

|  |  |
| --- | --- |
| **PES UNIVERSITY** |  |
| **B. Tech, Sem VII**  **Session: Aug-Dec, 2020**  **UE17EC415: Speech Processing**  **Topic : Music Genre Classification using Machine Learning** | |

**Project Report: SP\_ASSIGNMENT2 GROUP-26**

***Submitted by:***

*Shobhit Tuli PES1201700192*

*Meghana Goswami PES1201700244*

*Utkarsh Raj PES1201700387*

**INTRODUCTION**

*As the quantity of music being released daily continues to sky-rocket, especially on internet platforms such as Soundcloud and Spotify the need for accurate meta-data required for database management and search/storage purposes climbs in proportion. Being able to instantly classify songs in any given playlist or library by genre is an important functionality for any music streaming/purchasing service, and the capacity for statistical analysis that correct and complete labeling of music and audio provides is essentially limitless.*

*So our project focuses on classifying music into their respective genre automatically using a machine learning algorithm that works on MFCC(Mel-Frequency Cepstral Coefficients) feature extraction.*

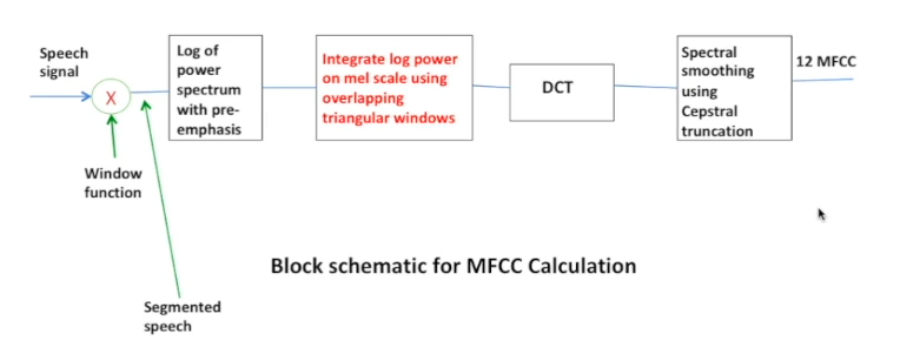
**HYPOTHESIS:**

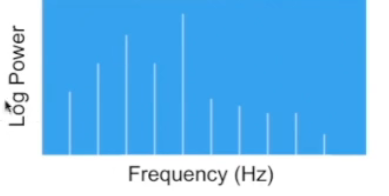
*To predict Music Genres based on MFCCs of Audio Files.*

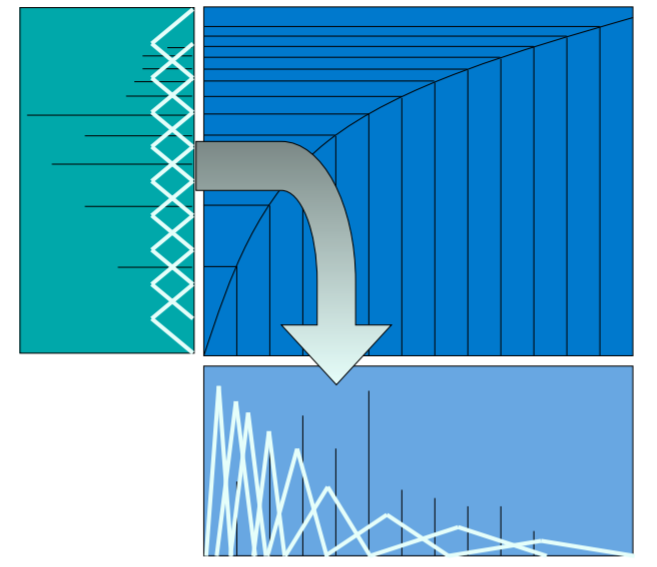
**What are MFCCs?**

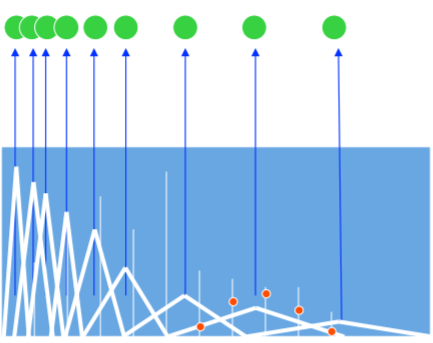
* *Sound generated due to excitation sources by glottis are filtered by the shape of the vocal tract including tongue,teeth etc.*
* *Shape of articulators determines the sound that comes out.*
* *If shape is determined correctly we can accurately represent phoneme being produced.*
* *We can synthetically produce sound if the shape is determined correctly.*
* *MFCCs accurately represent the envelope of the short time power spectrum.*
* *These parameters quantify the envelope which represents the vocal tract parameter.*

**MFCC Calculation:**





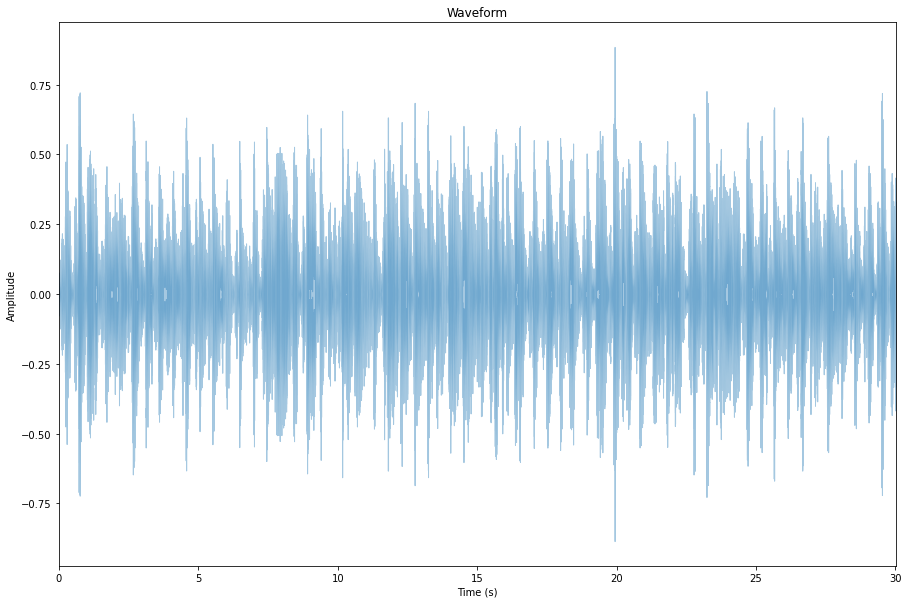




**Overview and plot of Audio file:**

We load the Signal File with a sampling rate of 22050. So Signal will have Sr\*T values or amplitudes.

Upon plotting the waveform of the signal we get.



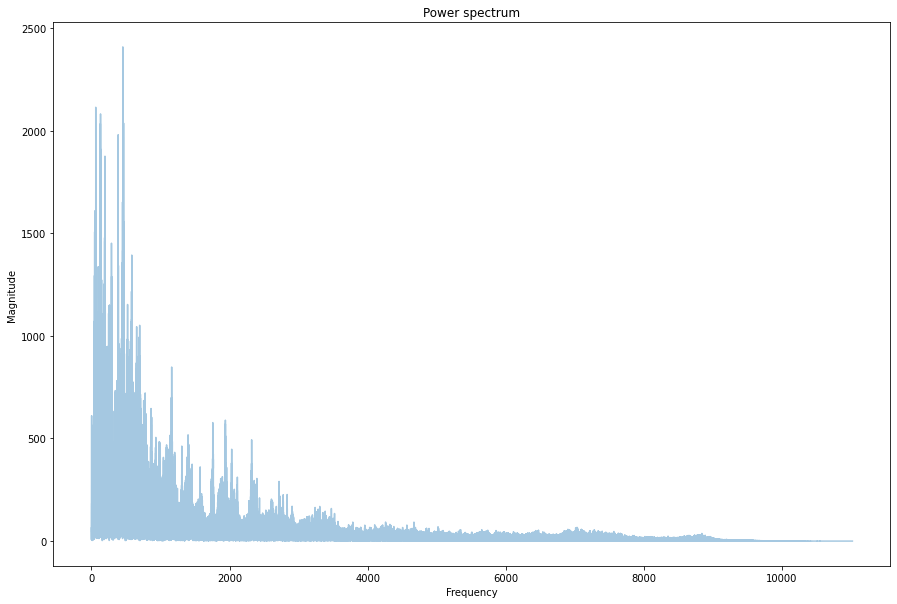
We move from Time domain to Frequency domain.

For that we do Fast Fourier Transform.

We expect to get as many complex values as samples for which we take the magnitude into account.

These magnitudes Represent contribution of that particular frequency to the sound.

We plot the Power Spectrum



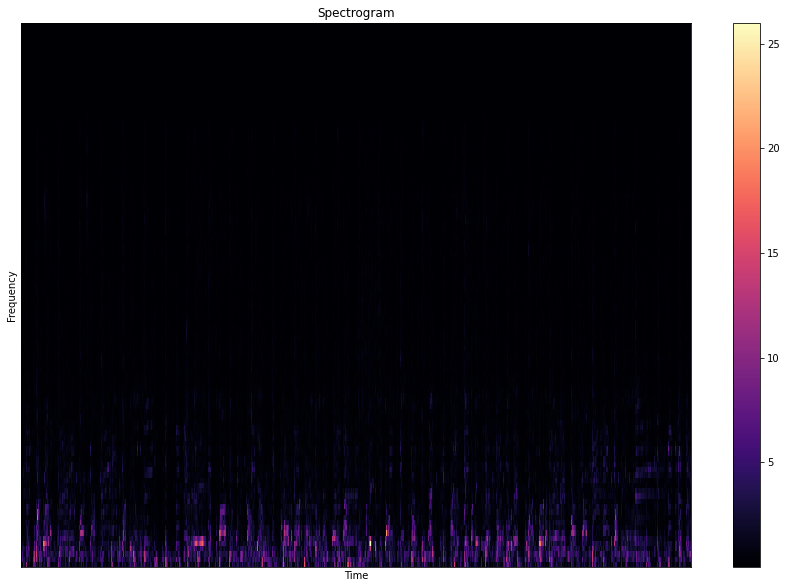
Most of the energy is concentrated in the lower frequencies. The higher we go, the less energy and less contribution.

We have only taken half of the plot because of the presence of mirror image on the other which basically repeats information.

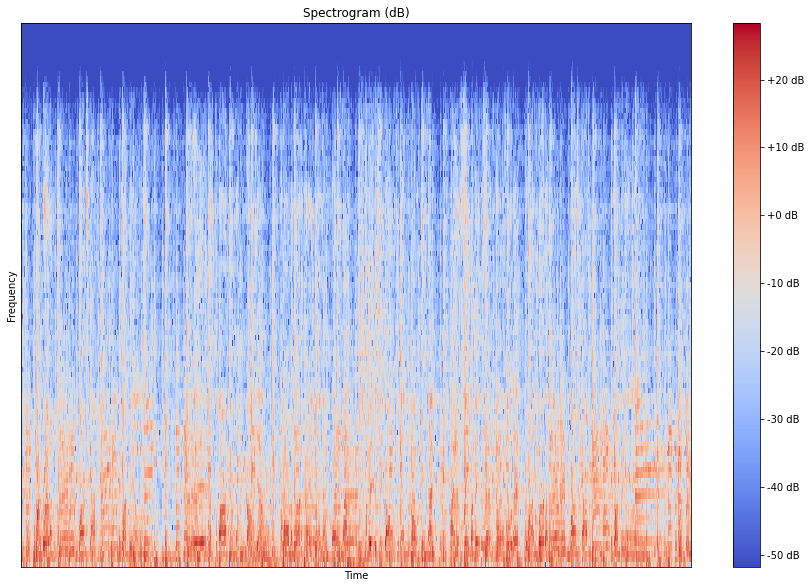
This Power spectrum is like a static snapshot.

We want to know how each frequency is contributing to overall sound for which we take STFT.

We Use Spectrogram to get an overview of stft.

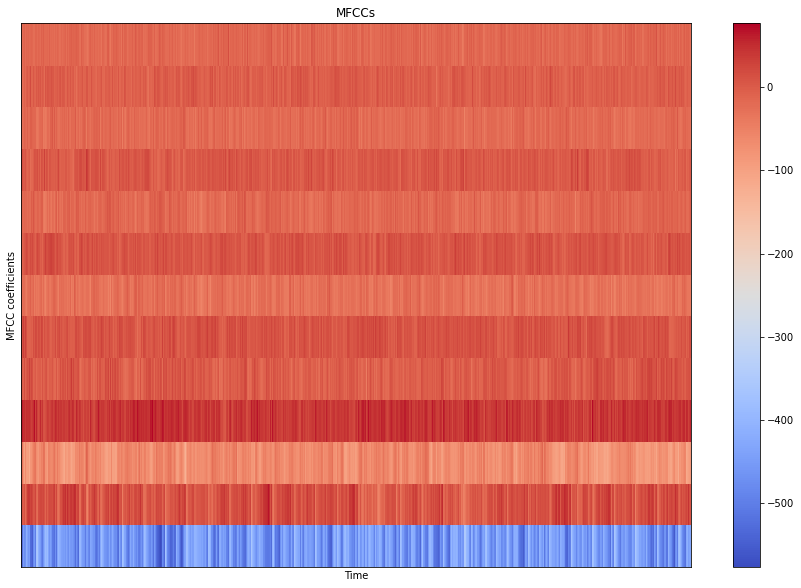


This is linear but we perceive sound non linearly so the plot is again taken on a logarithmic scale.



The picture Becomes more clear and we can say that lower frequencies have a bigger contribution.

For acquiring MFCC we use a number of samples per fft,window size and also the MFCC we want(around 13).



Y-Axis is divided into 13 intervals because we specified 13 Coefficients.

We basically see how MFCC is evolving over time.

**Pre-Processing of music data**

We extract the inputs and the targets like the labels and MFCC's from the music data set and store that in adjacent files so that we can use it when we train our neural network.

So, first of all, for a music genre classifier, we need a music data set.

The dataset is then divided into ten different folders where each folder has different musical genres like blues, classical, rock, reggae, etc and inside each genre folder, we have a hundred different songs where each song contains only 30 seconds of the original song.

Function:

Define a high-level function that will call save MFCC to input data and other values which are relative to the MFCC feature itself. Here we take the number of MFCC’s to be 13 and the number of FFT’s to be 2048. We give the number of segments as five because we know that in deep learning we need a lot of data. Thus segmenting a single data into five different data increases the amount of input data.

Step 1 – Build a dictionary to store data.

Step 2 – loop through all the genres using os.path and enumerate the number of times.

Step 3 – Passing reduced dataset for the quick running of the program

Step 4 – save semantics labels like blues, rock, etc by taking the last index of the datapath

Step 5 – Process files for a specific genre.

* Loading audio files using librosa(audio processing Library)
* Taking sample rate = 22050 which is a customary value for sample rate.
* Process segments extracting MFCC and storing the data
* In data, we need to take a number of samples per segment.

(Sometimes we don’t have an expected number of samples because the duration of the song might be slightly more or less so we will have more or fewer vectors in MFCC and thus we don’t want to pass them through our training process. We need to ensure that we have the same number of MFCC's vectors for each segment)

* Store MFCC for segments if it has the expected length and also the labels.

Step 6 – Now we need to store all the processed data in a JSON file.

So in JSON files, we have all the values of mapping, MFCC, and labels.

**IMPLEMENTATION OF NEURAL NETWORK**

This project is basically a multiclass classification problem which has 10 different genres (blues,classical,disco,hip hop,etc.)

Our objective is to predict the genre of a given audio correctly using its MFCC.

Basic steps of Implementation are as follows:

* Loading the data

We load the data from the dataset path and then convert the lists obtained into array using numpy .Input array containing the mfcc and output array containing the labels (hip hop ,rock ,classical, etc)

* Splitting the data into training and testing datasets.

Using sklearn we import train\_test\_split and we split the data into testing and training (70% training and 30% testing)

* Building the network architecture.

Multilayer perceptron:

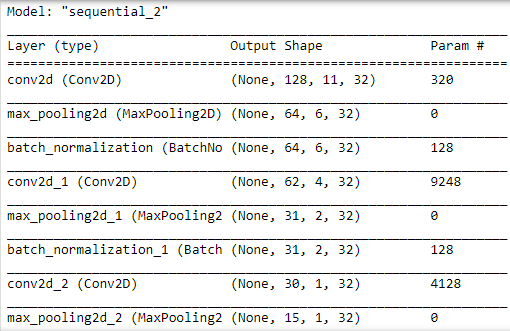
Using keras and tensor flow we build our model with an input layer followed by 3 dense hidden layers and then the final output layer.

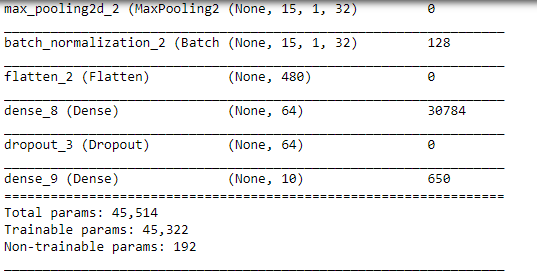
Convolutional Neural Network:

Again using keras and tensorflow we created our CNN model architecture in order to get a better result. We used an input layer followed by 3 convolutional layers each containing Convolutional Layer, Pooling Layer, and Fully-Connected Layer . We stack these three to form a convolutional network .

* Compiling the network.

We add an adam optimizer and a sparse loss function and then compile it.

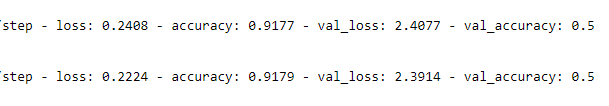




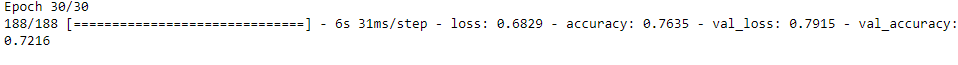
* Training the network.

We now train our network using the training dataset and later validate it using our testing dataset.

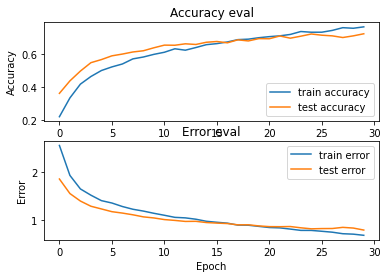
Multilayer perceptron:

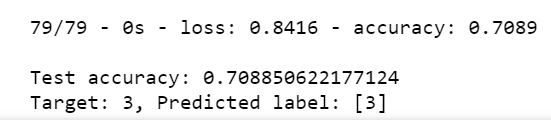


Convolutional Neural Network:



* Predicting the output.





**References:**

[1]:Musical instrument identification using MFCC ;Monica S. Nagawade, Varsha R. ; [2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)](https://ieeexplore.ieee.org/xpl/conhome/8241508/proceeding)

[2]:<https://towardsdatascience.com/implementing-alexnet-cnn-architecture-using-tensorflow-2-0-and-keras-2113e090ad98>

[3]: pesuacademy.com;UE17EC415: Speech Processing;Dr. Shikha Tripathi

[4]:Dataset: https://www.kaggle.com/andradaolteanu/gtzan-dataset-music-genre-classification