Audio Instruments Identification Using CNN and XGBoost

MSc Research Project
Data Analytics

Akshay Dahiya
Student ID: X17170494

School of Computing
National College of Ireland

Supervisor: Dr. Anu Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Akshay Dahiya……………………………………………………………………………
Student ID:	x17170494………………………………………………………………….……
Programme:	Master's in Science……………………………… **Year:** 2018/2019
Module:	Data Analytics……………………………………………………………………….………
Supervisor:	Dr. Anu Sahni……………………………………………………………………….………
Submission Due Date:	12-August-2019……………………………………………………….………
Project Title:	Audio Instruments Identification using CNN and XGBoost
Word Count:	……………23………………………… **Page Count**…6913…………………………………………….…….

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……

Date: ……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Audio Instruments Identification Using CNN and XGBoost

Akshay Dahiya
X17170494

Abstract

This paper is brief research on how to identify the audio instruments using machine learning. Algorithms of machine learning and artificial intelligence systems are sky boosting the music industries. With the help of machine learning, problems can be resolved with a much better solution. Creating music artificially using state of art software's likes **FL Studio, tractor, Soundation** and many others software for producing the digital and repeated pattern instrument sounds are being used in creation of music. Hence, music produced digitally, at different **Beats Per Minute (BPM)** therefore, it is now easy to recognize what kind of music is playing by using certain applications like **Shazam, Spotify, Sony Track ID** for identification of the music artist and the genre of the music. But finding instruments that were used to create the music is quite difficult. Sometimes even the applications which is used for identification of music , playing outside on speaker shows wrong results because of the working algorithm behind the applications do not understand the parallel pattern of the audio clips. As many instruments are used to produce the proper audio file so it is difficult to read the pattern of the instrument for the machine. That's the real game in this research. After training the data, which are audio clips of many instruments, this model can recognize the audio instruments in the audio clips of the music. This will help to boost in applications development like Shazam, Spotify, Sony Track ID and to music developers, producers to improve their skills.

Keyword - Beats per minute (BPM), CNN (Convolution Neural Network), XGBoost (Extreme Gradient Boosting), Librosa, MFCC (Mel Frequency Cepstral Coefficients).

1 Introduction

Multimedia is most important and most profitable part in the business these days. In early stages, music was produced by the proper hardware of audio instruments and then it was recorded by the recorder with help of high-quality mics. Then it was transfer to the machines to make copies. But now, multimedia content is created with the help of high-tech computer's software. In today's generation computer are used to create music albums instead of the original hardware instruments. With the help of many software technologies, algorithms and artificial intelligence, machine can now recognize the speech of the human and even more, many other hardware that can connect to computers to transfer the sound directly to the machines with automatic conversion of analog to digital format of the audio file. As Musical Instrument Digital Interface (**MIDI**) that was introduced for digitalization of the music by converting the analog signals to digitally which act as an interface between the instruments like synthesizers, drum machines, guitars etc. and the computer which records and understand the parallel patterns of the recorded music based on digital signal processing. So, it is hard to capture the different pitches created by the instruments at the same time, therefore some application for identification of music like Shazam, Sony Track ID gives unsuccessful results. In past, many have tried to create the instrument recognition algorithms with the help of machine learning and artificial intelligence, and it was started since 1990's. Some of the classifier created like naïve Bayesian classifiers, linear classifiers, and neural networks that gave 98% of accuracy for simple classifications (Pdfs.semanticscholar.org, 2019). The limitations of the classifiers also depend on the data set. There is much more work has been done after that. Accuracy is totally depending on the data which is used by the algorithms. Some of the classifiers has limitations due to difference in the class families and some recognize them as from the same class family. And some of the classifiers tested on the same pitches. Which gives unexpected results. We can eliminate this limitation by using the large data sets with different classes and train them to identify more and more speech and instruments of the audio clips and improving the identification, machine can learn now human like through experience on trained data. This research of instrument identification using **CNN (Convolution Neural Network)** and **XGBoost** is based on 4 different instrument classes which are **Acoustic Guitar at 120 BPM, Acoustic Guitar at 85 BPM, Drum Assorted Hit and Drum Kits.** The technique used in this research are CNN and XGBoost where accuracy achieved is 79% in identification of the instruments of different classes and at different pitches with different BPM.

- **Motivation**

When we hear any song or music playing somewhere on other place and we want to know what music is playing. With the help of some software we can know that what song is playing by just using a simple application like very famous Li-Chung Wang's Shazam application which was introduce in 2003. This application makes very simple in identification of music with some extra details like genre of the music, artist of the music, make of the music album etc. They have around 8 million song's fingerprints in their data base. That will make the accuracy rate much more higher in identification so whenever you want to know the song you can any time use this app and you can here that song again and again or even you can buy the album when it detects the song which support the musician and the music industry. Improvement in audio processing technology, developer can develop the algorithm to that will recognize the music within seconds no matter what music is played. This is how software and artificial intelligence working these days with the help of computer. Now what about the musician and composers, there are thousands of thousand instruments that were used in composition of music albums. The identification of the instrument with the help of an application or software makes it easier for the composer or musician to use his/her talent to make his/her own music by just knowing the audio instruments that were used in his/her favorite music composer playlist. As all things now going digital, music studios use digital format directly to create music of an audio instruments instead of using the audio instruments itself. This even further helps in the identification of the audio instruments for the machine learning methods algorithms.

2 Related Work

The related work shows how many and what kind of research has been done till date. It will help to understand how the development through time had happened over the proposed model and what limitation of the model is described and what kind of improvement will be need. It will help more to understand about the CNN and XGBoost model of the machine learning that how they implemented and useful over the time. However, most of the related work has been done on audio speech recognition (ASR). There is very less research I found of audio instrument recognition using the CNN and the XGBoost algorithms. And XGBoost is recently developed so there is less related work on this algorithm.

2.1 Audio Identification Using Machine Learning Algorithms

Many approaches were taken in past in the audio speech recognition. This was the 1st study in this field by using Gaussian and kNN classifiers by using 30 orchestral instruments. It was the 1st step in the recognition of the instrument detection using Gaussian and kNN algorithms. 1498 audio clips of 30 solo instrument known as orchestral class. Here we use different instrument at different BPM. (Eronen and Klapuri, n.d.) this paper describes the design of many piece of music data base like classical music, royal free music, jazz music etc. each consist of many audio sample (Pdfs.semanticscholar.org, 2019) paper shows investigation over those audios with prediction were obtain 28 different architecture. This research also done using the Gaussian technique with MFCC. The data set contain many different artists of the music.

2.2 Sparse Coding and Identification of Music Genre with many other Technique

(Mac.citi.sinica.edu.tw, 2019) use the sparse coding with Support vector machine algorithm which achieve 95% of accuracy rate on 24 different instruments by SVM model. The problem with the mono instrument that they used single class instrument to produce different sound. But some research is done on Polyphonic Instrument Identification in which many instruments were added to make more complex identifications of the audio. It is difficult task as many sounds has different patterns from which model can get confused by overlapping of the audio wave. (Anon, 2019) As the separation of the patterns need factorization through sparse and is recently used in machine learning which makes it difficult and complex by doing many coding. But in this research, the single classifier is used in each testing model. There is another very famous model known as SVM (Support Vector Machine) which has approach for the recognition of audio sounds. This model use both linear and non-linear formula. For the classification of the data it separated the hyper plane. But only works on fixed length of data and cannot work for classification of multiple length data. The function is used to fix the length of the data using fitting function then SVM is used. This makes it difficult to use, as compare to CNN and the XGBoost model as they handle much more complex data by using features and labelling the data. (Arxiv.org, 2019) this paper shows the identification of the music genre by using CNN and the ensembles with XGBoost, the paper contain large set of 2.1 million YouTube videos and extracted 10 seconds of audio clips from that videos. The audio has 40540 sounds clips were selected of different genre for the testing the models of machine learning. The result of identification of the genre for by VGG 16 CNN Fine tuning or Transfer learning was 64 and 63 respectively and the XGBoost feature based model got 59 percent of accuracy. The data set was very noisy because it was taken from the YouTube channel and split them into desired form. So, the data lost most of the information and did not used any preprocessing for the noisy data. Fan, Y., & Feng, S. (2016) In this research paper, Yong Fan and Shuang Feng uses 45 songs from the US top 45 song list of year 2015. They played the 45 song on speaker and record it from the laptop microphone with the noise in it to check the robustness of the model with noise in audio files. After using the model they found that after breaking the audio into small piece of 2 second, 3 second, 4 second, 5 second and in 6 second, the accuracy rate is much higher when the use lengthy clips of audio like 100 % in 6 second audio clip but in 2 second, model shows 60% accuracy rate. If the clips of audio are longer, they have bigger in size and need more space for the data storage and the model will takes time to implement on longer audio clips if audios are in couples of thousands. That's why there is a need of model that run on 1 or 2 seconds clips and gives higher accuracy rate.

2.3 Identification on Recorded Audio from Mic

(Quaero.org, 2019) In this research, Sebastien Fenet, Gael Richard and Yves Granier, they used the machine learning algorithm on 120 hours long of real audio broadcast which also consist of lots of noise thus the database in this research consisted of 30,000 French songs. Hashing technique was used, and the database pruning technique is adopted during the implementation. 5 second audio clips are used for the analysis with 50% of overlapping. 97.4% percent of the accuracy rate detected even in the pitch shifting of the audio clips. The preprocessing is done efficiently which takes lots of time by doing overlapping of audios with another and shifting the pitch makes time consuming. The algorithm "SAF" took 0.43 seconds per signal to compute the dentification and after using the pruning technique it was reduced to 0.28 seconds, but this makes model more complex to the developer and take time to implement for the identification of the audio. Yang, G., Chen, X., & Yang, D. (2014) this research is based on the major problem of the music and speech identification models which is the space saving data base that were created using the fingerprint technique. The model used so robust that they identify the audio clip even the distortion, then fingerprint was extracted from audio clip based on the content of it of the audio clip and kept in the database. When the query arrives, the matching fingerprint using searching algorithms identify the clip effectively. The extraction feature is used with the help of MFCC function. In past decade, all research is based on the accuracy of the identification and recall rate of the audio file, in which both aspects are important for large scale systems. This research shows the data was made smaller and the identification of audio is much quicker compares to others. As conducted experiment shows that 200,00 songs were used, and 20,000 clips are compressed in MP3 format which is compression technique and the recall rate of compressed file is 98%. But in this research, .wav format is used for the audio clips which is much better technique then MP3 format for the compression. It has higher compression ratio compare to MP3 format so that it is much more space saving for the database and the recall rate is same as the model designed. (Pdfs.semanticscholar.org, 2019) Janet Marques and Pedro J. Moreno in June showed their research implementing the Gaussian Mixture model and SVM model with FFT based cepstral coefficient, and FFT based Mel Cestral coefficient technique for the musical instrument classification. The model showed 70% of the error rate on Trambone and harpsichord instruments classified incorrectly. However, rest of the classification shows 17% error rate on 8 different instruments. But when using Mel Cestral coefficient technique and FFT based cepstral coefficient shows the improvement by decreasing the error rate by 2%. Which shows using the MFCC helps to improve the model to predict more accurately. This research uses both techniques while extracting the features of trained data before applying our model.

3 Research Methodology

There are two methodology used in the data mining related research which done on basis of KDD or CRISP-DM architecture. In my case, KDD (knowledge Discovery in Database) is used because business layer is not used here. The identification of the audio instruments detection does not have any direct relation with business. This process includes the raw data base along with preprocessing sampling and transform it. Machine learning methods (algorithms) are applied on the transformed data to calculate the patterns which we produce by making spectrogram of the of the audio .wav files. And then evaluate the model that were created thoroughly. The basic steps were followed as shown in figure 1 in this research. (Google.ie, 2019)

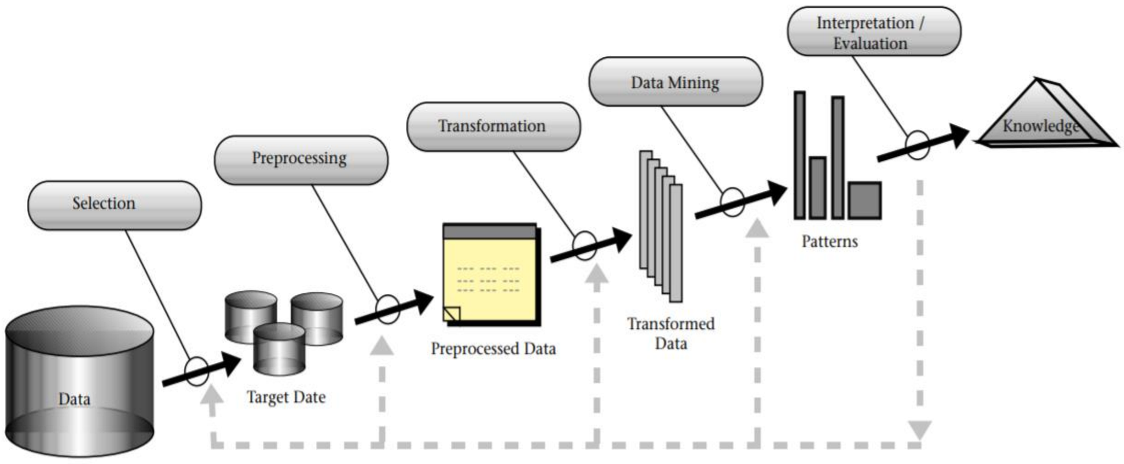


Figure 1 KDD Architecture

Project Design Flow

The goal of the project is training our model to achieve good accuracy and can have used for further improvements. KDD is standardizing process in which market can easily use by applying different ideas or models even with little knowledge of machine learning.

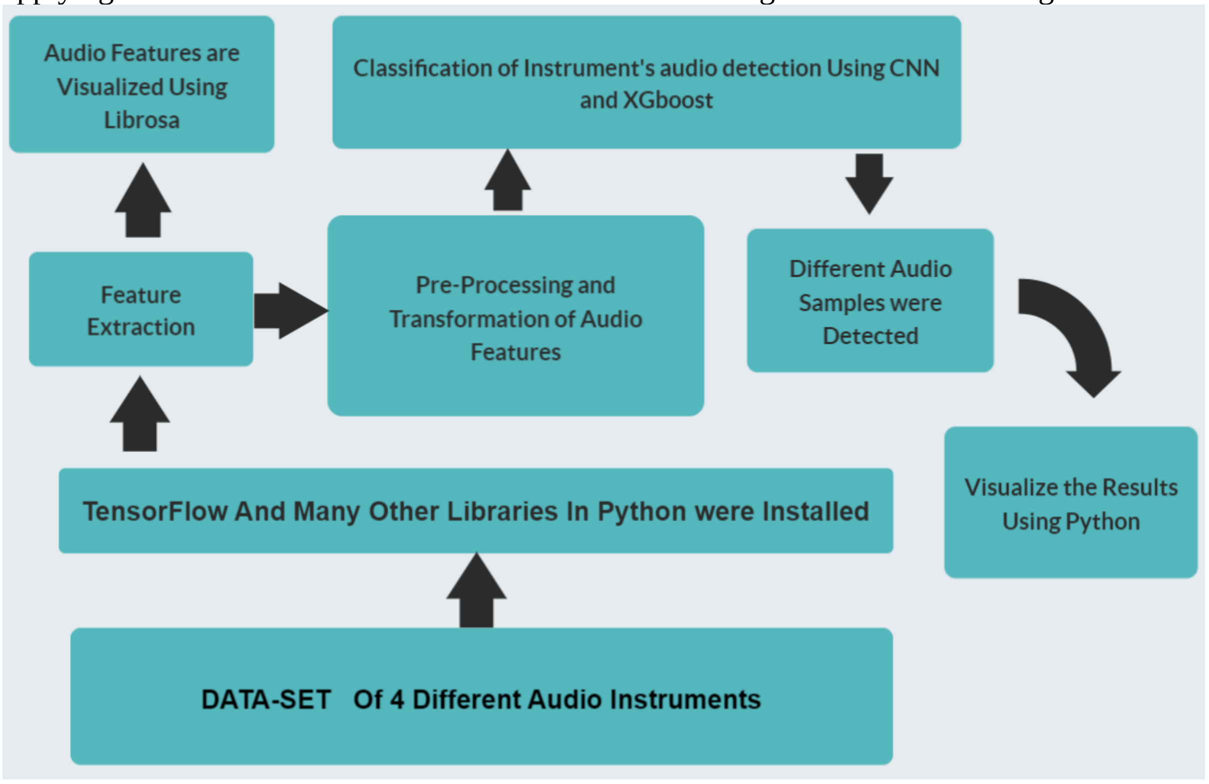

Audio Features are Visualized Using Librosa
Classification of Instrument's audio detection Using CNN and XGboost
Feature Extraction
Pre-Processing and Transformation of Audio Features
Different Audio Samples were Detected
Visualize the Results Using Python
TensorFlow And Many Other Libraries In Python were Installed
DATA-SET Of 4 Different Audio Instruments

Figure 2 Workflow Architecture

In this project, the data set is taken from the Kaggle (Kaggle.com, 2019) in which the all the audio files are in .wav format. There are 4 different classes of audio instruments Acoustic Guitar at 120 BPM with 135 audios .wav file, Acoustic Guitar at 85 BPM with 131 audios

.wav files, Drum Assorted Hit with 367 .wav files and Drum Kits with 567 audios .wav files. All the data framework is done after installing the TensorFlow (TensorFlow, 2019) which is machine learning platform and 2nd step according to our workflow. These files are extracted into the feature vectors technique with the help of Librosa Library which is used in python (Conference.scipy.org, 2019).The Librosa library change audio signals and present them in one-dimension NumPy array, here **y** is along with sample rate denoted by Hertz (in Hz). The converted audio analog to digital format which looks like in figure 3. (En.wikipedia.org, 2019)

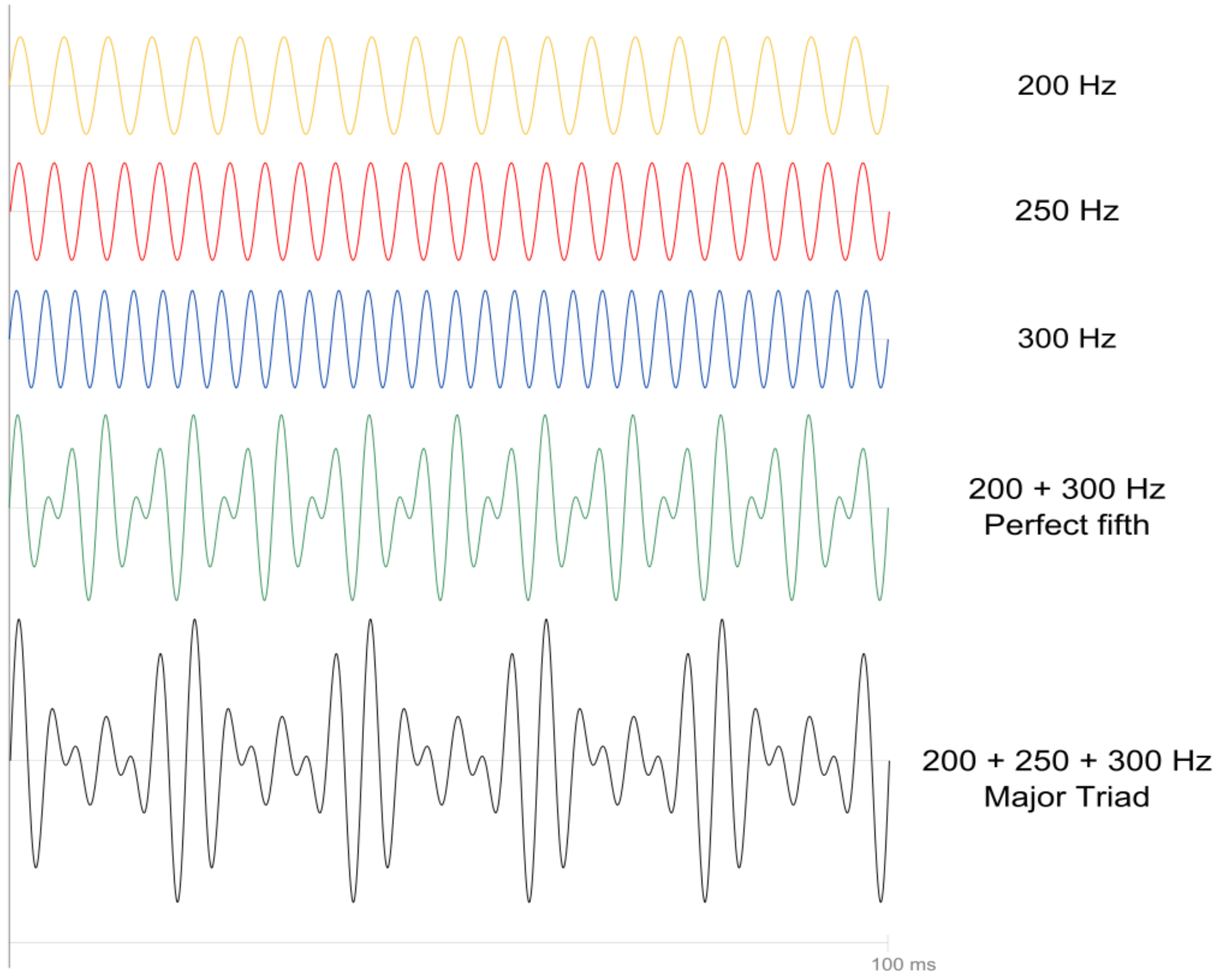


Figure 3 Wave Spectrogram Example (En.wikipedia.org, 2019)

The python libraries are used to convert further digital Hz wave format into the arrays. These transformed data now can be used for the models. The matplotlib is used for the visualization for making 2d plots of arrays. It provides simple plots with few commands. This package is used for the music and audio speech analysis in machine learning. Creating function for the spectrogram which is basically a digital format of an audio file. Creating spectrogram of all 4 different classes which are in the data set. After creating the all 4 different spectrograms, normalization is done over the mean to use a feature of neural network. The **MFCC** (Mel Frequency Cepstral Coefficients) function is used to convert the domain into the frequency domain signals using Mel Filters. The main aim to use MFCC that it has the feature which can frame the signals into the short frames which is used in speech recognition and audio recognition which was introduced by the David and Merelstein in 1980 which is famous for automatic speech recognition (ASR). This is 4th step according the architecture of workflow in figure 2 and about this function (MFCC), we will discuss further in implementation on how it works. The **parser** function is used to return the features that were created by the MFCC. The **Keras** library is used for building CNN model and XGBoost model making it simple. After creating and loading all the data and libraries, as from the block diagram above which is the process and workflow of this project, the transformed data, various model are applied like CNN model and XGBoost model which is a part of the Ensembles model. The audio files of the data set are split into 80:20 ratio in with 80 percent of data is made structured and 20 percent of data remains unstructured for the testing purpose by the models which will discuss in the implementation section. The image is in wave spectrum format. The amplitude is pitch level. This converted spectrum is used for the analysis for the identification of the audio instruments using machine learning models and compare them to check the accuracy score.

4 Design Specification

When developing the project, various approaches were taken for the identification of the musical instruments in the audio. Some of them are failed to give expected results because of the limitations of the data and some gave very good results. The two models are used for the identification of the instruments in the audio clips are **CNN** and the **XGBoost**.

Convolution Neural Network

CNN- which is upgraded version of neural network. This model was first described by the Rumelhart, Hinton and Williams. CNN has one main motive that it can detect the patterns of the image. It can take image as input and assign the learnable weights to very points in the image and these points are able to differentiate between one another. The CNN has two special layers, *convolution layer* and *pooling layer.*

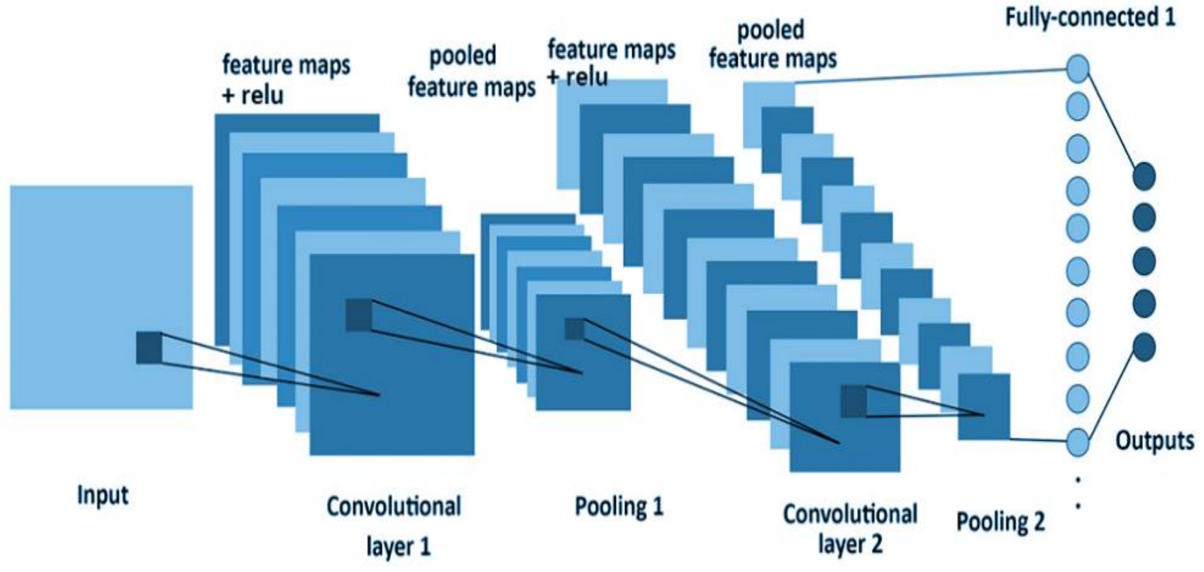


Figure 4 CNN Architecture

Convolutional Layer – This above figure 4 shows the basic architecture of the Convolution neural network. The input is image in 2 dimensions. It composed set of convolutional kernels or can say each neuron act as kernel. The small area is taken from the image as a receptive field. Convolutional layer always works on dividing the input image into receptive field and convolve them with the specific weight which comes into feature maps plus the ReLU or convolutional layer 1. Here ReLU stands for Rectified Linear Unit and it is computed function of Smaller block will help local correlated pixel value. The activation threshold is on zero. It boosts the convert rates around 6 times from the comparison with SIGMOID and TANH. (Mishra, 2019). As different part is taken from the image to extract the feature with the same weights. This function works on the numbers of layers depends on the type and size of the filters. (Anon, 2019).

Pooling Layer **–** The pooling layer work on the output of the feature maps and work on it by reducing the pixels resolutions. This helps to reduce the size for the representation which decrease the weight and amount of computational power of the machine. This process will happen on every piece of the image representation individually. There are many pooling functions like average of the neighborhood, weighted average on the distance of central pixels, in my research I used max pooling which is the most famous and most popular in process with the maximum output (Mishra, 2019). It also reduces the overfitting and increase the generalization this will help the CNN to reduce the complexity of the networks (Arxiv.org, 2019).

Neural Network

Activation Function **–** It helps in learning the patterns and act as decision function. It boosts the leaning process of the CNN adds the non-linearity and transforms the output layer. There is different activation function like **sigmoid, maxout, ReLU, tanh** is used to calculate the nonlinear features. ReLU is most preferable activation function as its boost the converting power as we talked earlier in convolution layer and it will help also in overcome the gradient problem. (Anon, 2019).

Dropout **–** It is a function which is patented by the google and used for reducing the overfitting by stopping the co-adaptations on training data. It works on the terms of dropping out the weight units which are visible or hidden in the CNN. In our case, the training stage, the dropped is **0.5** which is lower. It helps to improve the speed of the training the data and is widely used in the deep neural learning. (En.m.wikipedia.org, 2019).

Fully Connected **–** This is the final layer in convolution neural network model which consist neurons fully connected with the previous activated layers. These layer gives the output of the CNN by analyzing the previous layers output as an input and this operation is global. (En.m.wikipedia.org, 2019).

XBGoost (Extreme Gradient Boosting)

XGBoost stands for **Extreme Gradient Boosting** was developed by the University of Washington. Tianqi Chen and Carlos Guestrin was presented their paper in SIGKDD Conference 2016 (Medium, 2019). This is the most used algorithm model amongst the data scientist and can be used many situations of machine learning challenges. This is because it runs 10 times faster than any other model and it is popular for its scalability in all scenarios (Arxiv.org, 2019). There are many other boosting algorithms likes parallel boosting (Static.googleusercontent.com, 2019), Regression tree boosting (Cs.cornell.edu, 2019), Stochastic Gradient Boosting (Cse.iitrpr.ac.in, 2019) but XGBoost is one of the major boosting algorithms (Arxiv.org, 2019).
It is written on C++ platform and faster than other classifiers. The main feature of this algorithm is that it is parallelizable and can use multi core CPU and GPU of the machine which can help to execute the model quicker and can train the data much faster of very large data set. Boosting is technique which works on the principle of ensembles.

For example, there are 4 boxes and have + and – classes randomly in it
Box 1- In the first box, the split has been done D1 where left on D1 is + and on the right is – but its mis calculation shows that some of the + are also on the right side of the split line D1.
Box 2- In the box 2, gives the weight to + points which is located on the right side of the line D1 and makes the vertical line D2 with weighted classes but here also D2 says that on the left side there is 3 – classes which is incorrectly classified.
Box 3 – In this box, the classifier gives weight to those – 3 classes and makes D3 horizontal split line but it still fails to fulfill the desired requirement of having the + classes on the wrong side.
Box 4 – Now the weighted classifier is added, and we can see that it is perfectly does the job to classifying all the points accurately (DataCamp Community, 2019)

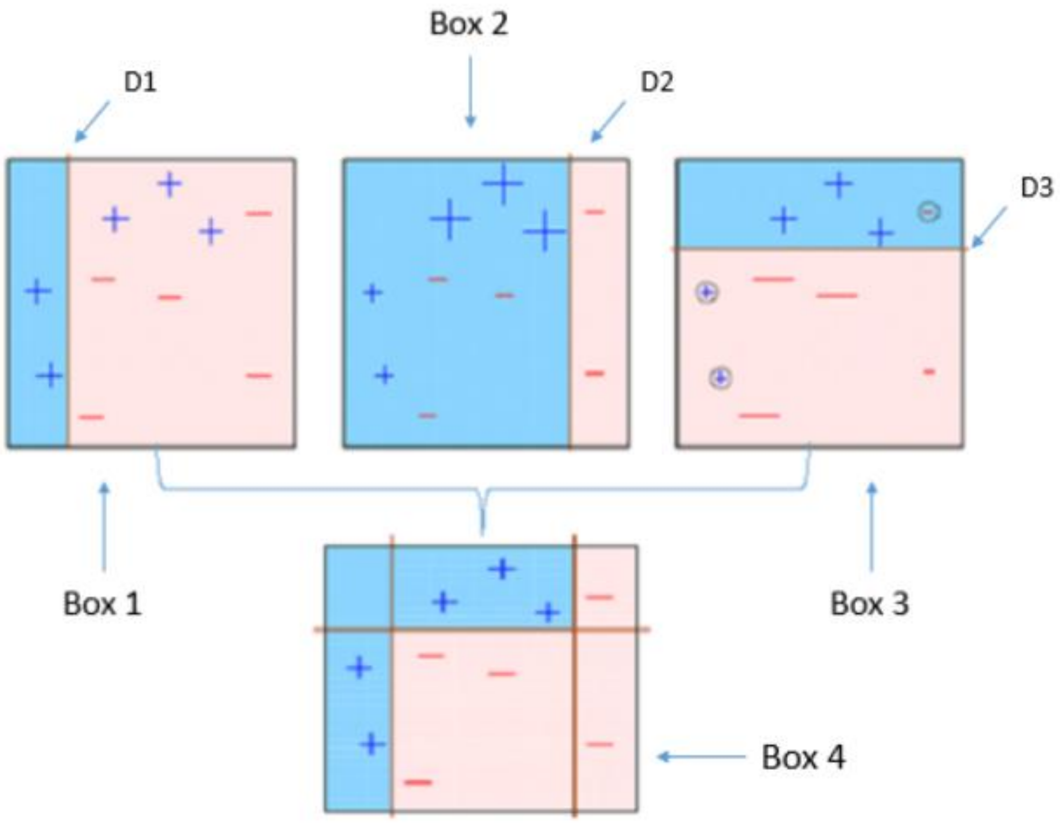


Figure 5 Box Example of XGBoost [29]

This is the basic idea of the boosting algorithms which was built for the weak and eliminates misclassification errors of the models. The XGBoost is used to solve regression problems.

5 Implementation

We will discuss the implementation of the model used in this research and see what results it gives from the audio data. The discussion will be done from the scratch of implementing the methods and the basic background structure that were involved in the model.

Loading Packages

Before going any further, many packages should have to install on machine. **TensorFlow** is one of the main platforms for the whole process. It is an artificial intelligence library that uses in detection of audio video, visualizations of graphs, text-based detection. It is used where developers are creating many layers of neural network. For mathematical calculations, **NumPy** is install and **SciPy.fft** (Fast Fourier Transform) were imported into python. It is used to analysis converted signal of original domain to frequency domain representation and vice versa. One of the main packages is **Librosa,** it is used to analysis the audio music transformed data and Librosa.display and Matplot is used to show chronograph, spectrograph of the audio file which is transformed and used to compare and in identification of the instruments of an audio clips. Sklearn.metrics is used for the accuracy score of the model and for creating the confusion matrix. **Keras** is also used, it runs on top of the TensorFlow. It helps in building the blocks of the neural network like dropout, activation function, dense function of the network. It also fastens the process of the code process on image and text analysis.

Now creating the function of spectrogram which shows the amplitude and the frequency in Hz. After loading the file of acoustic guitar of 85 bpm, it gives the sample rate output is 22050 hertz and the sampling rate is 124518 meters at amplitude graph. As we discuss earlier, sound is analog signal and to analyze the audio, the converted format is needed that represent the NumPy array format to sample the original signal. So here we have now sample rate per second with the amplitude spectrogram. By plotting the figure size of graph is (14, 8) and adding the subplot of (211) in python. And for the 2nd graph of same audio file is spectrogram which shows frequency, we add subplot (212). Then we get both graphs of the audio file together.

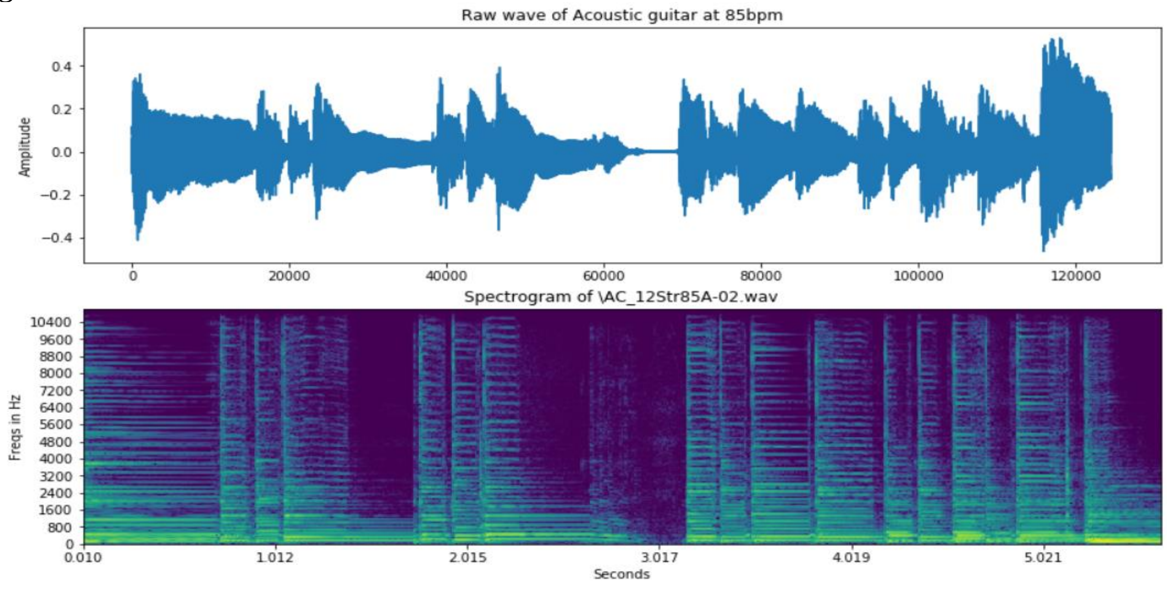


Figure 6 Acoustic Guitar at 85 BPM Amplitude and Spectrogram graph

We can see from here it's really a complex form of digital audio format. The above figure 6 shows two graphs in which the upper one shows the amplitude in meters of the selected audio file and below that spectrogram is shown frequency in hertz per second. In that way we must repeat the process of other remaining 3 different of audio instrument samples which are acoustic guitar at 120 BPM, Drum Assorted Hit sample and Drum Kit sample.

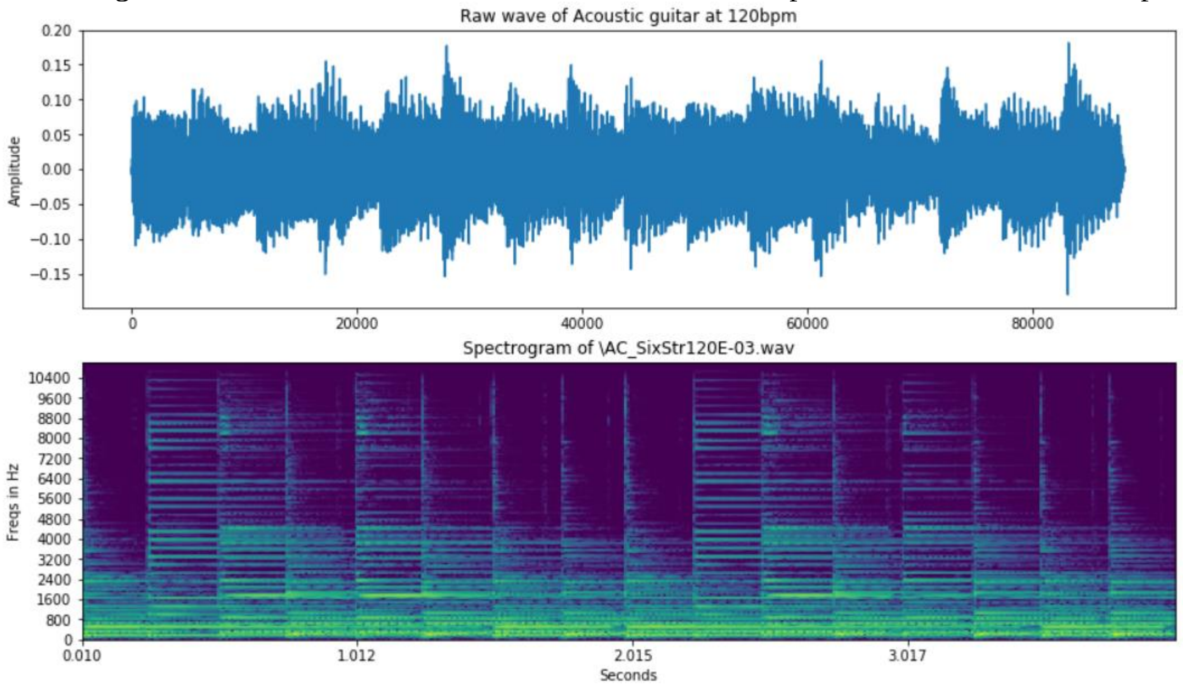


Figure 7 Acoustic Guitar at 120 BPM Amplitude and Spectrogram graph

The above figure 7 shows the amplitude graph and spectrogram graph of Acoustic guitar at 120 BPM after loading the file (AC_SixStr120E-03.wav) from the trained folder of the data.

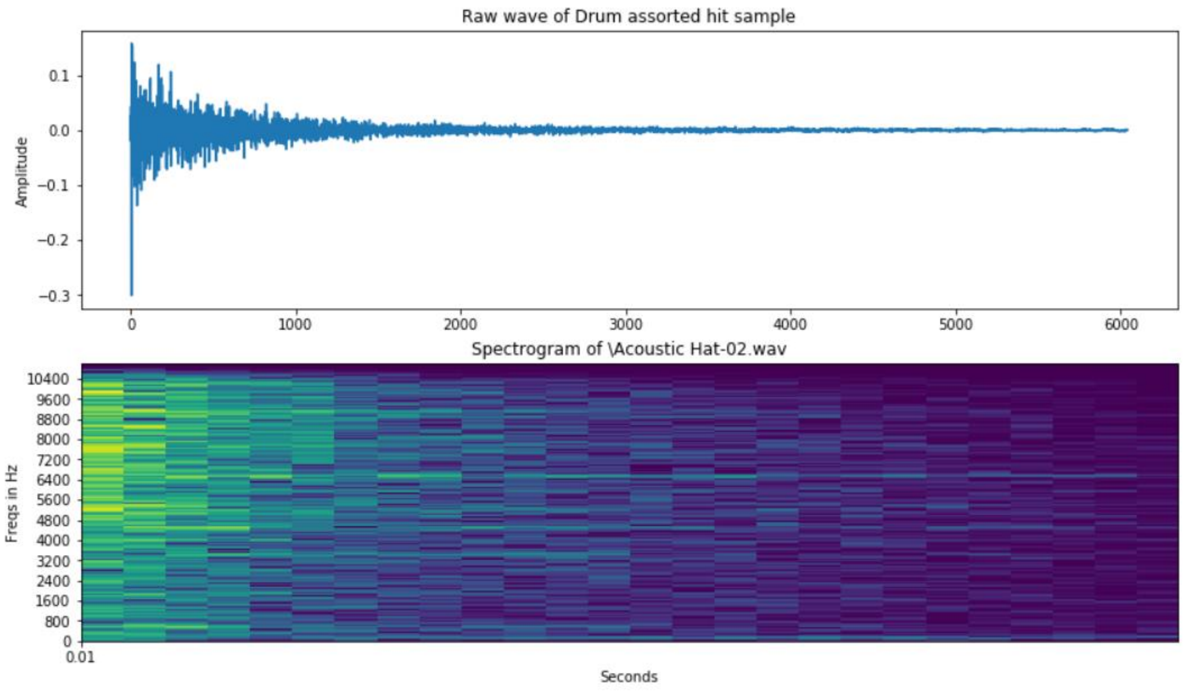


Figure 8 Drum Assorted Hit Amplitude and Spectrogram graph

The above figure 8 shows the Amplitude rate and the spectrogram graph frequency per second of the Drum Assorted Hit when (Acoustic Hat-02.wav) file is loaded.

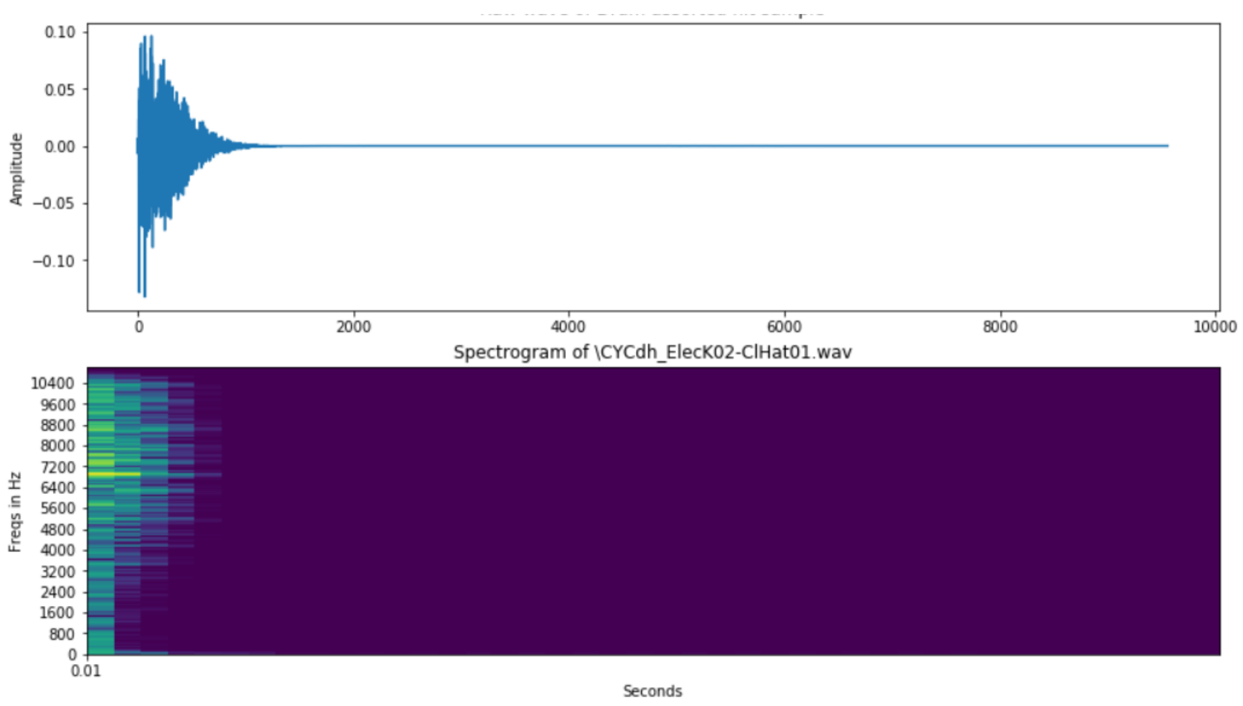


Figure 9 Drum Kit Amplitude and Spectrogram graph

The above figure 9 shows the amplitude graph and below that the spectrogram graph is given of Drum Kit, the file name (CYCdh_ElecK02-ClHat01.wav).

As we can see from the above graphs, the image looks almost similar and it is difficult to see the difference in between the 4 different audio instruments but the machine knows the difference because machine can structure this form into the array form. This is sample of array of Acoustic Guitar at 120 BPM which look like this

```
Array ([-4.2533859e-05, -3.0661986e-04, -5.5540766e-04, ...,
        3.7841720e-04, 2.4527789e-04, 1.4289166e-04], dtype=float32)
```

This array formed will be used to for the identification of the audio instrument by the algorithm that we used. Now normalizing sample number which is basically in array form, the spectrogram over the mean and the standard deviation to use this feature of neural network. For this there is simple method which is

```
mean = np.mean(spectrogram, axis=0)
std = np.std(spectrogram, axis=0)
spectrogram = (spectrogram - mean) / std
```

From and after this we will make Mel Power Spectrogram graph of Hertz vs Time to use the feature in neural network. after creating function, we get the Mel spectrogram of figure size (12,4) which is Hertz vs Time from figure 10.

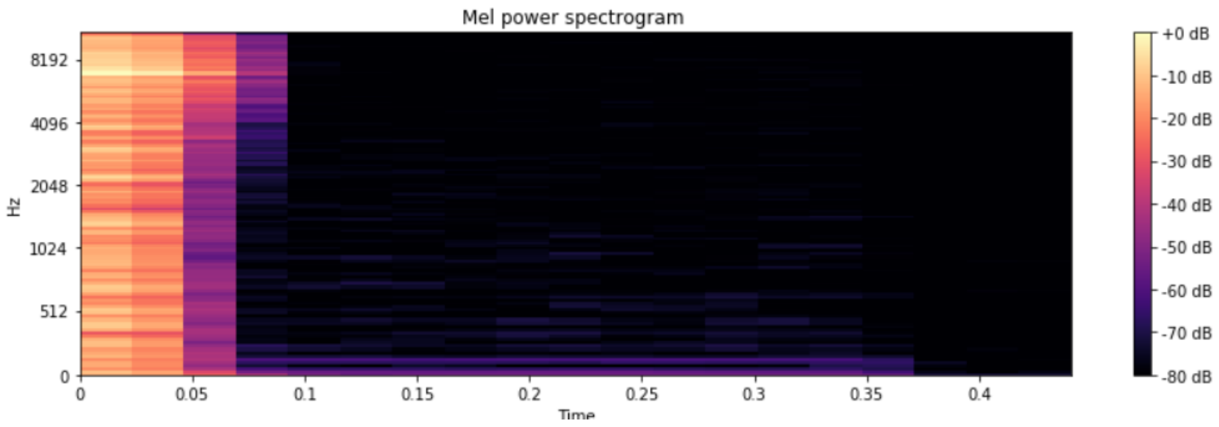


Figure 10 Mel power Spectrogram graph

Now from here we can use MFCC (Mel- frequency Cepstral Coefficients) feature to use in Neural network to improve the identification of the audio signals in the noisy signals (Fayek, 2019). By writing the function mfcc = librosa.feature.mfcc(S=log_S, n_mfcc=10), We can get this mel- spectrogram (figure 10) which is much more improved.

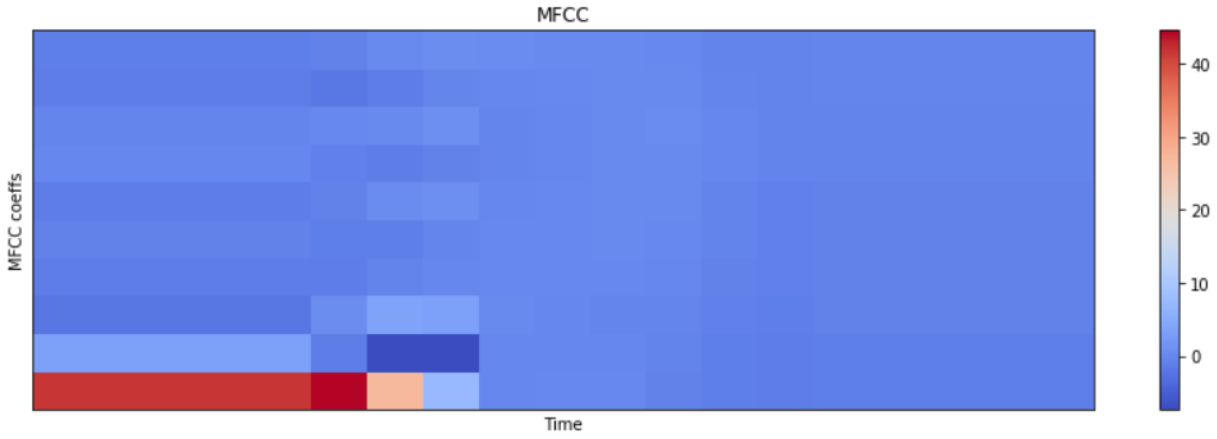


Figure 11 MFCC Spectrogram

Now we have everything is ready to compare the graphs. To show how much the difference between the two audio instruments amplitude graph and spectrogram graph looks like, we must load files of two different audio instruments in my case I used Guitar and the Drum. The figure size remains same as before which is (14,8).

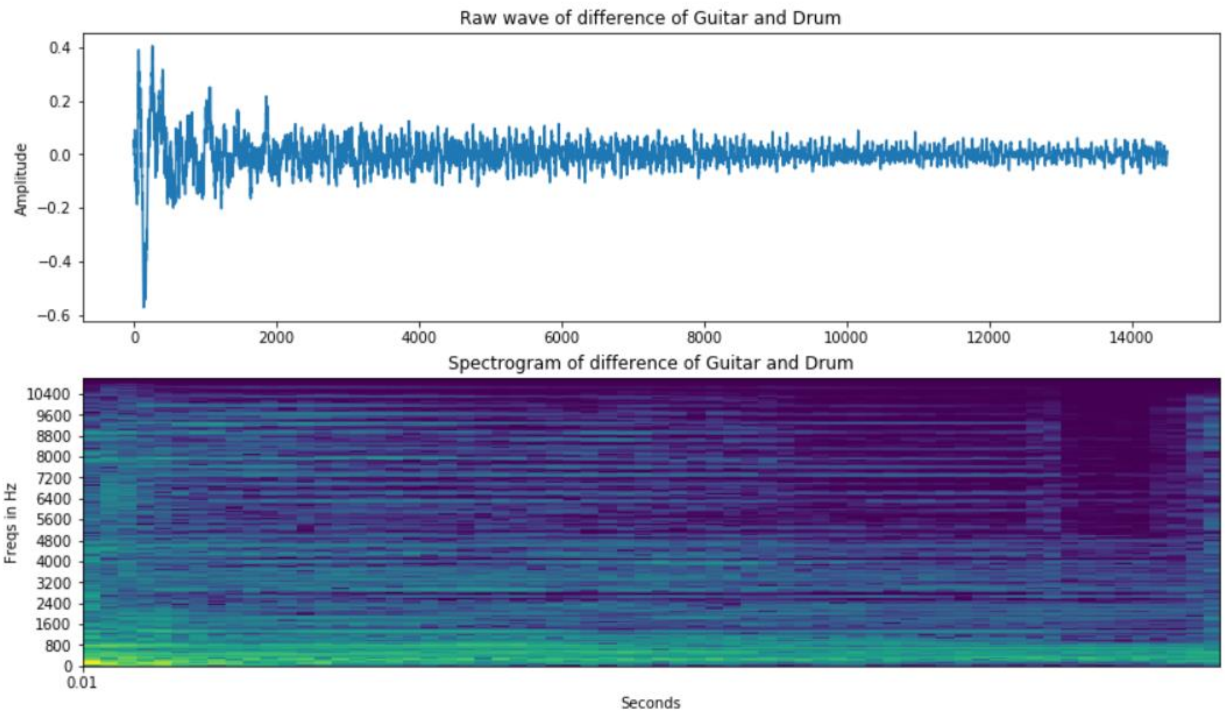


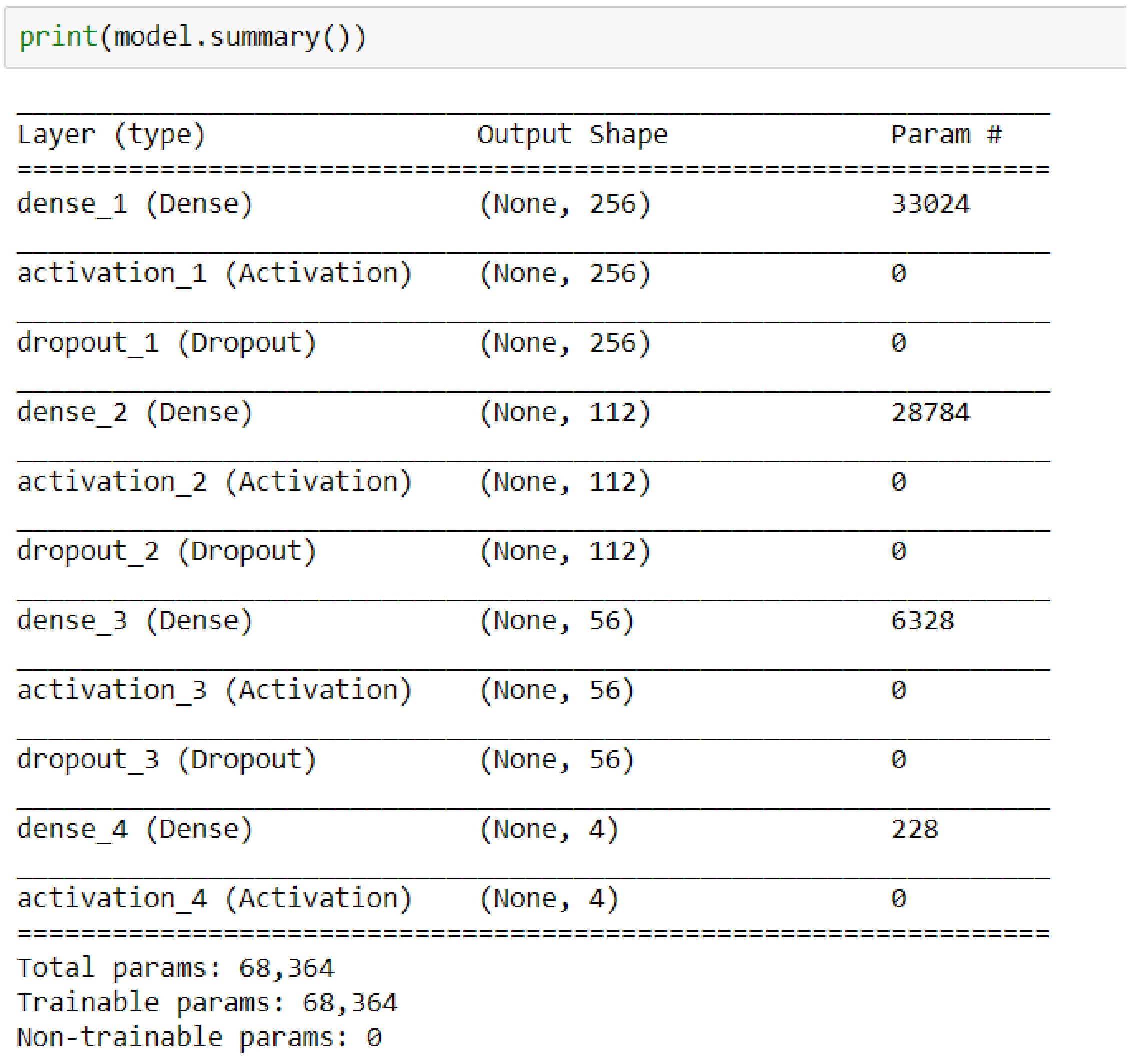
Figure 12 Difference Between the Guitar and drum Spectrogram graph

Comparing these two graphs, we can now ready to implement our two model. We load 3 files train_labels.csv, test_labels.csv, and valid_labels.csv which are made during the data set segregation during structuring the unstructured data and giving labels them. The parser function is used to load and extract the features from the loaded data. The Kaiser technique is used for making faster extraction. This is done for all 3 .csv files (train_labels.csv, test_labels.csv and valid_labels.csv) turn by turn. Here it takes the feature and labels and combine them together. Now with the help of LabelEncoder function, it gives the array matrix which looks like this of train_y (Train data)

```
Array ([[0., 1., 0., 0.],
       [0., 1., 0., 0.],
       [0., 1., 0., 0.],
       ...,
       [0., 0., 0., 1.],
       [0., 0., 0., 1.],
       [0., 0., 0., 1.]], dtype=float32)
```

1st Model CNN

Now developing the model of CNN. 1st importing the libraries like Keres, Keras.model, Sequential, Dense, Activation, dropout, keras.layers, convolution2d, maxpooling2d and metrics. Here the dense is 256, activation is ReLU and dropout is 0.5. after running it, the total parameters of CNN are 68,364 in which they are all trainable and none of them are non-trainable.



```
print(model.summary())
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_1 (Dense)              (None, 256)               33024     
_________________________________________________________________
activation_1 (Activation)    (None, 256)               0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 112)               28784     
_________________________________________________________________
activation_2 (Activation)    (None, 112)               0         
_________________________________________________________________
dropout_2 (Dropout)          (None, 112)               0         
_________________________________________________________________
dense_3 (Dense)              (None, 56)                6328      
_________________________________________________________________
activation_3 (Activation)    (None, 56)                0         
_________________________________________________________________
dropout_3 (Dropout)          (None, 56)                0         
_________________________________________________________________
dense_4 (Dense)              (None, 4)                 228       
_________________________________________________________________
activation_4 (Activation)    (None, 4)                 0         
=================================================================
Total params: 68,364
Trainable params: 68,364
Non-trainable params: 0
```

Figure 13 Model Parameters

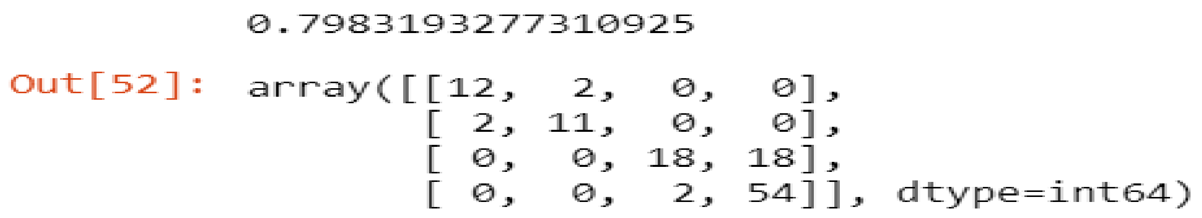
The batch size is 3 and epochs is 15 we can change epochs for better result, but it will take time to process the data. But at 15, here it runs 15 times for training the model, but the validation loss did not improve from 8.11416. It is important to validation should near about 0.5. after doing this it gives the test accuracy which we will discuss in evaluation.

2nd Model XGBoost

For applying this model, we must load all the files again (test, train and valid) and extract feature using MFCC which will give an array of features and array of labels. Then it shows the sample output. After indexing and selecting the data of test, train, valid to convert the column into numeric value and giving labels to category wise of 4 object that we have which is acoustic_guitar_120bpm, acoustic_guitar_85bpm, drum_assorted_hit, drum_kits. Applying the model on the data that we prepared according to the architecture of our model that we discussed before. Setting XGB classifier with learning rate of 0.025, estimaters at 700, and max depth of trees at 5. Then print the accuracy rate that will discuss in the evaluation.
The identification of the audio instrument between the 4 different classes has been done.

6 Evaluation

According to block diagram of the proposed research, the evaluation is carried out in last. The results of the two model CNN and XGBoost comes out and we found that CNN fails to deliver the expected accuracy because the of the limited dataset. CNN getting confused in identification of the instruments of 4 different classes. Test accuracy on normal test set: 11.7647%. From here we can see that because of the less data CNN in not able to identify the audio instruments and cannot predict the difference between them. The accuracy of the CNN is 11.76% on this data set. It's not that what we are expected and after analyzing it thoroughly, we find the data have less audios of different classes. If the data set have more audio clips to train and test, then over the fine tuning to the network we can get much better accuracy in the identification of the audio instruments.
Here **XGBoost** shows good accuracy here in the prediction as it is the best classifier for the extracting features of the frequencies from the audio signals and on these features XGBoost shows its performance.



```
         0.7983193277310925

Out[52]: array([[12,  2,  0,  0],
                [ 2, 11,  0,  0],
                [ 0,  0, 18, 18],
                [ 0,  0,  2, 54]], dtype=int64)
```

Figure 14 Confusion Matrix Array

The accuracy rate of the XGBoost model is 79.8% percent. Here is the array of the XGBoost prediction where Y axis is actual rate and x axis shows the prediction. From this we get the confusion matrix.

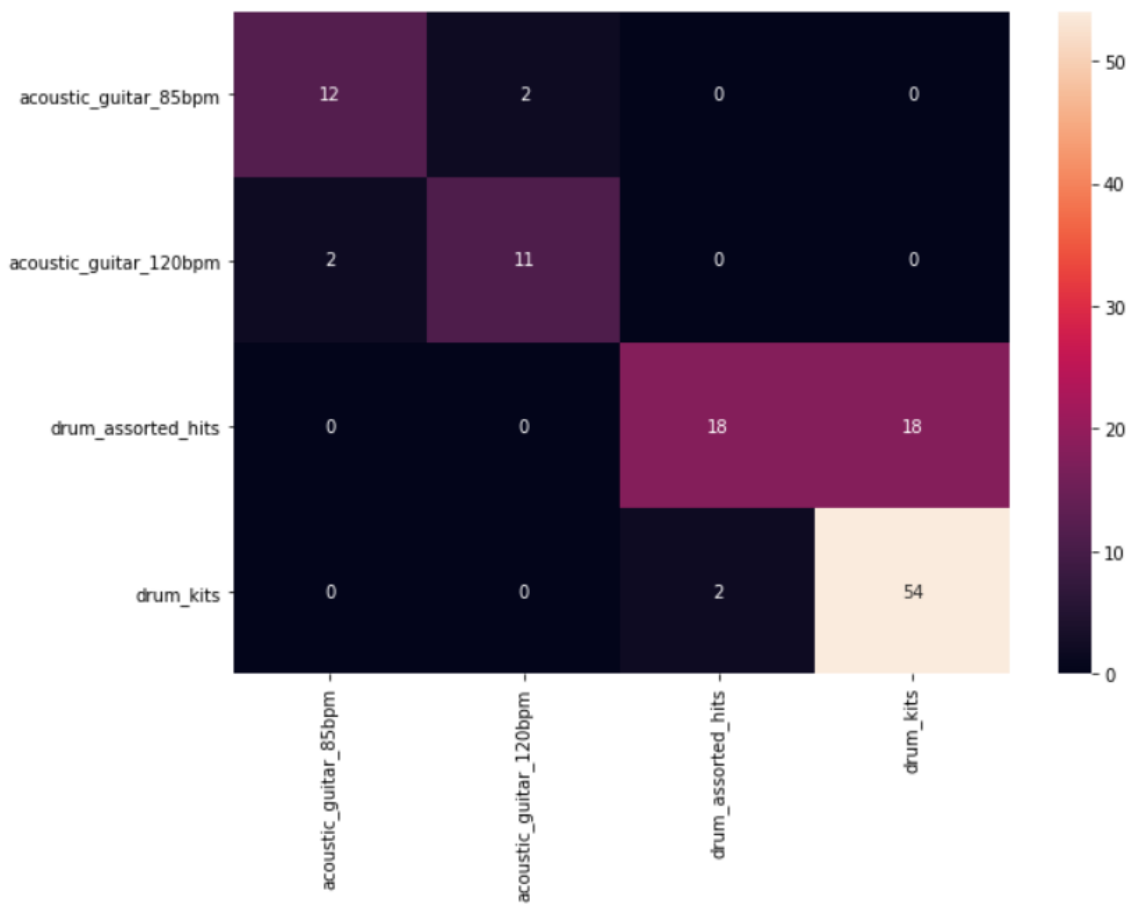
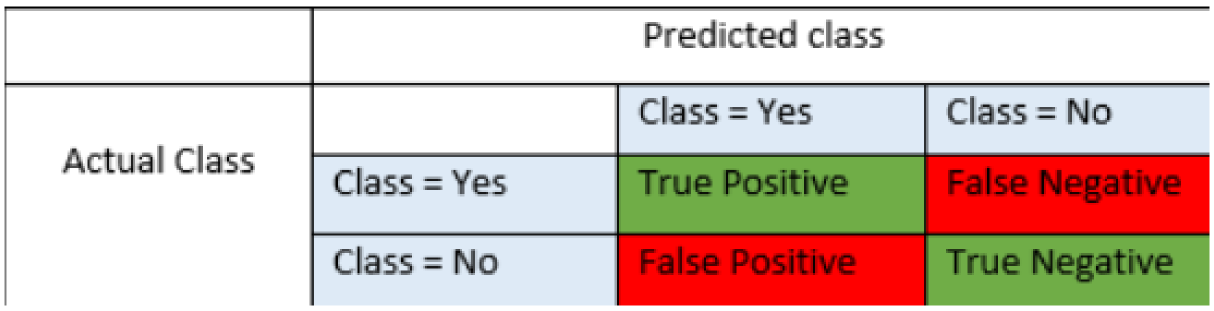


Figure 15 Results of Confusion Matrix

From the above confusion matrix, we can easily tell the performance of the model on the tested data.



	Predicted class	
Actual Class	Class = Yes	Class = No
Class = Yes	True Positive	False Negative
Class = No	False Positive	True Negative

Figure 16 Confusion Matrix Example

The green box shows the true positive and true negative of the observation that are correctly predicted. Red box of false positive and false negative that we should minimize in the observation.

True Positive - it shows the prediction of actual class is true and the value of predicted class is also true.

True Negative - it shows the actual class value is false and predicted value is also false

False Positive - when observation of class is false, and the predicted class is true

False Negative - when the actual class is false, and the prediction is also false.

On the above observation, following parameters can be calculated.

Precision – It is calculated to know the positive predicted value and the formula is

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall - It is also known as true positive rate or sensitivity for perfectly classified positive classes.

$$\text{Recall} = \frac{tp}{tp + fn}$$

Accuracy

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

F1 Score

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

6.1 Discussion

There are two models used in this research paper.1st is CNN (Convolutional Neural Network) and 2nd is XGBoost which is a part of ensembles. The 1st model is failed to deliver the expected results. On finding and searching about this failed model, I found that this happened because of the smaller data and there is less classes in the dataset. CNN model needs big data set to train the model. It is getting confused in Acoustic guitar 85 BPM and Acoustic guitar 120 BPM. Having big data set enough can give CNN much more better results as studied in many past research papers. The audio of Drum assorted hit and drum kit also get confused in the XGBoost model but on very low scale but in between the acoustic guitar 85 BPM and acoustic guitar 120 BPM shows very good results. The good thing of XGBoost is the performance of the model is depending on what parameters are used to implement. We used the learning task parameter which is known as multi:softmax parameters. The multi:softmax parameter is specifically made for the learning task and corresponded learning objective. This parameter decides on the learning of data scenarios. CNN also takes time to implement as in this case there all files are readied many times over and over but in XGBoost model it is little bit faster because the use of parser technique. This technique is gained much more popularity in algorithm choice than others as the stable version of XGBoost is recently released in May 2019.

7 Conclusion and Future Work

This research shows many models that shows poor results, and some shows good result in terms of identification of audio instruments like in case of CNN shows very bad results and XGBoost shows good accuracy. It can solve real world problem with scalable and minimal amount of resource. This algorithm can also apply on different length of audio clips, we used on 1 seconds of length to determine the difference in the model to make it fast and accurate. In future we can add more and more instrument into the data base and make fingerprints of the audio clips to make space saving data base and compress them into .wav format which is better than mp3 format without losing any quality of the audio clips. Adding more complex pattern and parallel pattern will also help to make our model robust into the identification with any noise included in it. From business perceptive, the industry will also easily adopt the technology for further research because the concept of identification of audio is old and famous like in Shazam and google also prepared an audio detection application. The audio instrument identification helps further to musician and composers to improve their skills.

References

1. Pdfs.semanticscholar.org. (2019). [online] Available at: https://pdfs.semanticscholar.org/9618/ecdd92c85f2813dc83e3470405fb27552bc8.pdf?_ga=2.175902048.338553485.1564965372-1597238659.1564834314 [Accessed 11 Aug. 2019].
2. Eronen, A. and Klapuri, A. (n.d.). Musical instrument recognition using cepstral coefficients and temporal features. *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*.
3. Pdfs.semanticscholar.org. (2019). [online] Available at: https://pdfs.semanticscholar.org/8c03/0a736512456e9fd8d53763cbfcac0c014ab3.pdf?_ga=2.64366639.1260401242.1565484433-1365741540.1565484433 [Accessed 11 Aug. 2019].
4. Mac.citi.sinica.edu.tw. (2019). [online] Available at: http://mac.citi.sinica.edu.tw/~yang/pub/yu14icassp.pdf [Accessed 11 Aug. 2019].
5. Anon, (2019). [online] Available at: https://www.researchgate.net/publication/309155620_Low-latency_sound_source_separation_using_deep_neural_networks [Accessed 11 Aug. 2019].
6. Arxiv.org. (2019). [online] Available at: https://arxiv.org/pdf/1804.01149.pdf [Accessed 11 Aug. 2019].
7. Fan, Y., & Feng, S. (2016). A Music Identification System Based on Audio Fingerprint. 2016 4th Intl Conf on Applied Computing and Information Technology/3rd Intl Conf on Computational Science/Intelligence and Applied Informatics/1st Intl Conf on Big Data, Cloud Computing, Data Science & Engineering (ACIT-CSII-BCD). doi:10.1109/acit-csii-bcd.2016.076
8. Quaero.org. (2019). [online] Available at: http://www.quaero.org/media/files/bibliographie/fenet_a_scalable_audio_ismir2011.pdf [Accessed 11 Aug. 2019].
9. Yang, G., Chen, X., & Yang, D. (2014). Efficient music identification by utilizing space-saving audio fingerprinting system. 2014 IEEE International Conference on Multimedia and Expo (ICME). doi:10.1109/icme.2014.6890236
10. Pdfs.semanticscholar.org. (2019). [online] Available at: https://pdfs.semanticscholar.org/4ba5/990bb9e0ac8106e4ce8adf4a3c30eb9ac9a7.pdf?_ga=2.%2013564031.971215055.1554546610-339426852.1554546610 [Accessed 11 Aug. 2019].
11. Google.ie. (2019). *Redirect Notice*. [online] Available at: https://www.google.ie/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=2ahUKEwjJs8uajO_jAhVks3EKHRwtAW4QFjAAegQIARAC&url=https%3A%2F%2Fwww.aaai.org%2Fojs%2Findex.php%2Faimagazine%2Farticle%2Fview%2F1230%2F1131&usg=AOvVaw37VrwwiSqRBvyce_1h2bhP) [Accessed 11 Aug. 2019].
12. Kaggle.com. (2019). *Audio_data*. [online] Available at: https://www.kaggle.com/loganclaws/audio-data [Accessed 11 Aug. 2019].
13. TensorFlow. (2019). *Why TensorFlow | TensorFlow*. [online] Available at: https://www.tensorflow.org/about/ [Accessed 11 Aug. 2019].
14. Conference.scipy.org. (2019). [online] Available at: http://conference.scipy.org/proceedings/scipy2015/pdfs/brian_mcfee.pdf [Accessed 11 Aug. 2019].

15. En.wikipedia.org. (2019). *Harmony*. [online] Available at: https://en.wikipedia.org/wiki/Harmony#/media/File:Major_triad.svg%20[Accessed%207%20Apr.%202019 [Accessed 11 Aug. 2019].
16. Mishra, M. (2019). *Convolutional Neural Networks, Explained*. [online] Datascience.com. Available at: https://www.datascience.com/blog/convolutional-neural-network [Accessed 11 Aug. 2019].
17. Anon, (2019). [online] Available at: https://www.researchgate.net/publication/277411157_Deep_Learning/link/55e0cdf908ae2fac471ccf0f/download [Accessed 11 Aug. 2019].
18. Mishra, M. (2019). *Convolutional Neural Networks, Explained*. [online] Datascience.com. Available at: https://www.datascience.com/blog/convolutional-neural-network [Accessed 11 Aug. 2019].
19. Arxiv.org. (2019). [online] Available at: https://arxiv.org/ftp/arxiv/papers/1901/1901.06032.pdf [Accessed 11 Aug. 2019].
20. Anon, (2019). [online] Available at: https://www.researchgate.net/publication/220355039_The_Vanishing_Gradient_Problem_During_Learning_Recurrent_Neural_Nets_and_Problem_Solutions [Accessed 11 Aug. 2019].
21. En.m.wikipedia.org. (2019). *Convolutional neural network*. [online] Available at: https://en.m.wikipedia.org/wiki/Convolutional_neural_network#Dropout [Accessed 11 Aug. 2019].
22. En.m.wikipedia.org. (2019). *Convolutional neural network*. [online] Available at: https://en.m.wikipedia.org/wiki/Convolutional_neural_network#Dropout [Accessed 11 Aug. 2019].
23. Medium. (2019). *XGBoost Algorithm: Long May She Reign!*. [online] Available at: https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d [Accessed 11 Aug. 2019].
24. Arxiv.org. (2019). [online] Available at: https://arxiv.org/pdf/1603.02754.pdf [Accessed 11 Aug. 2019].
25. Static.googleusercontent.com. (2019). [online] Available at: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/36296.pdf [Accessed 11 Aug. 2019].
26. Cs.cornell.edu. (2019). [online] Available at: http://www.cs.cornell.edu/~kilian/papers/fr819-tyreeA.pdf [Accessed 11 Aug. 2019].
27. Cse.iitrpr.ac.in. (2019). [online] Available at: http://cse.iitrpr.ac.in/ckn/courses/f2012/thomas.pdf [Accessed 11 Aug. 2019].
28. Arxiv.org. (2019). [online] Available at: https://arxiv.org/pdf/1603.02754.pdf [Accessed 11 Aug. 2019].
29. DataCamp Community. (2019). *Using XGBoost in Python*. [online] Available at: https://www.datacamp.com/community/tutorials/xgboost-in-python [Accessed 11 Aug. 2019].
30. Fayek, H. (2019). *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*. [online] Haytham Fayek. Available at: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html#fn:1 [Accessed 11 Aug. 2019].