Music Genre Classification With Deep Learning

Name :	ID :
Fhamid Mottaki Aurnob	170204058
Mohammad Najrul Islam	170204061
Rafsan Habib	170204069

Introduction

We've all listened to music on a streaming app. Music Genre Classification System is one way to demonstrate reasoning. Classifying music based on genre is a necessary first step in creating an effective recommendation system. Sound files will be handled in Python, audio features will be calculated from them, Machine Learning Algorithms will be used to see the outcomes, and so on. The main goal is to construct a machine learning model that categorizes music samples into distinct genres in a more systematic manner. Its goal is to forecast the genre based on the input, which is an audio file. The goal of automating music classification is to make it easier and faster to choose tunes. Manually classifying songs or music necessitates listening to a large number of tracks before deciding on a category. This takes a lot of time and effort. By automating the categorizing of music, useful information like trends, popular genres, and performers can be found more quickly and conveniently knowing what types of music you like is the first step towards figuring this out. In our project, we have implemented Music Genre Classification by using Long Short Term Memory(LSTM). For this project, the dataset that we will be working with is GTZAN Genre Classification dataset which consists of 1,000 audio tracks, each 30 seconds long[2]. It contains 10 genres, each represented by 100 tracks.Also we showed performance comparison by tuning hyper parameters to increase the accuracy. We were able to train a model that can classify music from 10 different genres by using LSTM. Our task was challenging because at first our model gave the accuracy 33%. Our main challenge was to increase the model accuracy.

Motivation

The motivation of this project is to build a music genre classification system for apps that let us stream music or buy songs of our preference. Since the music industry is a crucial one that makes tons of money and music is also a favorite pastime among both the older generation and the younger ones but the taste and preference of music varies from person to person, it's instrumental to build a system that can classify music based on its genre, based on which a recommendation system can be built.

People rely on music to spend their leisure time but not everyone's choice is the same, it varies from person to person. An app suggesting all mixed up genres of music to everyone is not going to perform well financially or compared to the demand of the market and the users. This is why an effective recommendation system of a music genre which can be classified properly is important.

Methodology

In our proposed Methodology, we have first collected the audio files from GTZAN Music Dataset. From the dataset, we pre-processed the audio files. After that, we tried to visualize the audio files. After doing so, we applied a feature extraction process, because the model can not run text values for that we converted it to numerical values. Then, we trained our model by using an RNN model consisting of

LSTM layers. Finally, we have evaluated the performance by using Confusion Matrix, also calculated the Precision, Recall and F1-score.

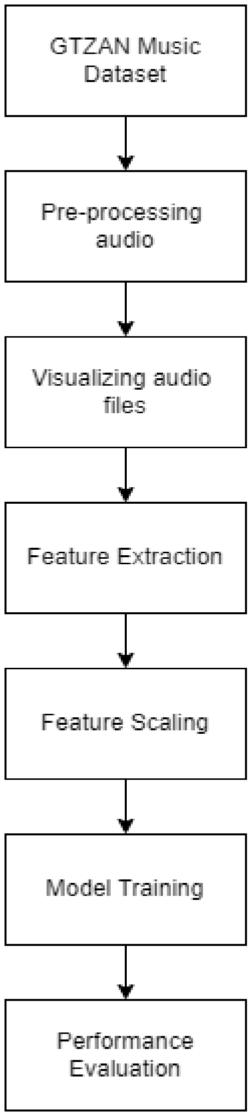


Fig : Workflow diagram of the RNN model

A. Understanding Audio Files

Librosa is a python package for music and audio analysis. It provides the building blocks necessary to create music information retrieval systems. By using Librosa, we can extract certain key features from the audio samples such as Tempo, Chroma Energy Normalized, Mel-Frequency Cepstral Coefficients, Spectral Centroid, Spectral Contrast, Spectral Rolloff and Zero Crossing Rate. For that, we used Librosa to understand audio files type, by taking sample files from the dataset[2].

B. Visualizing Audio Files

Waveforms are visual representations of sound as time on the x-axis and amplitude on the y-axis. They are great for allowing us to quickly scan the audio data and visually compare and contrast which genres might be more similar than others.

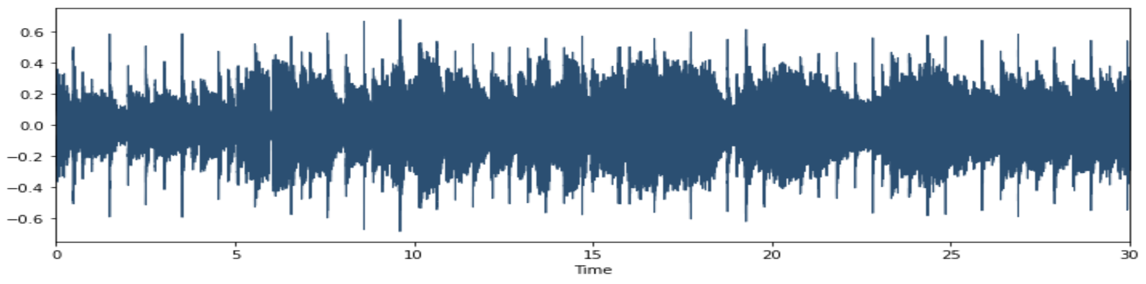
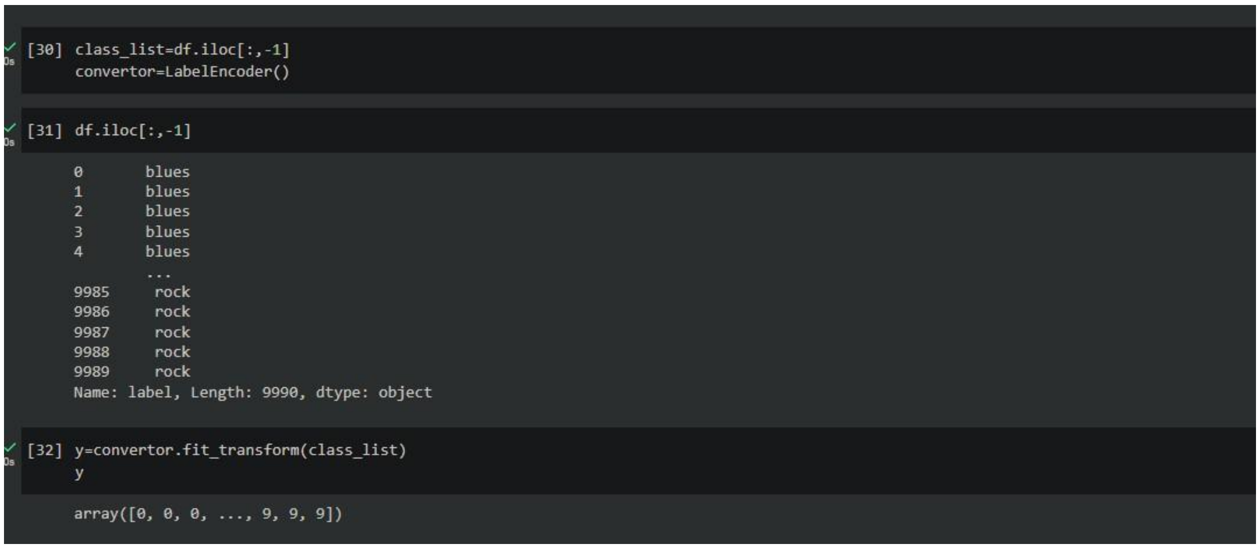


Fig: Raw waves of Samples

C. Feature Extraction

Preprocessing of data is required before we finally train the data. We will try and focus on the last column that is ‘label’ and will encode it with the function LabelEncoder() of Sklearn.preprocessing.

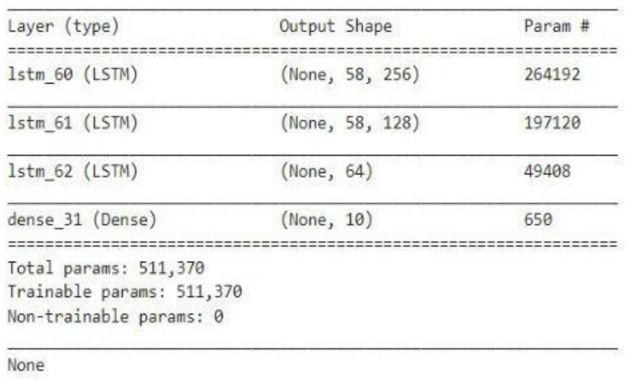


```
[30] class_list=df.iloc[:,-1]
     convertor=LabelEncoder()

[31] df.iloc[:,-1]

     0       blues
     1       blues
     2       blues
     3       blues
     4       blues
             ...
     9985     rock
     9986     rock
     9987     rock
     9988     rock
     9989     rock
     Name: label, Length: 9990, dtype: object

[32] y=convertor.fit_transform(class_list)
     y

     array([0, 0, 0, ..., 9, 9, 9])
```

We can't have text in our data if we're going to run any kind of model on it. So before we can run a model, we need to make this data ready for the model. To convert this kind of categorical text data into model-understandable numerical data, we use the Label Encoder class.

D. Feature Scaling
Standard scaler is used to standardize features by removing the mean and scaling to unit variance.Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data.

E. Model Training
Now comes the last part of the music classification genre project. The features have been extracted from the raw data and now we have to train the model. There are many ways through which we can train our model. Some of these approaches are: Multi class Support Vector Machines, K-Means Clustering, K-Means Neighbors, Convolutional Neural Networks, Recurrent Neural Network. For this project, we will be using Long Short Term Memory. Algorithm for training our model. We chose this approach because various forms of research show it to have the best results for this problem. For the LSTM model, we had used the Adam optimizer for training the model. The epoch that was chosen for the training model is 600. All of the hidden layers are using the Relu activation function and the output layer uses the softmax function. The loss is calculated using the sparse-categorical-cross entropy function. Dropout is used to prevent overfitting. We chose the Adam optimizer because it gave us the best results after evaluating other optimizers.



```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_60 (LSTM)               (None, 58, 256)           264192
_________________________________________________________________
lstm_61 (LSTM)               (None, 58, 128)           197120
_________________________________________________________________
lstm_62 (LSTM)               (None, 64)                49408
_________________________________________________________________
dense_31 (Dense)             (None, 10)                650
=================================================================
Total params: 511,370
Trainable params: 511,370
Non-trainable params: 0
_________________________________________________________________
None
```

Fig. 7. LSTM Model Architecture.

F. Performance Evaluation

After getting the training accuracy,testing accuracy in our dataset by using LSTM. After training the algorithm, we have done the performance evaluation test on our test dataset to evaluate our LSTM trained model.

V. EXPERIMENTS

A. Dataset

Classification dataset which consists of 1,000 audio tracks, each 30 seconds long. It contains 10 genres, each represented by 100 tracks. The Genres are: Blues, Classical, Disco, Hip-hop, Jazz, Metal, Pop, Reggae, Rock. All the levels are equally distributed, here we can see a pie chart from the above discussion.

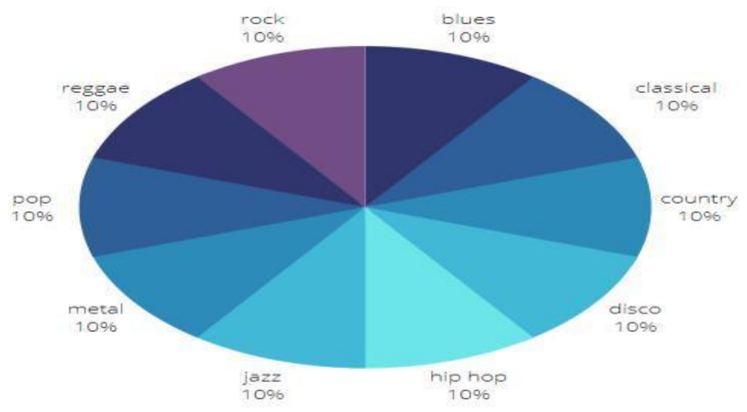


Fig. 8. Distribution of Music Genres.

B. Evaluation Metrics

In this section, we will discuss the Performance evaluation of our LSTM trained model. Also we will show how we increased our training and testing accuracy by tuning. We will give a short description in this section. After applying epoch= 7, batch size= 128, learning rate= 1e-2 as hyperparameter. We get,

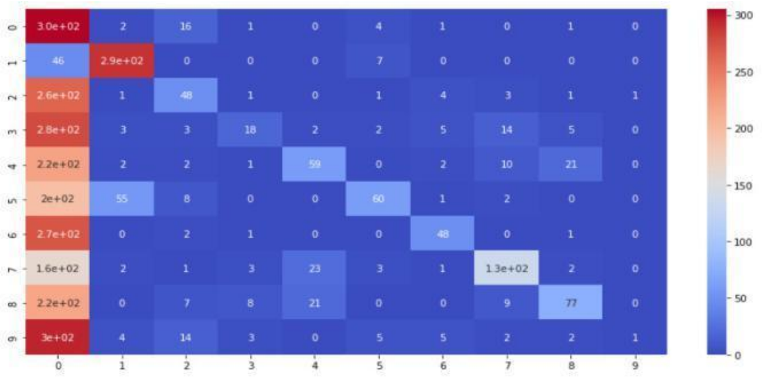


Fig. 9. Confusion Matrix of LSTM Model.

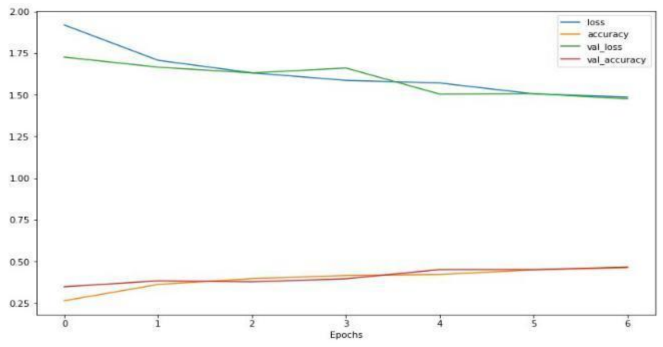


Fig. 10. Loss Curve of LSTM Model.

After applying epoch= 40, batch size= 220, learning rate= 1e-1 as hyperparameter. We get,

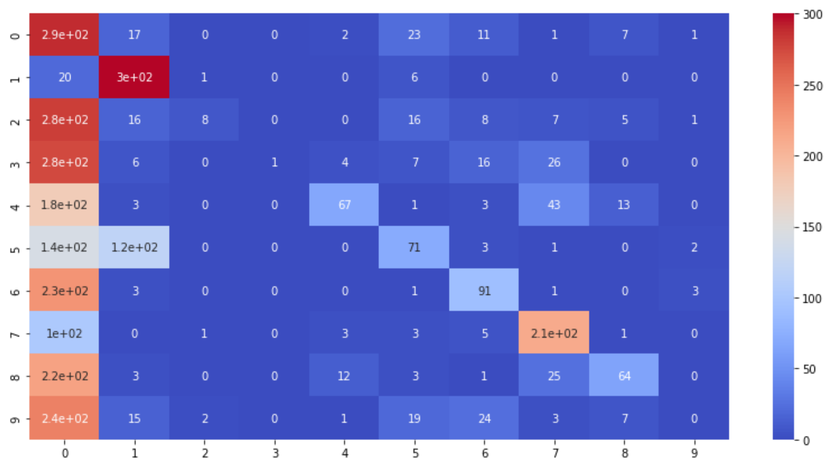


Fig. 11. Confusion Matrix of LSTM Model.

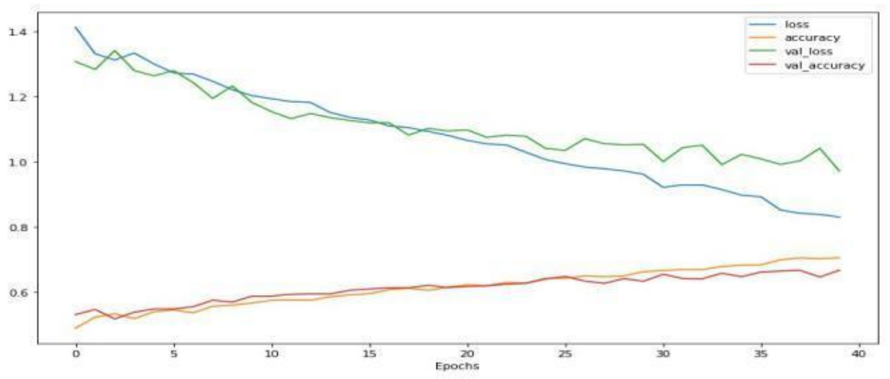


Fig. 12. Loss Curve of LSTM Model.

After applying epoch= 50, batch size= 220 ,learning rate= 1e-1 as hyperparameter. We get,

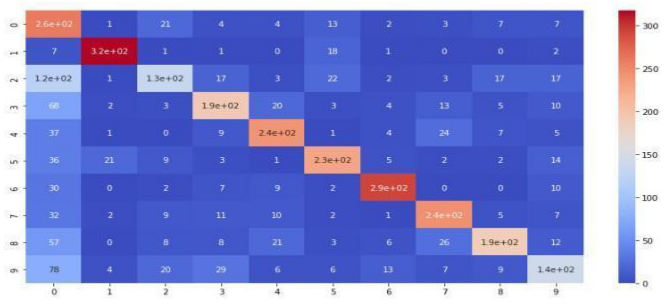


Fig. 13. Confusion Matrix of LSTM Model

In the above we showed how we increased our training and testing accuracy by hyperparameter tuning.

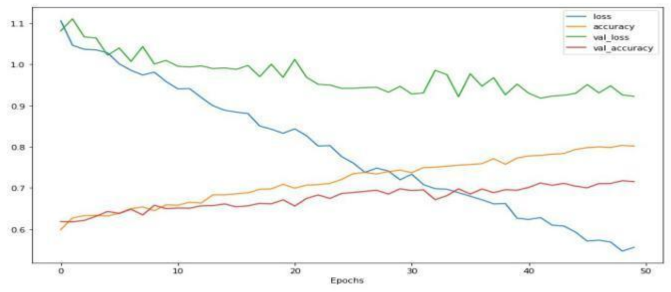


Fig. 14. Loss Curve of LSTM Model

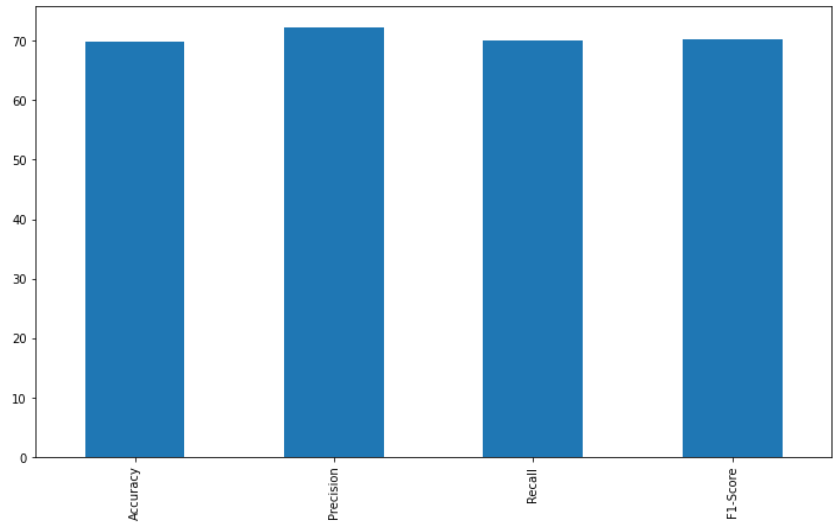


Fig. 15. Chart of Precision, Recall, F1-Score

C. Result

In this section, we have shown experiment comparisons of different experiments that we made. Here, our first experiment started with epoch= 7, batch size= 128, learning rate= 1e-2. Here we got the accuracy of 0.4626. In the Experiment 2 we used epoch with 20 and having same batch size and learning rate. We got Again we used same epoch and learning rate but this time we used batch size=220. Then we get the accuracy of 0.5867. Finally we used same batch size but increased the learning rate as 1e-1 and also increased epoch number as 120. After training the model we got accuracy of 0.9178 or 91.78%.

No. of Experiments	epoch	Batch size	Precision	Recall	F1-Score	Accuracy
1	7	128	55.47	33.16	29.98	33.45
2	20	128	65.22	46.36	46.36	45.92

3	40	128	70.56	57.76	59.61	57.51
4	40	220	72.83	72.50	72.40	72.37
5	50	220	73.70	73.41	73.32	73.37
6	90	220	91.71	91.47	91.50	91.51
7	120	220	91.81	91.78	91.70	91.78

TABLE I

PERFORMANCE COMPARISON USING LSTM MODEL

VI. CONCLUSION AND FUTURE DIRECTIONS

In conclusion, the experimental results show that our multi-step classifier based on Long Short-Term Memory (LSTM) model is effective in recognizing music genres. For 10-genre classification, the accuracy was 60% using a single LSTM[4]. But we achieved an accuracy of 91.78%, which was better than one of the four genre classification approaches having an accuracy of 51.88%. There is no denying that since all of this research has had to deal with the same issues in GTZAN, the results remain comparable. Hence, our future aim is to improve our accuracy by using more LSTM layers, adding dense layers and combining CNN-RNN model.