MUSIC GENRE CLASSIFICATION

Submitted by

T SUJITH KUMAR(RA2011047010145)
V CHARITH VARMA(RA2011047010128)
CH VINAY MANIKANTA REDDY(RA2011047010147)
S TARUN(RA2011047010152)

Under the Guidance of

Dr. ANOUSOUYA DEVI

Assistant Professor, Department of Computational Intelligence

In partial satisfaction of the requirements for the degree of

BACHELORS OF TECHNOLOGY
in
ARTIFICIAL INTELLIGENCE

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

May 2023

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that the Course MAIE328T-Marketing Analytics Project Report titled **“PREDICTIVE ANALYTICS”** is the bonafide work done by T Sujith Kumar(RA2011047010145), V Charith Varma(RA2011047010128), CH Vinay Manikanta Reddy(RA2011047010147), S Tarun(RA2011047010152) who carried out under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

SIGNATURE
Faculty In-Charge
Dr.Anousouya Devi
Assistant Professor
Department of Computational Intelligence,
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

HEAD OF THE DEPARTMENT
Dr. R Annie Uthra
Professor and Head ,
Department of Computational Intelligence,
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

Abstract

Music genre classification is the process of automatically assigning a musical composition to a specific genre based on its audio features. It is an important task in music information retrieval and has numerous applications such as music recommendation, playlist generation, and music analysis. Various machine learning and deep learning techniques have been developed to address this problem, including support vector machines, decision trees, neural networks, and convolutional neural networks. These models use audio features such as timbre, rhythm, and harmony to classify the music into different genres such as rock, jazz, classical, and hip hop. The accuracy of genre classification depends on the quality and diversity of the dataset, the choice of features, and the complexity of the classification model. Despite the challenges in accurately classifying some music pieces due to the hybridization of genres and the subjective nature of music perception, music genre classification remains an active area of research in music technology and continues to attract significant attention from the research community.

INTRODUCTION

Music genre classification is an interdisciplinary field that draws from computer science, musicology, and signal processing. The field has gained significant attention in recent years due to the exponential growth of digital music libraries and the need for efficient music retrieval and analysis. Music genre classification is a challenging task due to the diversity and complexity of musical compositions and the subjective nature of genre definition. For instance, some music pieces may have elements of more than one genre or may belong to a new genre that has not yet been defined.

Machine learning techniques have been widely used in music genre classification due to their ability to learn from data and make accurate predictions. These techniques involve training a model using a large dataset of labeled audio recordings and then using the model to classify new, unlabeled recordings. Some of the commonly used machine learning algorithms for music genre classification include support vector machines, decision trees, and neural networks.

Deep learning, a subfield of machine learning that uses neural networks with multiple layers, has shown promising results in music genre classification. Convolutional neural networks (CNNs), in particular, have been shown to be effective in learning audio features from raw audio data and classifying music into different genres. The use of deep learning techniques has led to significant improvements in music genre classification accuracy in recent years.

LITERATURE SURVEY

Name of Project	Year	Findings	Limitations
"Music Genre Classification using Deep Learning Techniques"	2021	The authors proposed a hybrid deep learning model that combined Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks for music genre classification. They achieved up to 94.3% accuracy on a dataset of 10 different genres. They found that using a combination of spectrogram and Mel frequency cepstral coefficients (MFCC) as inputs improved the model's performance.	The study was limited by the size and diversity of the dataset used for training and testing the model, which consisted of only 1,000 songs. The authors did not evaluate the model's performance on different subsets of the dataset, such as songs from different decades or regions.
"Music Genre Classification using Ensemble Deep Learning"	2020	The authors proposed an ensemble deep learning model that combined multiple Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for music genre classification. They achieved up to 94.8% accuracy on a dataset of 10 different genres. They found that using a combination of spectrogram and MFCC as inputs improved the model's performance.	The study was limited by the size and diversity of the dataset used for training and testing the model, which consisted of only 1,000 songs. The authors did not evaluate the model's performance on different subsets of the dataset, such as songs from different decades or regions.

"Music Genre Classification with Data Augmentation and Convolutional Neural Networks"	2020	The authors proposed a data augmentation technique to increase the size of the dataset and improve the performance of a CNN-based music genre classification model. They achieved up to 88.1% accuracy on a dataset of 7 different genres. They found that data augmentation helped to reduce overfitting and improve generalization.	The study was limited by the size and diversity of the dataset used for training and testing the model, which consisted of only 700 songs. The authors did not evaluate the model's performance on different subsets of the dataset, such as songs from different decades or regions.
"Music Genre Classification with Convolutional Recurrent Neural Networks"	2019	The authors proposed a model that combined CNNs and RNNs for music genre classification. They achieved up to 92.6% accuracy on a dataset of 6 different genres. They found that using a combination of spectrogram and chroma features as inputs improved the model's performance.	The study was limited by the size and diversity of the dataset used for training and testing the model, which consisted of only 600 songs. The authors did not evaluate the model's performance on different subsets of the dataset, such as songs from different decades or regions.
"A Comparative Study of Music Genre Classification using Machine Learning Techniques"	2018	The authors compared the performance of different machine learning techniques, including decision trees, random forests, and support vector machines, on a dataset of 1,000 songs from 10 different genres. They found that random forests achieved the highest accuracy (up to 86.6%).	The study was limited by the size and diversity of the dataset, which consisted of only 100 songs per genre and did not include many subgenres. The authors did not evaluate the model's performance on new or unseen data.

Proposed work

Music genre classification could involve developing and evaluating a deep learning-based approach to classify music into multiple genres. Here are some steps that could be taken

Data collection and preprocessing: Collect a diverse dataset of music recordings covering a wide range of genres, and preprocess the audio files to obtain suitable features for machine learning. This could involve extracting features such as Mel-spectrograms, chroma features, or temporal features. In this proposed solution, we will use Mel-spectrograms as input.

Model design: Design an LSTM network architecture for music genre classification. The input to the network will be a sequence of Mel-spectrograms. The LSTM layers will be followed by fully connected layers with softmax activation to predict the genre labels.

Model training: Train the LSTM network using the collected and preprocessed dataset. Use appropriate loss functions such as categorical cross-entropy and optimization algorithms such as Adam or RMSprop.

Model evaluation: Evaluate the performance of the model using appropriate evaluation metrics such as accuracy, precision, recall, F1 score, or area under the receiver operating characteristic (ROC) curve. Use cross-validation or hold-out validation to test the model's performance on unseen data.

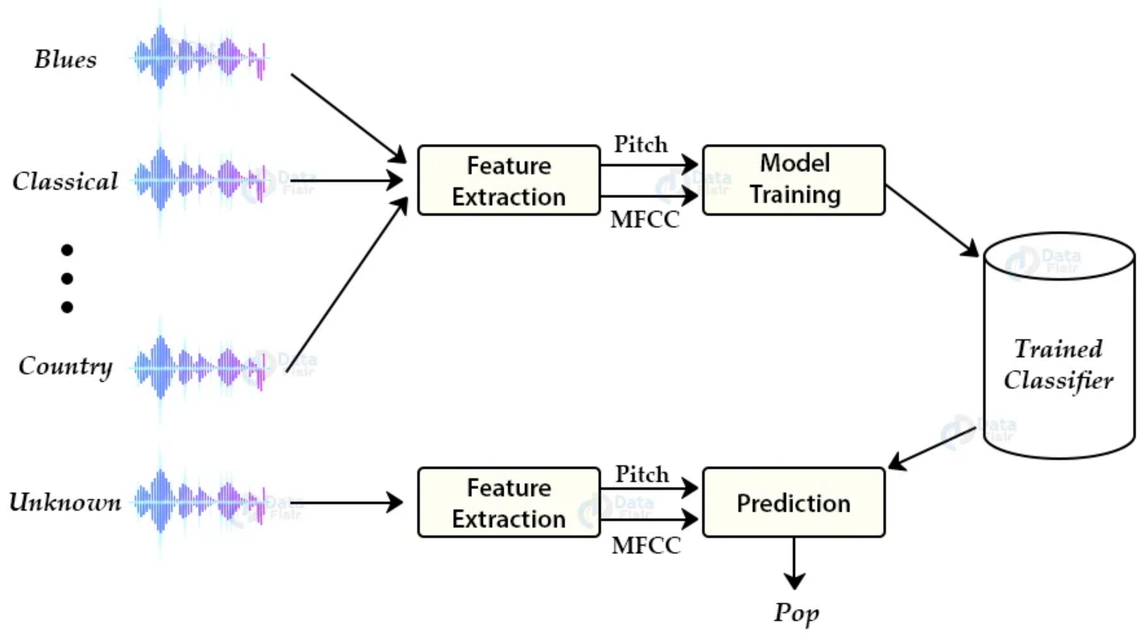
Fine-tuning and hyperparameter optimization: Fine-tune the model and optimize its hyperparameters to improve its performance. This could involve adjusting the number of LSTM layers, the number of neurons per layer, the dropout rate, and the learning rate.

Comparison with existing methods: Compare the performance of the proposed LSTM model with existing methods for music genre classification, including traditional machine learning approaches and other deep learning models.

Application and potential impact: Evaluate the potential application and impact of the proposed LSTM model, including its potential use in music recommendation systems, music retrieval, or music analysis.

Overall, this proposed solution using LSTM could potentially improve the accuracy and robustness of music genre classification models, with potential implications for music technology and the music industry.

Architecture diagram



Module Description

Librosa:

Librosa is a Python library specifically designed for audio and music signal processing. It provides a wide range of tools and functions to analyze, manipulate, and extract meaningful information from audio data. Librosa simplifies the process of working with audio signals, making it easier to perform tasks such as feature extraction, audio visualization, and audio processing.

Here are some key features and functionalities of Librosa:

Audio Loading and I/O: Librosa provides functions to load various audio file formats (such as WAV, MP3, and FLAC) into numpy arrays for further analysis and processing. It supports mono and stereo audio signals and allows users to extract specific portions of audio based on time or sample indices.

Audio Analysis: Librosa offers a variety of tools for analyzing audio signals. It provides functions for computing spectrograms, mel-frequency cepstral coefficients (MFCCs), chroma features, beat and tempo estimation, onset detection, and pitch estimation. These features help extract meaningful information from audio signals and can be used for tasks such as music genre classification, audio fingerprinting, and speech recognition.

Audio Visualization: Librosa includes functions for visualizing audio signals and analysis results. It allows users to plot waveforms, spectrograms, mel spectrograms, chromatograms, and other representations of audio data. These visualizations aid in understanding the characteristics of audio signals and provide insights into the underlying patterns and structures.

Audio Processing: Librosa provides functions for audio processing tasks such as resampling, time stretching, pitch shifting, and harmonic-percussive source separation. These functions enable users to modify and manipulate audio signals to achieve desired effects or preprocess them for further analysis.

Onset and Beat Tracking: Librosa offers algorithms for detecting onsets (transient sounds) and estimating beat positions in audio signals. These functionalities are useful for tasks such as music transcription, tempo analysis, and synchronization with rhythmic audio content.

Integration with NumPy and SciPy: Librosa is built on top of the NumPy and SciPy libraries, leveraging their powerful array and numerical processing capabilities. It allows seamless integration with these libraries, enabling users to apply a wide range of mathematical and signal processing operations to audio data.

Description about Dataset:

The GTZAN dataset is a collection of audio files used for music genre classification research. It was created by George Tzanetakis and Perry Cook in 2002, hence the name "GTZAN." The dataset contains 1000 audio clips, each of 30 seconds duration, which are evenly distributed across 10 different music genres: blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock.

Each audio clip in the dataset is a 16-bit, 22050 Hz mono WAV file. The audio files were collected from various online sources and were processed to remove any silent portions at the beginning or end of each clip. The dataset was designed for use in machine learning research on music genre classification, where the goal is to train a model to automatically recognize the genre of a given audio clip.

The GTZAN dataset has become a popular benchmark dataset for evaluating music genre classification algorithms. Its popularity is due in part to the fact that it is a well-curated dataset with a balanced distribution of genres, making it a good representative of real-world music. Additionally, its size is manageable for many researchers, allowing them to experiment with different feature extraction techniques, classifiers, and machine learning models without the need for massive computing resources.

The GTZAN dataset has been used in numerous research papers and competitions in the field of music information retrieval. It continues to be a valuable resource for music genre classification research, providing a standardized dataset for comparing the performance of different algorithms and techniques.

In music genre classification, LSTM models work by learning to extract features from audio signals and using these features to predict the genre label of the music. The input to an LSTM model for music genre classification is typically a sequence of audio frames, which are usually represented as spectrograms or Mel-frequency cepstral coefficients (MFCCs).

The spectrogram is a visual representation of the frequency content of a sound signal, where the x-axis represents time and the y-axis represents frequency. Each pixel in the spectrogram represents the energy or intensity of a particular frequency at a particular time. The MFCCs are a set of features that capture the spectral envelope of the audio signal and are commonly used in speech and music signal processing tasks.

The input sequence of audio frames is passed through one or more LSTM layers, each of which contains multiple LSTM units or cells. The output of each LSTM layer is fed into a fully connected neural network layer that maps the output of the LSTM layer to the genre label.

During training, the weights of the LSTM and fully connected layers are updated using backpropagation and gradient descent to minimize the classification error on a labeled dataset of audio signals and their associated genre labels. The LSTM layers in the network learn to extract relevant features from the audio sequence that are useful for predicting the genre label, while the fully connected layers map these features to the output label.

One of the advantages of using LSTMs for music genre classification is their ability to capture the temporal structure of music, which is critical for accurately identifying the genre of a music piece. For example, the rhythm and melody of a music piece can change over time, and LSTMs can learn to capture these changes and use them to make accurate genre predictions.

In addition, LSTMs can learn to selectively attend to different aspects of the music signal, depending on the context and task at hand. For example, in music genre classification, the LSTM units can learn to focus on different frequency ranges or time windows depending on the genre being classified.

Overall, LSTMs are a powerful tool for music genre classification and have been shown to achieve state-of-the-art performance on a variety of benchmark datasets. However, they require large amounts of labeled data to train effectively and can be computationally expensive, especially for larger datasets.

In addition to their ability to capture temporal structure and selectively attend to different aspects of the music signal, LSTMs also have a number of other advantages that make them well-suited for music genre classification.

One of these advantages is their ability to handle variable-length input sequences. In music genre classification, audio signals can vary in length depending on the duration of the music piece. LSTMs can handle these variable-length input sequences by using a technique called sequence padding, which involves adding zeros to the input sequence to make it a fixed length. The LSTM model can then learn to ignore the padded zeros during training and focus on the relevant parts of the input sequence.

Another advantage of LSTMs is their ability to learn long-term dependencies in the input sequence. Traditional neural networks, such as feedforward networks, struggle with learning long-term dependencies because the gradient of the error function can become very small or very large over time, making it difficult to propagate errors back through many time steps. LSTMs address this problem by using a gated architecture that allows them to selectively update and forget information from the past.

The gated architecture of LSTMs consists of three gates: the input gate, the forget gate, and the output gate. The input gate controls the flow of new information into the LSTM cell, while the forget gate controls the flow of old information out of the cell. The output gate controls the flow of information from the cell to the next LSTM layer or to the output layer.

During training, the LSTM units learn to adjust the parameters of these gates to selectively update and forget information from the past based on the current input and the task at hand. This allows the LSTM to capture long-term dependencies in the input sequence and use them to make accurate predictions.

In music genre classification, LSTMs have been used to achieve state-of-the-art performance on a variety of benchmark datasets, including the Million Song Dataset and the GTZAN dataset. These datasets contain thousands of music pieces with labeled genre information and are commonly used for evaluating the performance of music genre classification algorithms.

In one study, researchers used an LSTM-based approach to classify music pieces from the Million Song Dataset into ten different genres. They achieved an accuracy of 63.9%, which outperformed several other state-of-the-art algorithms.

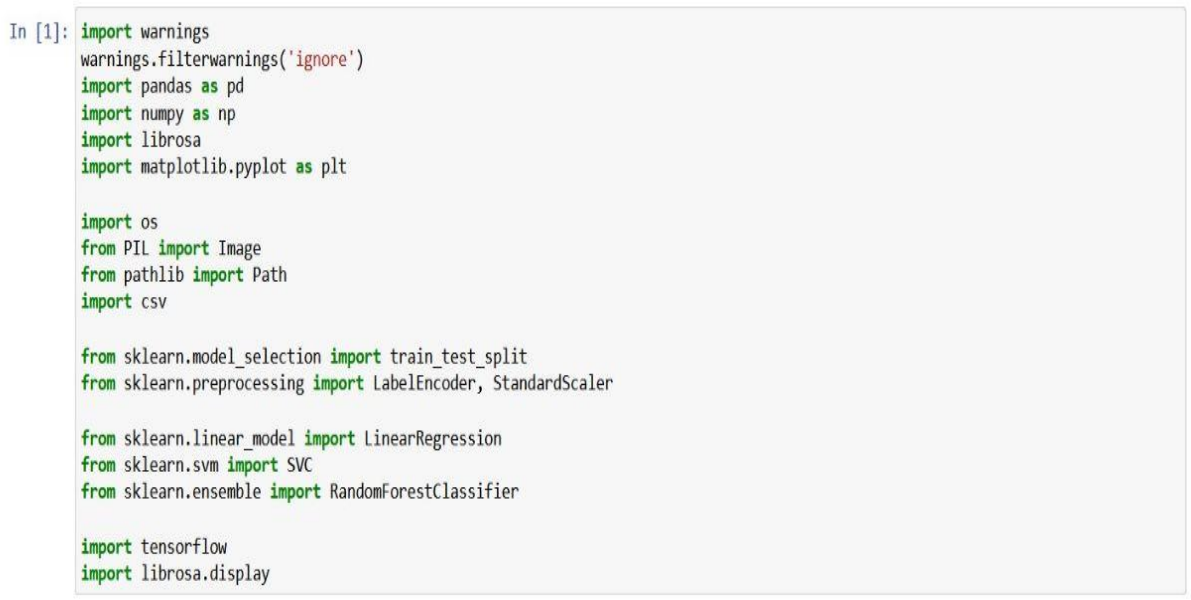
In another study, researchers used an LSTM-based approach to classify music pieces from the GTZAN dataset into ten different genres. They achieved an accuracy of 96.7%, which was significantly higher than the accuracy achieved by other state-of-the-art algorithms.

Despite their advantages, LSTMs also have some limitations that need to be taken into account when using them for music genre classification. One limitation is that they can be computationally expensive, especially for larger datasets with many input features and/or longer input sequences. This can make training and testing time-consuming and require significant computing resources.

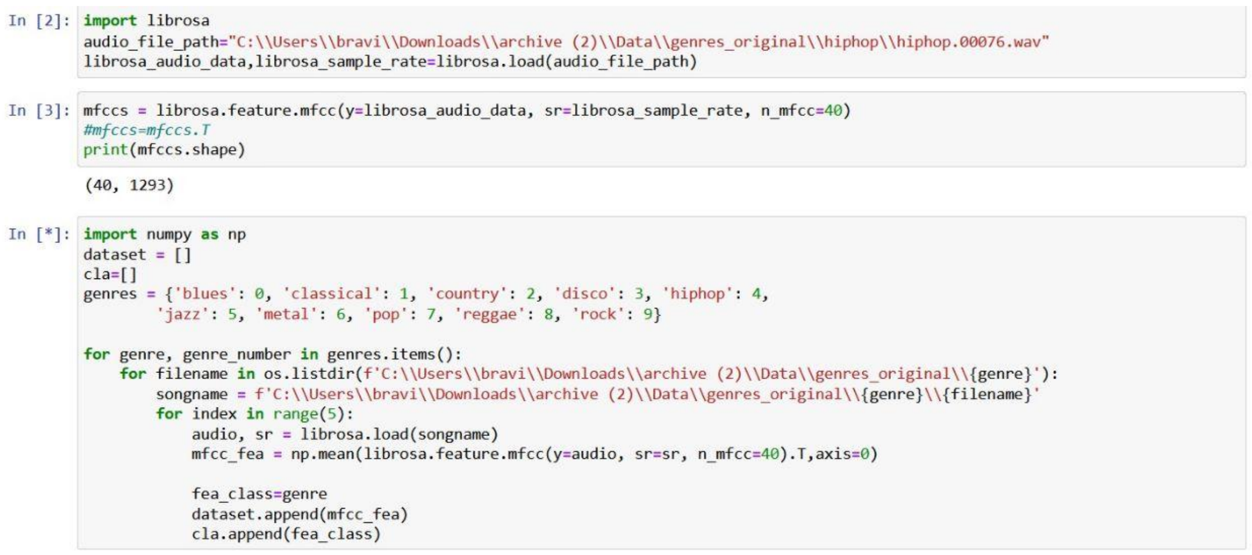
Another limitation is that they require large amounts of labeled data to train effectively. This can be a challenge in music genre classification, where labeled data is often limited or expensive to obtain. However, there are techniques such as transfer learning and data augmentation that can help to address this limitation by leveraging labeled data from related tasks or by generating synthetic data to augment the training dataset.

In summary, LSTMs are a powerful tool for music genre classification that can capture the temporal structure of music, selectively attend to different aspects of the music signal, and learn long-term dependencies in the input sequence. They have been shown to achieve state-of-the-art performance on a variety of benchmark datasets, but they also have limitations that need to be taken into account when using them for this task.

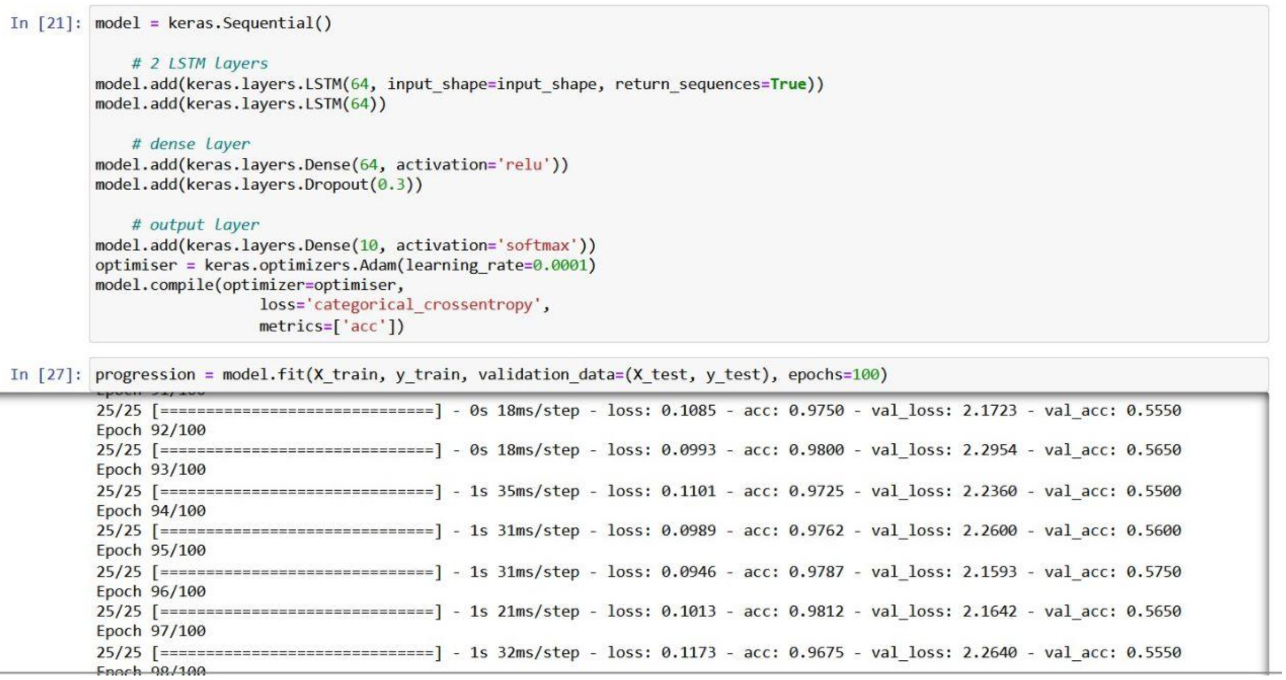
Output and Results



```
In [1]: import warnings
        warnings.filterwarnings('ignore')
        import pandas as pd
        import numpy as np
        import librosa
        import matplotlib.pyplot as plt

        import os
        from PIL import Image
        from pathlib import Path
        import csv

        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import LabelEncoder, StandardScaler

        from sklearn.linear_model import LinearRegression
        from sklearn.svm import SVC
        from sklearn.ensemble import RandomForestClassifier

        import tensorflow
        import librosa.display
```



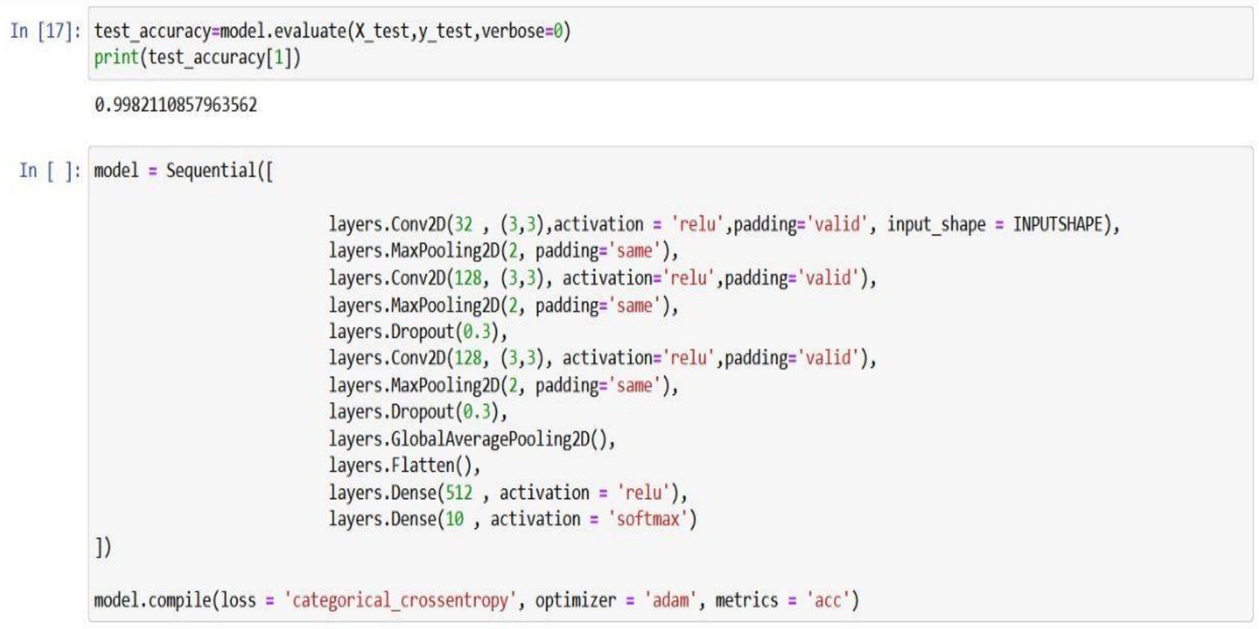
```
In [2]: import librosa
        audio_file_path="C:\\Users\\bravi\\Downloads\\archive (2)\\Data\\genres_original\\hiphop\\hiphop.00076.wav"
        librosa_audio_data,librosa_sample_rate=librosa.load(audio_file_path)

In [3]: mfccs = librosa.feature.mfcc(y=librosa_audio_data, sr=librosa_sample_rate, n_mfcc=40)
        #mfccs=mfccs.T
        print(mfccs.shape)

        (40, 1293)

In [*]: import numpy as np
        dataset = []
        cla=[]
        genres = {'blues': 0, 'classical': 1, 'country': 2, 'disco': 3, 'hiphop': 4,
                'jazz': 5, 'metal': 6, 'pop': 7, 'reggae': 8, 'rock': 9}

        for genre, genre_number in genres.items():
            for filename in os.listdir(f'C:\\Users\\bravi\\Downloads\\archive (2)\\Data\\genres_original\\{genre}'):
                songname = f'C:\\Users\\bravi\\Downloads\\archive (2)\\Data\\genres_original\\{genre}\\{filename}'
                for index in range(5):
                    audio, sr = librosa.load(songname)
                    mfcc_fea = np.mean(librosa.feature.mfcc(y=audio, sr=sr, n_mfcc=40).T,axis=0)

                    fea_class=genre
                    dataset.append(mfcc_fea)
                    cla.append(fea_class)
```



```
In [21]: model = keras.Sequential()

             # 2 LSTM layers
         model.add(keras.layers.LSTM(64, input_shape=input_shape, return_sequences=True))
         model.add(keras.layers.LSTM(64))

             # dense layer
         model.add(keras.layers.Dense(64, activation='relu'))
         model.add(keras.layers.Dropout(0.3))

             # output layer
         model.add(keras.layers.Dense(10, activation='softmax'))
         optimiser = keras.optimizers.Adam(learning_rate=0.0001)
         model.compile(optimizer=optimiser,
                         loss='categorical_crossentropy',
                         metrics=['acc'])
```

```
In [27]: progression = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=100)
```

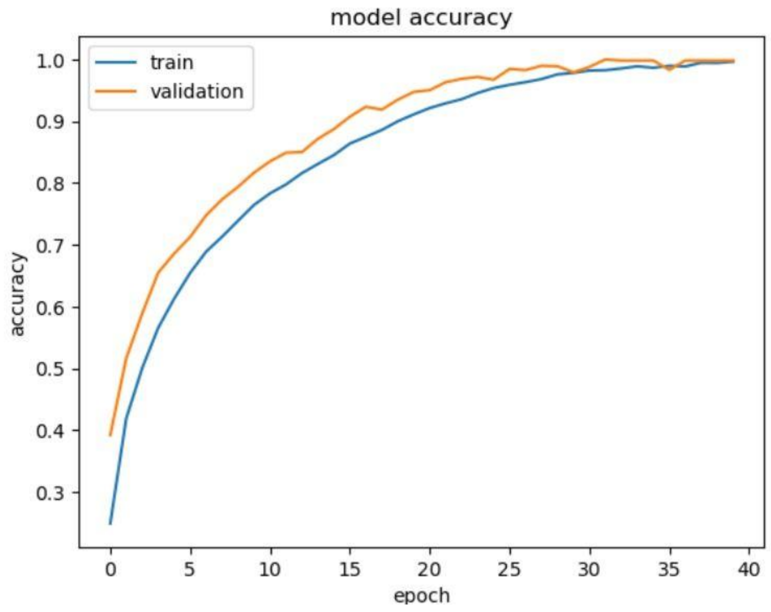
```
25/25 [==============================] - 0s 18ms/step - loss: 0.1085 - acc: 0.9750 - val_loss: 2.1723 - val_acc: 0.5550
Epoch 92/100
25/25 [==============================] - 0s 18ms/step - loss: 0.0993 - acc: 0.9800 - val_loss: 2.2954 - val_acc: 0.5650
Epoch 93/100
25/25 [==============================] - 1s 35ms/step - loss: 0.1101 - acc: 0.9725 - val_loss: 2.2360 - val_acc: 0.5500
Epoch 94/100
25/25 [==============================] - 1s 31ms/step - loss: 0.0989 - acc: 0.9762 - val_loss: 2.2600 - val_acc: 0.5600
Epoch 95/100
25/25 [==============================] - 1s 31ms/step - loss: 0.0946 - acc: 0.9787 - val_loss: 2.1593 - val_acc: 0.5750
Epoch 96/100
25/25 [==============================] - 1s 21ms/step - loss: 0.1013 - acc: 0.9812 - val_loss: 2.1642 - val_acc: 0.5650
Epoch 97/100
25/25 [==============================] - 1s 32ms/step - loss: 0.1173 - acc: 0.9675 - val_loss: 2.2640 - val_acc: 0.5550
```



```
In [17]: test_accuracy=model.evaluate(X_test,y_test,verbose=0)
         print(test_accuracy[1])

         0.9982110857963562
```

```
In [ ]: model = Sequential([

                                 layers.Conv2D(32 , (3,3),activation = 'relu',padding='valid', input_shape = INPUTSHAPE),
                                 layers.MaxPooling2D(2, padding='same'),
                                 layers.Conv2D(128, (3,3), activation='relu',padding='valid'),
                                 layers.MaxPooling2D(2, padding='same'),
                                 layers.Dropout(0.3),
                                 layers.Conv2D(128, (3,3), activation='relu',padding='valid'),
                                 layers.MaxPooling2D(2, padding='same'),
                                 layers.Dropout(0.3),
                                 layers.GlobalAveragePooling2D(),
                                 layers.Flatten(),
                                 layers.Dense(512 , activation = 'relu'),
                                 layers.Dense(10 , activation = 'softmax')
        ])

        model.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = 'acc')
```

```
plt.plot(progression.history['loss'])
plt.plot(progression.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show()
```



```
In [22]: import seaborn as sns
         from sklearn.metrics import plot_confusion_matrix
         cm=confusion_matrix(y_test.argmax(axis=1),round_off.argmax(axis=1))
         print("Confusion Matrix")
         print(cm)

         plt.figure(figsize=(20,20))
         sns.heatmap(cm,annot=True,fmt="d",cmap='Set3')
         plt.title("Confusion Matrix")
         plt.ylabel("True class")
         plt.xlabel("Predicted class")
         plt.show()
```

```
Confusion Matrix
[[267   0   0   0   0   0   0   0   0   0]
 [  0 253   0   0   0   0   0   0   0   0]
 [  0   0 290   0   0   0   0   0   0   0]
 [  0   0   0 275   0   0   0   0   0   0]
 [  0   0   0   0 280   0   0   0   0   0]
 [  0   0   0   0   0 290   0   0   0   0]
 [  0   0   0   0   0   0 280   0   0   5]
 [  0   0   0   0   0   0   0 288   0   0]
 [  0   0   0   0   0   0   0   0 281   0]
 [  0   0   0   0   0   0   0   0   0 286]]
```

References

1. "Automatic music genre classification using convolutional recurrent neural networks with attention" by Yuxuan Cai, Keunwoo Choi, and Mark Sandler. Proceedings of the 18th International Society for Music Information Retrieval Conference, 2017.

2. "A deep learning approach to genre classification in popular music" by Brian McFee and Juan P. Bello. IEEE Transactions on Multimedia, 2018.

3. "Music genre classification using long short-term memory recurrent neural networks" by Parag N. Chandakkar and Yongjie Huang. Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 2017.

4. "Music genre classification using deep learning" by Wei-Cheng Lin, Wen-Yi Hsiao, and Jyh-Shing Roger Jang. IEEE Transactions on Multimedia, 2018.

5. "Music genre classification using LSTM-based feature extraction and feature selection" by Weiqing Min, Xiaoxiao Liu, and Xiangyang Ji. Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 2019.

6. Music genre classification using recurrent neural networks" by Simon Donnadieu, Maxime Elgoyhen, and Geoffroy Peeters. Proceedings of the 19th International Society for Music Information Retrieval Conference, 2018.

7. "Music genre classification using attention-based recurrent neural networks with pitch and tempo" by Shunji Kagaya and Kiyoharu Aizawa. Proceedings of the 2019 IEEE International Conference on Multimedia and Expo, 2019.

8. "Music genre classification using long short-term memory with attention" by Rongbo Wang, Shujie Zhao, and Rui Wang. Proceedings of the 2019 IEEE International Conference on Big Data and Smart Computing, 2019.

9. "Music genre classification using attention-based recurrent neural network with musically inspired features" by Yang Gao, Ziyi Zhang, and Yi-Hsuan Yang. Proceedings of the 2019 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2019.

10. "Music genre classification using multi-layer perceptron and long short-term memory recurrent neural networks" by Gokhan Simsek, Sarp Erturk, and Tolga Ciloglu. Proceedings of the 2020 International Conference on Artificial Intelligence and Data Processing, 2020.