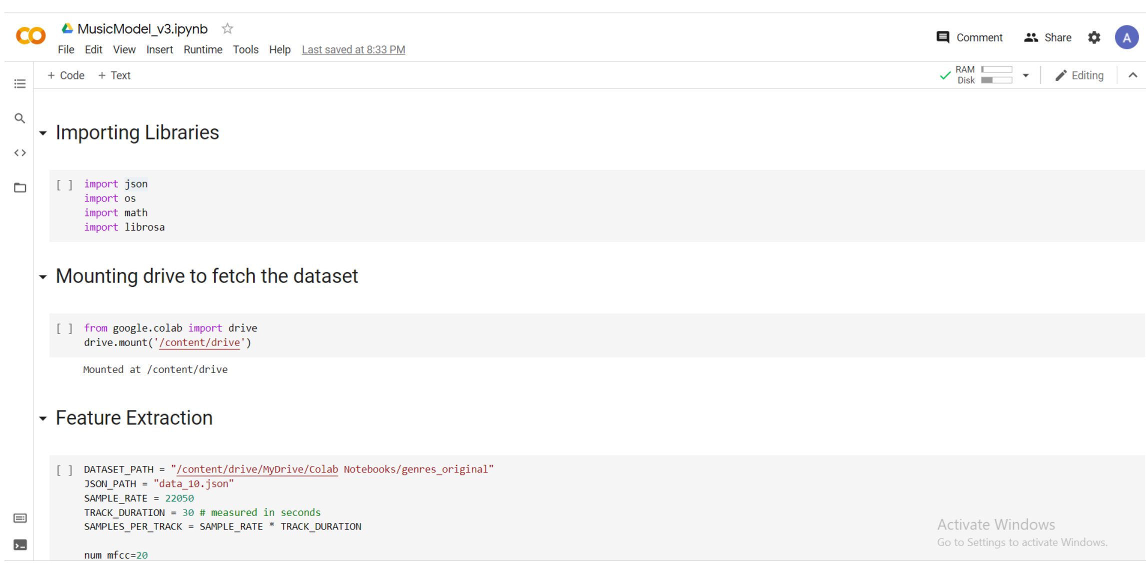
The other algorithm is the Sequential model, which will be applied to find the classification of the musical data. So, the basic steps are to compile and fit the algorithm on our dataset to find out the desired output for a user file.

5.3.6 Comparison between models and finding the best algorithm

We have selected the sequential model for our platform as it gives out an accuracy of 68%. The CNN 1-D model gives us an accuracy of 60%, and hence it was discarded for the final scratch.

6 Implementation

6.1 Implementation Details



MusicModel_v3.ipynb

File Edit View Insert Runtime Tools Help Last saved at 8:33 PM

```
n_fft=2048
hop_length=512
num_segments=10
```

```
data = {
    "mapping": [],
    "labels": [],
    "mfcc": []
}
```

```
samples_per_segment = int(SAMPLES_PER_TRACK / num_segments)
num_mfcc_vectors_per_segment = math.ceil(samples_per_segment / hop_length)
```

```
# loop through all genre sub-folder
for i, (dirpath, dirnames, filenames) in enumerate(os.walk(DATASET_PATH)):

    if dirpath is not DATASET_PATH:

        semantic_label = dirpath.split("\\")[-1]
        data["mapping"].append(semantic_label)
        print("\nProcessing: {}".format(semantic_label))

        for f in filenames:

            file_path = os.path.join(dirpath, f)
            try:
                signal, sample_rate = librosa.load(file_path, sr=SAMPLE_RATE)
            except:
                continue

            for d in range(num_segments):
```

MusicModel_v3.ipynb

File Edit View Insert Runtime Tools Help Last saved at 8:33 PM

```
import json
with open("data_final.json", "r") as fp:
    data = json.load(fp)
```

```
import numpy as np
x=np.array(data["mfcc"])
x1=np.array(data["mfcc"])
```

```
y=np.array(data["labels"])
y1=np.array(data["labels"])
```

Splitting dataset into training and testing set

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
x_train_1,x_test_1,y_train_1,y_test_1 = train_test_split(x1,y1,test_size=0.3)
```

Defining and Designing the Sequential model

```
#Sequential

import tensorflow.keras as keras
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(x.shape[1],x.shape[2])),
```

MusicModel_v3.ipynb

File Edit View Insert Runtime Tools Help Last saved at 8:33 PM

```
Epoch 30/50
219/219 [==============================] - 12s 54ms/step - loss: 0.3442 - accuracy: 0.9019
```

```
y_pred = model.predict(x_test)
```

```
model.evaluate(x_test,y_test,batch_size=32)
```

```
94/94 [==============================] - 1s 11ms/step - loss: 2.5620 - accuracy: 0.6953
[2.5619964599609375, 0.6952603459358215]
```

Defining and Designing the CNN Model

```
#CNN
import tensorflow.keras as keras
import keras
from keras.preprocessing import sequence
from keras.models import Sequential
from keras.layers import Dense, Embedding
from keras.layers import Input, Flatten, Dropout, Activation
from keras.layers import Conv1D, MaxPooling1D
from keras.models import Model

model2 = Sequential()

model2.add(Conv1D(120,5, padding = 'same',input_shape=(x1.shape[1],x1.shape[2]),activation = 'relu'))
model2.add(MaxPooling1D(pool_size=(6), padding = 'same'))
model2.add(Conv1D(120,5, padding = 'same' ,activation = 'relu'))
model2.add(Dropout(0.2))
model2.add(Flatten())
model2.add(Dense(10, activation = 'softmax'))
```

MusicModel_v3.ipynb

File Edit View Insert Runtime Tools Help Last saved at 8:33 PM

Saving the hierarchy file for the two models -> Sequential and CNN

```
model.save("/content/drive/MyDrive/Colab Notebooks/MusicModel_v3_model.h5")
model2.save("/content/drive/MyDrive/Colab Notebooks/MusicModel_v3_model_2.h5")
```

```
signal, sample_rate = librosa.load("/content/drive/MyDrive/Colab Notebooks/genres_original/disco/disco.00031.wav", sr=SAMPLE_RATE)
```

```
lst = []

for d in range(num_segments):

                    start = samples_per_segment * d
                    finish = start + samples_per_segment

                    # extract mfcc
                    mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_length)
                    mfcc = mfcc.T

                    # store only mfcc feature with expected number of vectors
                    if len(mfcc) == num_mfcc_vectors_per_segment:
                        lst.append(mfcc.tolist())
                        print("{}, segment:{}".format("/content/drive/MyDrive/Colab Notebooks/genres_original/disco/disco.00031.wav", d+1))
```

```
/content/drive/MyDrive/Colab Notebooks/genres_original/disco/disco.00031.wav, segment:1
/content/drive/MyDrive/Colab Notebooks/genres_original/disco/disco.00031.wav, segment:2
/content/drive/MyDrive/Colab Notebooks/genres_original/disco/disco.00031.wav, segment:3
/content/drive/MyDrive/Colab Notebooks/genres_original/disco/disco.00031.wav, segment:4
/content/drive/MyDrive/Colab Notebooks/genres_original/disco/disco.00031.wav, segment:5
```

MusicModel_v3.ipynb

File Edit View Insert Runtime Tools Help Last saved at 8:33 PM

Loading the two models for testing the output

```
var = keras.models.load_model("/content/drive/MyDrive/Colab Notebooks/MusicModel_v3_model.h5")
var1 = keras.models.load_model("/content/drive/MyDrive/Colab Notebooks/MusicModel_v3_model_2.h5")
```

```
final = np.argmax((var.predict(x) > 0.5).astype("int32"))
final_2 = np.argmax((var1.predict(x1) > 0.5).astype("int32"))
final_2 = final_2-(10*(final_2//10))
```

MusicModel_v3.ipynb

File Edit View Insert Runtime Tools Help Last saved at 8:33 PM

Printing the output for a user input

```
print("Using Linear model: Label of genre->", final)
print("Using CNN model: Label of genre->", final_2)
```

```
Using Linear model: Label of genre-> 3
Using CNN model: Label of genre-> 2
```

```
if final == 0:
  print(" The predicted genre is classical")
elif final == 1:
  print("The predicted genre is blues")
elif final == 2:
  print("The predicted genre is reggae")
elif final == 3:
  print("The predicted genre is disco")
elif final == 4:
  print("The predicted genre is rock")
elif final == 5:
  print("The predicted genre is jazz")
elif final == 6:
  print("The predicted genre is country")
elif final == 7:
  print("The predicted genre is pop")
elif final == 8:
  print("The predicted genre is hiphop")
else:
  print("The predicted genre is metal")
```

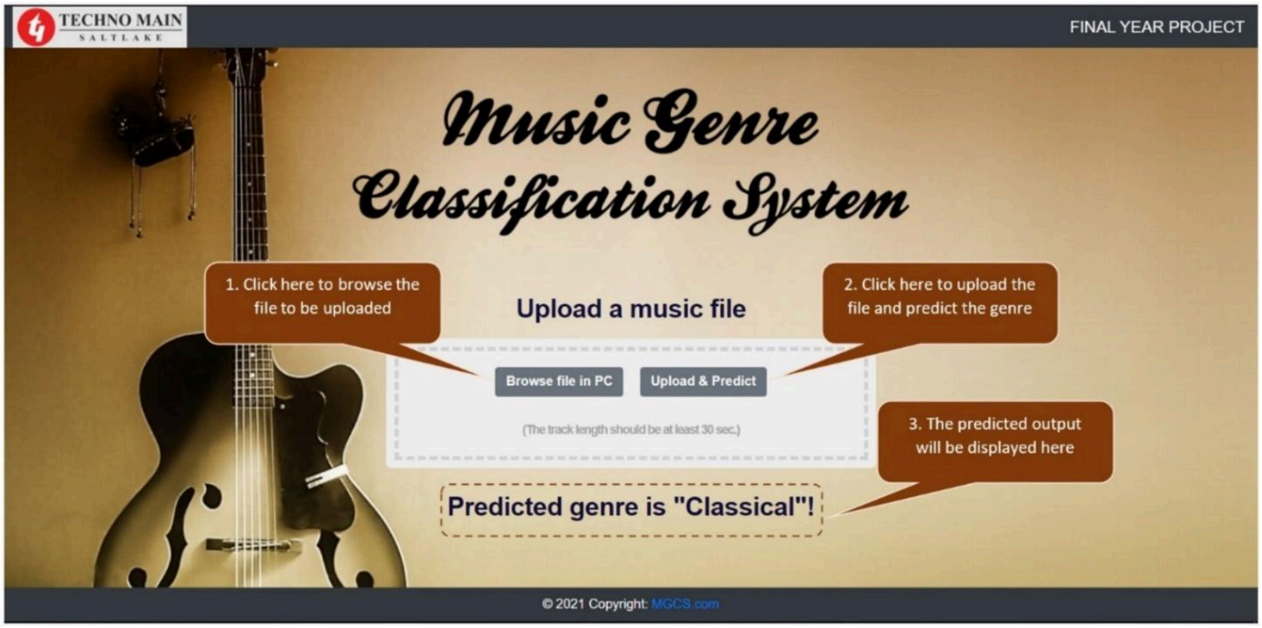
```
The predicted genre is disco
```

6.2 System Installation Steps

System Installation Steps:

1. As we are writing our code in python, we are using Google Colab as our editor and compiler.
2. The Music Genre Classification System has been compiled and executed in Google Colab.
3. Different modules have been imported like librosa (for musical analysis), json, os and math, numpy, tensorflow, keras for CNN and Sequential models, and the pre-processing libraries from keras modules.
4. The dataset (.wav musical files) has been uploaded to Drive and then accessed in Google Colab and mounted to the code.
5. Pre-processing the data and feature extraction has been implemented, compiled, and executed successfully.
6. In the next step, we are importing the required modules and splitting the dataset.
7. After splitting the dataset, we are training and testing the dataset.
8. Our two major models, CNN (1D) and Sequential model, have been employed, and the models have been saved in .h5 format for further use.
9. Our two major models, CNN (1D) and Sequential model, have been compiled and executed successfully with the desired accuracy, giving 68% for the Sequential model and 60% for the CNN model.
10. Next, as for our User Interface, we are using Flask as the framework (API of Python for building web applications) for the HTML Page for our Music Genre Classification System.
11. The preferred .h5 model (Sequential Model) file has been uploaded to the Flask implementation code and merged with the HTML code.
12. The implementation code for Flask and HTML has been written in Visual Studio.
13. Now for the main execution, in the HTML page, the necessary music file is uploaded, and the corrected genre has been predicted as per our accuracy.
14. We have also taken other music files from outside the dataset to determine the accuracy.

6.3 System Usage Instructions



7 Test Results and Analysis

Serial No.	***Test ID***	***Test Case***	***Expected Result***	***Observed Result***	***Status***
1	T-DTP-1	Preprocessing the music dataset	Preprocessed music dataset	Preprocessed music dataset	Passed
2	T-DTP-1.1	Verify the authenticity of the music file format (.wav)	.wav music file	.wav music file	Passed
3	T-DTP-1.2	Loading of raw dataset	Dataset loaded	Dataset loaded	Passed
4	T-DTP-1.3	Cleaning, structuring, and enriching raw data into a desired format	Wrangled data	Wrangled data	Passed
5	T-DTP-1.4	Grouping of similar data based on the chosen parameter	Grouped data	Grouped data	Passed
6	T-FE-2	Engineering and Analysis of music data	Analysed music data	Analysed music data	Passed
7	T-FE-2.1	Decomposition of	Segmented	Segmented	Passed

		pre-processed data into its constituent frequencies	data	data	
8	T-FE-2.2	Spectral Analysis of the frequencies	Audio analysed segmented data	Audio analysed segmented data	Passed
9	T-FE-2.3	Construction of MFCCs (Mel-Frequency Spectrograms)	MFCC constructed	MFCC constructed	Passed
10	T-TRTS-3	Forming training and testing dataset	Trained & tested dataset	Trained & tested dataset	Passed
11	T-TRTS-3.1	Splitting the preprocessed data into training and testing dataset	Split dataset	Split dataset	Passed
12	T-TRTS-3.2	Manipulation of audio data	Manipulated audio data	Manipulated audio data	Passed
13	T-TRTS-3.3	Applying the fundamental ML algorithms to compare the accuracies of the model	Accuracy should be less than 35%	Accuracy less than 35%	Passed
14	T-TRTS-3.4	Noting down the highest accuracy before the application of Sequential and CNN algorithms	Accuracy should be less than 35%	Accuracy less than 35%	Passed
15	T-CNN-4	Applying the CNN algorithm to classify the genre	CNN should work with the dataset	CNN works with the dataset	Passed
16	T-CNN-4.1	Apply ReLU function to increase non-linearity	Non-linearity should be increased in the CNN	Non-linearity increases in the CNN	Passed
17	T-CNN-4.2	Apply a pooling layer to each feature map	The feature map should pooled	The feature map gets pooled	Passed
18	T-CNN-4.3	Flatten the pooled images into one long vector, and input the vectors into CNN	Flattened pooled feature maps	Flattened pooled feature maps	Passed
19	T-CNN-4.4	Processes the feature through the network	Preprocessed feature	Preprocessed feature	Passed

20	T-CNN-4.5	Trains through forward propagation and backpropagation for many epochs	Successful iterations of epochs	Successful iterations of epochs	Passed
21	T-SEQ-5	Applying Sequential Model algorithm on our dataset	Sequential Model should work with the dataset	Sequential Model works with the dataset	Passed
22	T-SEQ-5.1	Importing TensorFlow	TensorFlow package imported	TensorFlow package imported	Passed
23	T-SEQ-5.2	Designing the multiple layers of the Sequential Model	Multiple layers are formed	Multiple layers are formed	Passed
24	T-SEQ-5.3	Compiling the model with the desired loss function and optimizer	Loss function and optimizer should be working correctly	Loss function and optimizer work correctly	Passed
25	T-SEQ-5.4	Training and testing the model	Music genre with an accuracy of >60%	Music genre with an accuracy of >60%	Passed
26	T-CMP-6	Comparison B/W models and finding the best algorithm for our classification system	Sequential model should be the best algorithm	Sequential model is the best algorithm	Passed
27	T-UI-7	Developing User Interface platform using Flask	Output should correctly flow in the console.	Output correctly flows in the console.	Passed
28	T-M-8	Incorporating the user interface and backend algorithms into the platform	A working UI	A working UI	Passed

8 Conclusion

8.1 Project Benefits

The project benefits are as follows:

i. Creating a platform to classify different songs on the basis of their genre.
ii. Incorporating the model into a user-friendly system where genre classification is deemed a priority

8.2 Future Scope for Improvements

Our future scope entails the inclusion of more genres, varied formats of musical files, and less segmentation time in terms of processing, though keeping the accuracy of the model intact.
We also aim to increase the accuracy of our prediction through the usage of more advanced algorithms and paradigms.

9 References / Bibliography

i. Music Genre Classification: Matthew Creme, Charles Burlin, Raphael Lenain, Stanford University, December 15, 2016.
ii. Mingwen Dong. Convolutional neural networks achieve human-level accuracy in music genre classification. CoRR, abs/1802.09697, 2018.
iii. Music Genre Classification: Derek A. Huang, Arianna A. Serafini, Eli J. Pugh.
iv. https://arxiv.org/ftp/arxiv/papers/1803/1803.04652.pdf
v. http://cs229.stanford.edu/proj2018/report/21.pdf
vi. Music Genre Classification using Machine Learning Techniques by Hareesh Bahuleyan(University of Waterloo).
vii. G. Tzanetakis and P. Cook. Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10(5):293–302, July 2002.
viii. Changsheng Xu, N. C. Maddage, Xi Shao, Fang Cao, and Qi Tian. Musical genre classification using support vector machines. In 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP ’03)., volume 5, pages V–429, April 2003.
ix. Music Genre Classification: Robert Adragna, Yuan Hong (Bill) Son.