INSTRUMENTAL SOUND CLASSIFICATION; A WAY TO SPEECH RECOGNITION USING CONVOLUTIONAL AND RECURRENT NEURAL NETWORK

BY

MD. JEWEL

ID: 162-15-8071

AND

ABDULLAH AL FOYSAL

ID: 162-15-7683

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Raja Tariqul Hasan Tusher

Sr. Lecturer

Department of CSE, Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

JULY 2020

APPROVAL

This Project titled **“Instrumental Sound Classification; A Way to Speech Recognition Using Convolutional & Recurrent Neural Network”**, submitted by Md. Jewel and Abdullah Al Foysal to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering (BSc) and approved as to its style and contents. The presentation has been held on 8th July 2020.

BOARD OF EXAMINERS

[illegible]

Dr. Syed Akhter Hossain **Chairman**
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

[illegible]

Dr. Sheak Rashed Haider Noori **Internal Examiner**
Associate professor &Associate Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

[illegible]

Md. Zahid Hasan **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Md. Motaharul Islam.

Dr. Md. Motaharul Islam **External Examiner**
Professor
Department of Computer Science and Engineering
United International University

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Raja Tariqul Hasan Tusher, Sr. Lecturer, Department of CSE, Daffodil International University.** We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:

[illegible]

Raja Tariqul Hasan Tusher

Senior Lecturer

Department of CSE

Daffodil International University

Submitted by:

Jewel

(Md. Jewel)

ID: 162-15-8071

Department of CSE

Daffodil International University

Foysal

(Abdullah Al Foysal)

ID: 162-15-7683

Department of CSE

Daffodil International University

ACKNOWLEDGEMENT

First, we express our heartiest thanks and gratefulness to the Almighty for His divine blessing makes us possible to complete the final year project/internship successfully.

We grateful and wish our profound indebtedness to **Raja Tariqul Hasan Tusher, Sr. Lecturer, Department of CSE, Daffodil International University.** Deep Knowledge & keen interest of our supervisor in the field of "*Deep Learning*" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express our heartiest gratitude to Allah and Head**,** Department of CSE, for his kind help to finish our project and also to other faculty members and the staff of CSE department of Daffodil International University.

We would like to thank our entire course mate at Daffodil International University, who took part in this discussion while completing the course work.

Finally, we must acknowledge with due respect the constant support and patients of our parents.

ABSTRACT

Sound or audio classification is always a challenging task since it’s not always handy to collect the dataset. Even after the collection, it’s not guaranteed that any specific model of Machine learning or deep learning will perform better. In this project, we have designed a novel approach to classify musical instruments of different categories using Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN). Nowadays acoustic scene sound, music, and speech are considered to be handled in the domain of audio since they have resemblance for the application of digital signal processing (DSP). Recently the domain of image classification has grown up very swiftly by the application of different machine learning models, so it is high time to study numerous extensible models in the domain of audio classification. But collecting audio data is not always feasible again training a network based on a small dataset and get the maximum accuracy is a quite challenging task in the deep learning approaches. We took that challenge and successfully built two models based on convolutional and recurrent neural networks. In order to classify instrumental sounds, initially, we have to extract different features from the audio samples. We used the best feature extraction technique which is Mel Frequency Cepstral Coefficient (MFCC). MFCCs work like a human auditory system, that’s why it provides extremely typical features from audio or music samples. However, we got around 95.76% and 87.62% accuracy for convolutional and recurrent neural network models respectively on 10 different music instrumental classes. Applying the same techniques we tried to classify human emotion from their speech. It’s easy for humans to recognize others' emotions hearing their voice but for machines, it gives a nightmare experience. So we analyzed different techniques to perform speaker discrimination and speech analysis to find efficient algorithms to perform this task.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
Introduction

1.1 Introduction

Recently, the application of DSP (Digital Signal Processing) for musical instruments and analysis of speech has been growing rapidly. Moreover, online access music data is being demanded day by day on the internet at the same time the computational tools of development for summarization, analysis, classification with indexing has also increased. The solutions for a music-related task is provided by MIR (Music Information Retrieval). The sound classification of the musical instrument is a subtask of this MIR which also deals with the identification of musical instruments [1]. The application of instrumental music analysis can be found in polyphonic audio recognition and separation, music transcription (automatic), beat tracking, and many more [2].

Generally, the musical instrumental sounds might carry enormous information on nearby occurring events. The human can easily perceive and discern many of these sound events but still, machine lags far behind than human in the automatic sound processing which is known as state of the art. So in order to develop a robust system, further research is a demand of modern time which will allow the system to recognize a huge range of audios including different musical instruments in an efficient method [3].

Broadly speaking, different types of audio classification tasks can be classified into three sub-categories including speech recognition (acoustic model), music classification, and acoustic scene classification. Since they have different types of signal characteristics, each of the subcategory tasks is different in the input features of audios but the recent advancement in deep learning has profoundly encouraged to build a single audio or music instrumental model for the application in numerous cross-domain tasks. For example, a CRNN (Convolutional Recurrent Neural Network) model was used by Adavanne et. al. for the detection of the sound event [4], audio classification of bird [5], and musical emotion recognition [6]. However, relying on the task, most of the models of audio classification use a range of suboptimal settings for the representation of time-frequency as input in the terms of filter bank size and type, time-frequency magnitude compression and resolution. This utterly influences in the architecture of the model, for example, convolutional layer (2D or 1D) choices, and the shape of filter [7]. A waveform based model can solve issues like that which takes and processes raw input signals. Recently, Schrauwen and Dieleman built CNN models using the raw waveforms as input for the task of auto-tagging music [8].

For speech recognition, CLDNN (Convolutional Long short-term memory Deep Neural Network) was used by Sainath et. al. [9]. DCNN (Deep Convolutional Neural Network) was used by Dai et. al. with the residual connections in order to recognize environmental sounds [10]. Most of them used filters based frame-level (samples are long enough to several hundred) which were cautiously configured for the first convolutional layer.
In this paper, we have demonstrated the method of extracting musical instrument features and then based on these extracted features the class of the instrumental audio was classified. The block diagram for this basic operation is given in the fig.1.1.1 step by step,

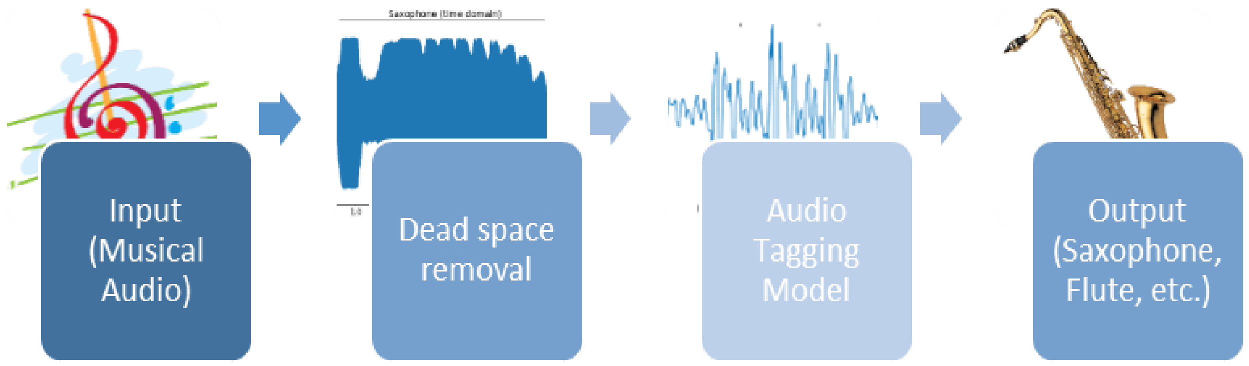


Fig 1.1.1: Basic audio tagging and feature extraction

1.1.1 Human Emotion Recognition

For human voice, some points are to be noted.

- Although emotion detection from the speech is a relatively new field of research, it has many potential applications. For the interaction with computers and humans or human to computer systems, improved services could be provided by computers by being more adaptive to their various emotions. In virtual worlds, the emotion recognition system could help in simulation to make more realistic avatar interaction.

- For detecting emotion the body of work in speech is generally quite limited. As a result, the researchers are debating on what features have more influence to recognize the emotion in speech. They believe they have considerable uncertainty to the best algorithm for the classification of emotion, and especially in which emotions are to class together.
- For a machine to understand the mindset/mood of the humans through a conversation, it needs to know who is interacting in the conversation and what is spoken, so we implement a speaker and speech recognition system first and perform speech analysis on the data extracted from prior processes.
- Understanding the mood of humans can be very useful in many instances. For example, nowadays those computers which possess the ability of perceiving and responding to human for non-lexical communication like emotions. For such a case, after the detection of human emotions, the machine could gain the customization power to change the settings according to the user needs and preferences.

1.2 Motivation

Nowadays machine learning along with deep learning has become more popular and various types of useful applications in different fields have become common phenomena but at the same time, we have not advanced in the field of sound classification although many approaches have been taken so far. Moreover collecting datasets is one of the critical stages of audio or sound classification. When small dataset may create acute ups and bounce in the final accuracy of the developed model. So we wanted to develop such a model that will perform better even working with a small dataset. Sound classification will provide future advancement in the domain of science and technology a mile.

Different bots can carry out different orders from human but it’s not always smart for the person to care about the bot itself. So if a system can be developed which will help the bots to know different signals coming from a different source of audio and it will take the corresponding steps. For example, if a fire alarm is rung then all the bots present in that area or building will recognize and come forward to stop the fire where there may be used different types of sound to classify the various intensity of the fire. Based on that sound it the bot may take decisions about what they should do instantly. No human is required to give them direction manually.

That’s how we got motivated and wanted to work in this field. It’s high time for humans to take steps so that the bot can make important decisions even without human interference. Audio classification is the primary intention here since if we can classify audios accurately then it will help us to use that model for a certain bot. Music instrumental audio classification can be used for the same purpose. Again we must not forget it’s audio whether it’s coming from music instrumental sound or not. For clarity, it can be said that the fire alarm bot will recognize the audio but the source of the audio may generate musical sound too. That’s why music instrumental audio classification is vital.

On the contrary, for human emotion recognition is can be recognized applying almost the same techniques of sound classification apart from this we got this idea when we were doing our hobby project ‘Emotion Detection using Image Processing’. We got about 66% accuracy and we were trying to come with a good idea for increasing the accuracy. We tried increasing features and increasing the quality of the dataset, but we couldn’t improve it to a great extent. Hence we come up with the idea that if we integrate the speech features and image features can we have better accuracy for the model. We searched over the internet and we found that this area is currently under research and not so much work is done in this area. We also did watch Google’s Duplex call video and were so impressed by that it inspired us to do this project.

1.3 Rationale of the Study

Sound classification is very useful at present whether it is music instrumental or any type of audio, classification can give it another dimension. Bots nowadays bots can carry out human orders swiftly but the hearing sound they should gain the power of distinguishing it’s the source and certain sound can be used to apply for various orders. For example, the bot will give space when it will hear the sound of the horn. Instead of looking or taking photos from the backside, the bot will give space to others. Bots have been using for various purposes but undoubtedly applying these technologies will give them enormous strength to solve problems in many criteria. So, this research is expected to contribute significantly to overcoming this crisis.

Similarly, the machine should gain the power of distinguishing male or female voice and their emotions too since it's easy for human but the purpose of machine learning or deep learning is to make the work and behave like humans, the more correct the better.

1.4 Research Questions

Throughout this research, we have attempted to address the following questions concerning the process of music instrumental audio classification. The explanation of these questions will be elaborately discussed in our research.

- How to classify audios?
- How to apply CNN for audio classification?
- How to apply RNN for audio classification?
- Does CNN work better than RNN to classify music instrumental audios?
- How can we detect the emotions of human hearing their voice?
- Do we need both the male and female voices to detect human emotions?
- Which algorithm will be applied for emotion detection?

1.5 Expected Output

The purpose of the study is to classify different music instrumental sounds accurately regardless of the size of the dataset. For the human emotion identification from speech, we built another model applying almost the same techniques as audio classification.

However, the following specific outcomes have been expected to achieve the main goal of this paper:

1. Finding out the feature extraction techniques.
2. Finding out the process of applying CNN and RNN to classify audios.
3. Finding out the process of modifying the dataset.
4. Knowing which one is better to classify audios (CNN or RNN).
5. Finding out how to apply audio classification techniques for human emotion detection based on their speech.
6. Knowing the effect of male and female voice when building a model for better performance to detect the emotions.
7. Finding out the possible accuracy for the used dataset to make the project done.

8. Application of the developed model

1.6 Report Layout

We divided the whole report into 6 chapters to make it easier for the reader. We started with chapter 1, where various introduction is given for both of the instrumental audio classification and human emotion detection. Chapter 2 is for the background of the studies. We also subcategorized that segment into various parts for a better presentation. Research gaps were discussed here too. In chapter 3 we had discussed the research methodology where we also demonstrated how the dataset was collected and analyzed. The experimental result was introduced in Chapter 4. Chapter 5 is very important since it describes how we implemented our research. Then at last but not in the least, chapter 6 is for the summary, Recommendations, and conclusion of the whole project. It should be mentioned that all of these chapters were subcategorized into various sections in accordance with necessity and References as well as Appendices have been included at the end part of this report.

CHAPTER 2

Background

2.1 Introduction

Traditionally, audio classification tasks have mainly focused on speech due to its wide applicability. Recent works have explored environmental scene classification using acoustic features. The availability of different datasets like UrbanSound, ESC50, and AUDIOSET has further aided the process. Previous works have mostly focused on the classification of independently occurring acoustic events. In this work, we explore the classification of instrumental sounds of 10 categories in audio recordings. We prepared a new dataset which is a subset of another dataset (FSD Kaggle 2018 (Freesound Dataset Kaggle 2018)”). We presented a convolutional neural network model and a Recurrent neural network model for the classification of music instrumental sounds. Different feature extraction techniques were used to improve the classification results.

For emotion detection, we used Convolutional Neural Networks and LSTM to classify opposing emotions. We separated the speech by speaker gender to investigate the relationship between gender and emotional content of speech.

There are enormous types of spectral features and temporal features that can be used to extract human speech features. We used a statistic relating to the pitch, Formants of speech as inputs to classification algorithms, and Mel Frequency Cepstral Coefficients (MFCCs). The accuracy getting from the emotions of these experiments allows us to easily explain which types of features carry very emotional information and it also explains its reasons.

This is how it allows us to develop criteria for the class's emotions together. By using these tactics we can easily achieve a very high emotion recognition accuracy for further studies.

2.2 Related Works

Different methods have been proposed by many researchers to classify musical instruments using various tactics. But from the literature review, it is very evident that this field still requires enough research to get the best output with better accuracy especially when the dataset is small it becomes more difficult. So approaches have been described below.

A paper on different techniques of signal processing was presented by Daniel P. W. Ellis, Meinard Müller, Anssi Klapuriand Gaël Richardet.al [11]. This paper focuses on the processing of the musical signal by summarizing numerous research fields. The application of MFCCs in the musical instrument classification is also viewed.

The application of MFCC along with Timbral associated Descriptors of audio to identify musical instruments was proposed by Priyanka S. and Jadhav et.al[12]. In the paper k- nearest neighbor was used for feature extraction along with SVM (Support Vector Machine) and binary tree. They also compared the classification method and feature extraction of numerous combinations for identification accuracies.

Another paper mentioned earlier in the introduction part written by D. G. Bhalke, C. B. Rama Rao, and D. S. Bormane et.al. presented FFT (Fractional Fourier Transform) to use MFCCs in musical instruments classification. They also used features of temporal related such as attack time, zero crossing rates, decay time, and energy. The calculation of the zero-crossing rate is given in equation (1),

$$ZCR = \frac{1}{T} \sum_{0}^{T-1} |\operatorname{sgn}[x(y)] - \operatorname{sgn}[x(y-1)]| \qquad (1)$$

where x(y) is the signal of the yth sample, x(y-1) is the signal of (y-1) sample and T stands for sample in per frame.

They also calculated the sound sample energy by the equation (2),

$$Energy = \sum_{n=0}^{T-1} (|\,m(n)\,|)\ 2 \qquad (2)$$

Where m(n) is the signal at the nth sample, and T denotes the samples in 1 (one) frame.

Sound samples feature was proposed by Slim Essid, Gael Richard, and Bertrand David et.al. [13] for MFCCs. They showed how to use the Delta MFCC features which were gained by time as derivative of MFCCs. The SVM algorithm is also used for classification problems. Spectral width, spectral centroid, etc. are used as spectral features. To train data, one mapping versus SVM one is used. N. Ozkurt and F. A. Savaci et.al [14] used wavelet ridges to classify instruments where the first three levels of the wavelet decomposition are functioned as showed in fig.2.2.1. The wavelet decomposition provides three detailed coefficients and one approximate coefficient.

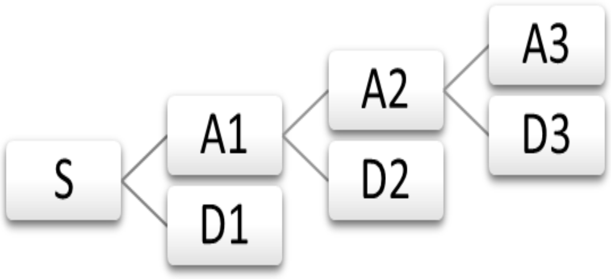


Fig.2.2.1: 3 levels of wavelet decomposition

Here, S= A3+D3+D2+D1

Where, S = signal frame, A = approximate coefficient, and finally, D = detailed coefficient.

Time scale feature related musical instrument classification was proposed by Farbod and Karthikeyan et.al [15] which was wavelet dependent. In the signal frame of the wavelet transform was taken continuously then features related to bandwidth and temporal variation were used to extract features.

On the other hand, for emotion detection, we started off with finding the already done researches and successful projects and papers over the internet and on the IEEE official website. We did find out the surveys made by Maisy Wieman, Andy Sun. Their research paper specifies the correct and on-point information about the vocal pattern analysis to detect the emotions. We also tried to find out the algorithms and methods used to determine the features from the vocal pattern. We were able to find out about the algorithm called MFCC (, Mel Frequency Cepstral Coefficients) [29].

2.3 Research Summary

Our prime concern was to find out the best accurate model which will be helpful for the bots to classify various audios regardless of the impact of the size of the dataset. So what we did, has been mentioned below,

1. Discussed profoundly various techniques of feature extraction available nowadays and we also discussed how to apply those techniques.
2. We showed how we preprocessed our dataset and made it ready for classification.
3. We made CNN model where images generated from different features extraction techniques were applied.
4. We also made RNN model for the same purpose.
5. We clearly distinguished the application of CNN and RNN for instrumental sound classification, where we got that CNN works better than RNN.

2.4 Scope of the Problem

A robust system needs to have the ability to classify various audios with better accuracy. But often the process gets hampered due to having lackings enough data and issues rise like the death spots of the audio signals. So building a model that will classify instrumental sounds more accurately was our prime concern. The hypothesis was that Recurrent Neural Network will work better than Convolutional Neural Network to classify instrumental sounds.

For emotion detection, to determine the emotions from the speech so as to give the machine a better approach to have a good conversation with humans. Moreover, machines should gain the ability to detect the emotion of humans since there may time comes, when humans may give oral orders to machines being emotional, so if the machine can understand it properly, it may be very helpful for the human race.

That’s why instrumental audio classification as well as human emotion detection from the speech is our proposed target. Throughout the project, we shall describe in further detail the whole procedures of our mission.

2.5 Challenges

We faced several challenges in conducting the study. In several stages, we had to change our decisions to cope with those challenges.

1. Identifying the right research topic.
2. Collecting dataset
3. Pre-processing the dataset
4. Removing death-spots from different recorded audios
5. Model building in CNN
6. Model building in RNN
7. Extracting various features for emotion detection.
8. Training the model again and again for better accuracy
9. Implementing our planned model

CHAPTER 3
Research Methodology

3.1 Introduction

The methodological portion of this research enables the reader to assess the general validity and reliability of research critically. The primary purpose of the current study is to show the procedure of extracting features from sounds. We used the MFC feature extraction technique which is popularly used nowadays for this purpose. The main point to understand about speech is that the sounds generated by a human are filtered by the shape of the vocal tract including the tongue, teeth, etc. This shape determines what sound comes out. If we can determine the shape accurately, this should give us an accurate representation of the phoneme being produced. The shape of the vocal tract manifests itself in the envelope of the short-time power spectrum, and the job of MFCCs is to accurately represent this envelope. The first step to classify instrumental music is to extract different features from audios i.e. identify numerous components of the audio signal while discarding all other kinds of stuff which might be the noise in the background or the dead space in the audio.

Nowadays Mel Frequency Cepstral Coefficients (MFCCs) are widely used to extract features in different audio recognition. MFCCs were introduced by Davis and Mermelstein in the 1980s, and have been state-of-the-art since MFCCs provides the efficient (effective) classification accuracy in the clean environment [16]. Moreover, the MFCC feature extraction process has become very common too in recent time, and the process is given in the fig.3 [17],

3.1.1 Steps at a Glance

We will give a high-level intro to the implementation steps, then go in-depth on why we did the things following the steps. Towards the end, we will go into a more detailed description of how to calculate MFCCs.

1. We have to frame all the signals into corresponding short frames.
2. From the short frame calculate the periodogram to estimate the power spectrum.
3. On the power spectrum the mel filterbank will be applied.

4. From the filterbank energies take it's logarithm value.

5. From all the log filterbank energies, take it's DCT.

6. Discard all other coefficient except the DCT.

There are a few more things commonly done, sometimes the frame energy is appended to each feature vector. Delta and Delta-Delta features are usually also appended. Filtering is also commonly applied to the final features. Fig.3.1.1.1 explains all of our proposed steps.

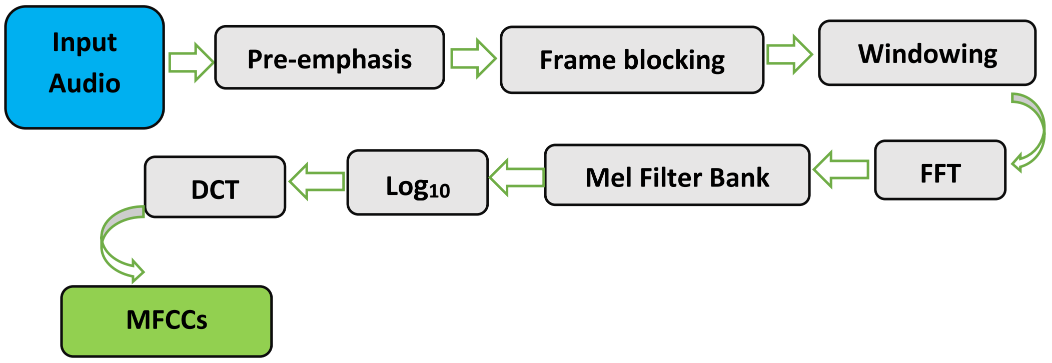


Fig.3.1.1.1: MFCCs block diagram

For implementation, MFCCs are less complex than most other feature extraction techniques and it is more robust as well as effective for various conditions [18]. MFCCs are designed on the basis of the human auditory system.

3.1.1.1 Pre-emphasis

Firstly, the audio signal gets passed through a filter so that the higher frequency can be emphasized. So the energy level gets increased for higher frequency signals. In this phase, the spectral flattening is performed using FIR (First order Infinite Impulse Response) filter [19], [20].

Equation (3) is representing the first-order filter (FIR),

$$H(z) = 1 - az^{-1}, \quad 0.9 \leq a \leq 1.0 \qquad (3)$$

3.1.1.2 Frame blocking:

The signal properties may be changed when it becomes too long and on the contrary, if the frame becomes too short then narrow-band components resolution will be sacrificed. That's why there is always a trade-off between them. The audio signal is segmented into a small segment of 20-30 ms which is called frame. The signal is generally divided into N samples where all the adjacent frames are separated by M (M<N). Normally the values of M = 100 and N = 256. Framing is required for any time-varying signal and its properties are stationary. That's how the spectral analysis of a short time is done.

3.1.1.3 Windowing:

1. The very next step of slicing the signal into frames, a window function is applied such as the Hamming window to each frame. The form of the Hamming window is given below,

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right) \qquad (4)$$

Here, $0 \leq n \leq N-1$, N = window length, Now before plotting the equation (4), it should be mentioned that among the signal frames (20-40 ms), 25ms is standard. So for a signal of 16 kHz, Frame length = 0.025 x 16000 = 400 samples. Frame step is normally 10ms (makes 160 samples), this is how the overlapping is allowed to the frame. So the first frame (for 400 samples) starts from sample 0, then the next 400 samples get started from sample 160, etc. until it reaches to the end. However, if the audio file can't be divided into an even number of the frames then it's padded with zero so that it does. Different types of windows and their comparison is given by Fredric J. Harris [21]. The plotting of the equation (4) is given below the fig.3.1.1.3.1 (200 samples),

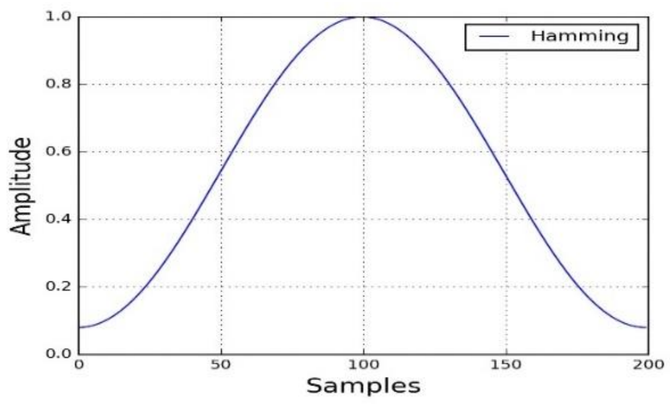


Fig 3.1.1.3.1: Hamming window

3.1.1.4 FFT (Fast Fourier Transform)

FFT converts time domain into the frequency domain. We perform FFT to get magnitude frequency from each frame. So the output of FFT is Periodogram or Spectrum [22],

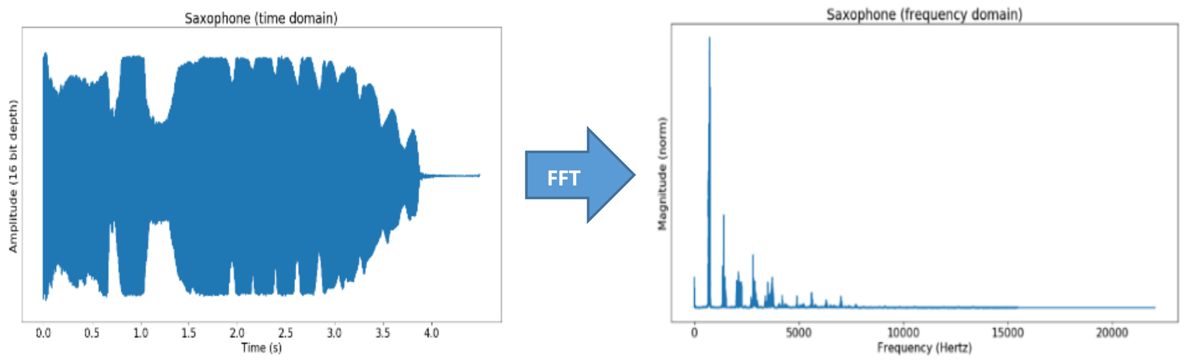


Fig. 3.1.1.4.1: FFT transformation

Fig.3.1.1.4.1 represents the FFT operation for Saxophone, it converted frequency into the magnitude-based domain.

3.1.1.5 Triangular bandpass filter

To get a smooth magnitude of the spectrum the magnitude frequency is multiplied by a set of 20 triangular bandpass filters which reduces the feature size and here Hertz = f.

$$\mathrm{Mel(f)} = 2595 \log_{10}\left(1 + \frac{f}{10}\right) \qquad (5)$$

Here equation (5) is used to convert the linear scale frequency into Mel scale frequency.

Triangular bandpass filters are used for extracting spectral envelop [23]. This is how Mel frequency filters are the triangular bandpass filters based non-uniformly on the Mel frequency axis, where there are more filters in low-frequency axis and fewer filters in number in the high-frequency region [24][25] which is evident in the fig.3.1.1.5.1 (for 26 filters) below.

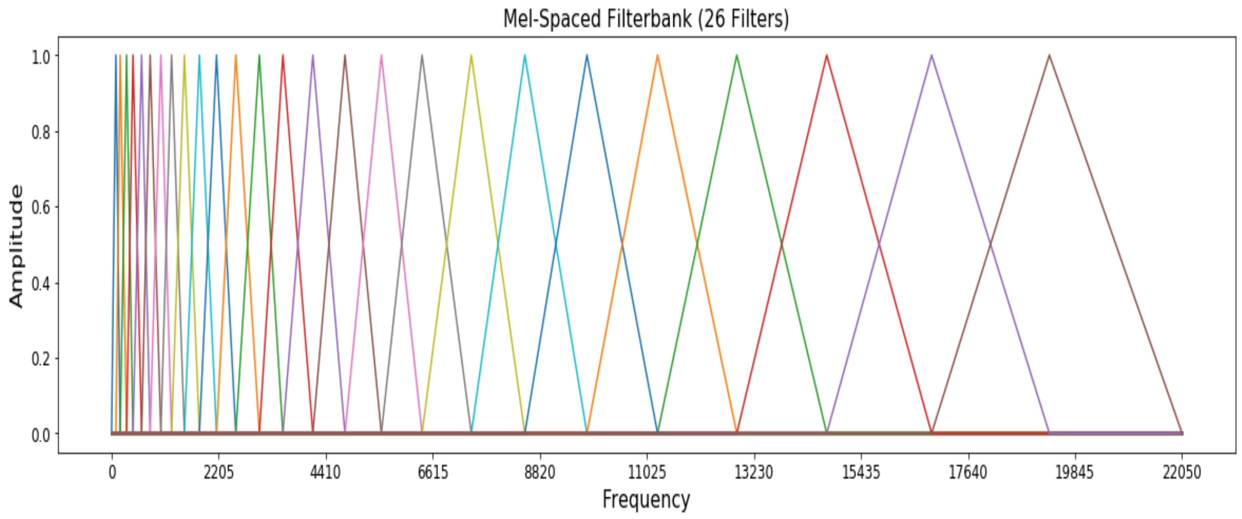


Fig 3.1.1.5.1: Filter bank

Equation (6) is used to produce a filter bank as visible in the fig.7

$$Hm(k) = \begin{cases} 0 & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \leq k \leq f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & f(m) \leq k \leq f(m+1) \\ 0 & k > f(m+1) \end{cases} \quad (6)$$

Here, M = number of filters (26 for the fig.6), f () = list of M+2, Mel spaced frequencies. The plot of 26 filters gets overlayed on each other. The first filter bank starts at 1st point, reaches its peak at the 2nd point then at the 3rd point return back to zero. The 2nd filter bank will start from 2nd point, reaching the peak at 3rd then back to zero at 4th, etc.

3.1.1.6 The logarithm of filters energy:

To calculate the logarithm of the sum of the filtered components of each filter (log-energy), equation (7) is used,

$$S(m) = \log_{10}\left[\sum_{K=0}^{N-1} |X(k)|^2 \,.Hm(k)\right] \qquad 0 \leq m \leq M \qquad (7)$$

Thus, every bin per frame of filter holds log energy gained by calculating the logarithm of the weighted sum of the spectral magnitude in the channel of the filter bank.

3.1.1.7 DCT (Discrete Cosine Transform):

DCT converts the Mel frequency domain (log power spectrum) into the time domain [26]. So in this final step, the output is Mel Cepstral Coefficients.

3.1.2 For Emotion Detection:

We used almost the same techniques of music instrumental audio classification in human emotion detection from the separate dataset. The prime difference was in the dataset. Apart from this, other things remained almost the same but just a few differences like the pitch of the sound for the human. Since while talking we use various pitches of sound and it greatly varies as soon as the gender gets changed. However, a better understandable model view is given below on fig **3.1.2.1**.

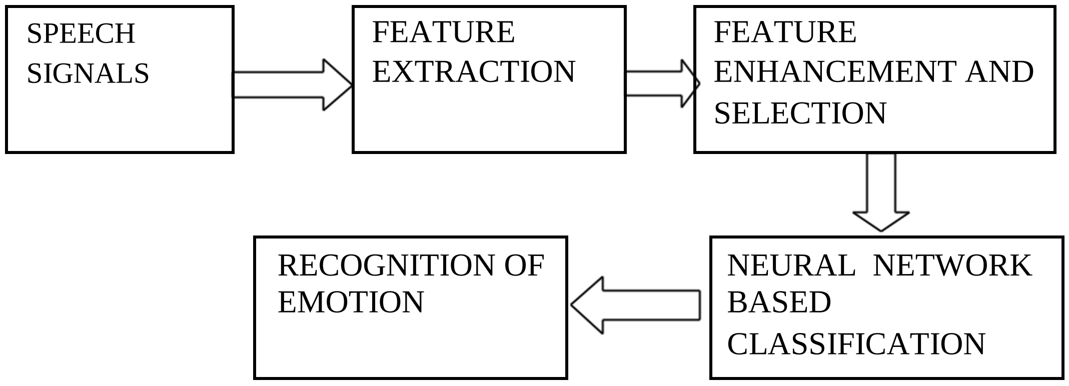

SPEECH SIGNALS
FEATURE EXTRACTION
FEATURE ENHANCEMENT AND SELECTION
NEURAL NETWORK BASED CLASSIFICATION
RECOGNITION OF EMOTION

Fig 3.1.2.1: Steps for detecting Emotions from Speech data

The following stages were implemented from left to right successively. At the very first stage, all the audio files were converted to a machine-readable audio format. Then for classification, we extract various techniques of extracting features. After that, we tried furthermore for better extraction of the features from human audios. So, when the data was ready to apply algorithm machine learning, we applied CNN on it. We applied CNN on the images of the audio files, which is known as a spectrogram. And this is how we finally got the desired result.

3.2 Research Subject and Instrumentation

3.2.1 Research Subject

We did research on audio classification where we worked with instrumental sounds. Literally audio classification becomes easier as soon as we can extract various features. So this research is all about feature extraction and then uses those features in our proposed CNN and RNN model. Finally, we predicted which model works better. After that, we applied the same tactics to classify human emotions which was an application of this research, and then at last we implemented the human emotion detection model on Flak.

3.2.2 Research Instrumentation

Our target was to achieve better accuracy even by training our model with a small dataset and our dataset is actually a subset of “FSD Kaggle 2018 (Freesound Dataset Kaggle 2018)”. That is a much bigger data set than what we'll be working with, it's about forty-two classes of audio and several gigabytes, the details about the dataset will be found in [27][28]. So our purpose is to focus on the implementation and then the model building.

3.2.2.1 Human Speech Datasets

We referred to two datasets RAVDESS and SAVEE Dataset for emotion detection. Only took the audio data from these datasets.

The dataset named RAVDESS is gender-balanced which is consisting of around 24 professional actors. The speech part of the dataset includes calm, happy, sad, angry, fearful, surprise, and disgust expressions, and the song part of the dataset contain calm, happy, sad, angry, and fearful emotions. Each and every single expression is produced by using two levels of emotional intensity, they also had an additional expression which was neutral. This is how, we got 2000 audio samples that were in the way format.

3.3 Data Collection Procedure

We've taken ten different classes that originally came from the Kaggle competition and we've only taken musical instruments, so we'll have ten different musical instruments and we'll try to classify each musical instrument using that dataset. We'll have 30 audio files to go along (in total 10x30=300 files). We'll also be working with the CSV file, so the CSV file is actually just to pair these weird obscure names of audio files with the actual classes of our musical instrument. The name of these 10 classes is 'Acoustic_guiter', 'Bass_drum', 'Cello', 'Clarinet', 'Double_bass', 'Flute', 'Hi-hat', 'Saxophone', 'Snare_drum', 'Violin_or_fiddle'.

3.3.1 Data Collection Procedure for human speech

The SAVEE dataset consists of recordings from 4 male actors in 7 different emotions, 480 British English utterances in total. 'DC', 'JE', 'JK', and 'KL' are the four male speakers recorded in the SAVEE database. There are 15 sentences for each of the 7 emotion categories, and one file for each sentence. It includes 'anger', 'disgust', 'fear', 'happiness', 'neutral', 'sadness' and 'surprise'. We made a customized dataset by using these two datasets. We have fewer samples of 'calm' emotion.

Hence we eliminate 'clam' samples to balance the dataset. Our dataset contains 7 folders, each represents the different emotions. Contain Separate emotion's voice/speech in each separate folder.

3.4 Statistical Analysis

We divided this statistical analysis into these following sub-categories:

3.4.1 Dataset Pre-processing & Cleaning

Before starting it should be mentioned that the whole model along with the pre-processing steps was run in a python (3.7) based environment. We also imported all the required libraries like 'Python speech features', 'Tqdm', 'Librosa', etc. The distribution of all classes from our audio dataset is visible in the fig 3.4.1.1, but the problem is there is a lot of dead space present in the audio files. We must remove these spots to get better performance.

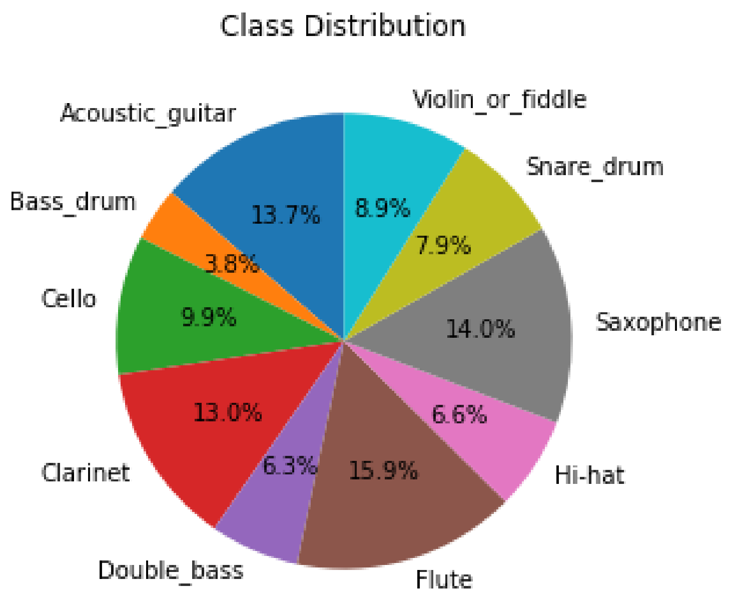


Fig 3.4.1.1: Before cleaning

Cleaning means removing the dead spots, typically which are the silent parts in the audio files. In fig.3.4.1.2 it is evident that the class distribution of our dataset has been changed after using the clean data set that was stored in a directory after removing the dead spots.

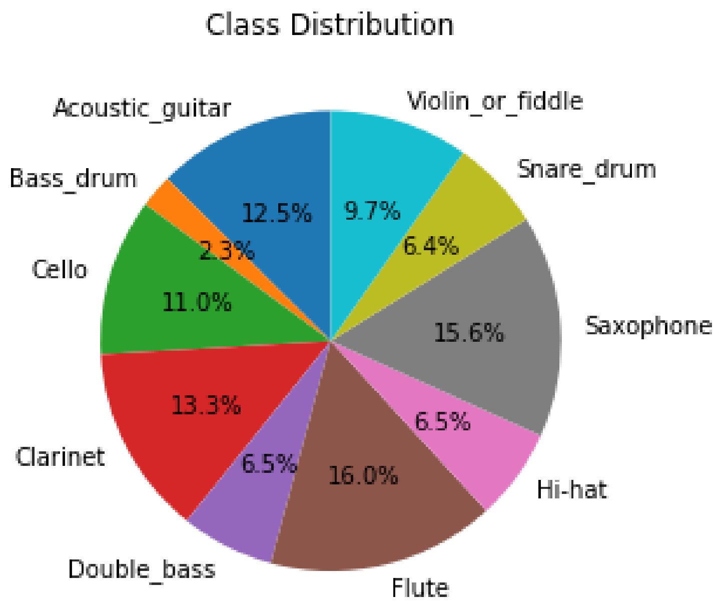


Fig 3.4.1.2: After cleaning

3.4.2 Speech Data View

The histogram of the dataset is given below, the number of various classes is displayed using histogram:

Our dataset contains 7 folders, each represents the different emotions. Contain Separate emotion's voice/speech in each separate folder. Labels assigned for the data is given below:

Labels Assigned for emotions :

Table 3.4.2.1: Labels of the dataset

Number	**Labels**
0	anger
1	disgust
2	fear
3	happy
4	neutral
5	sad
6	surprise

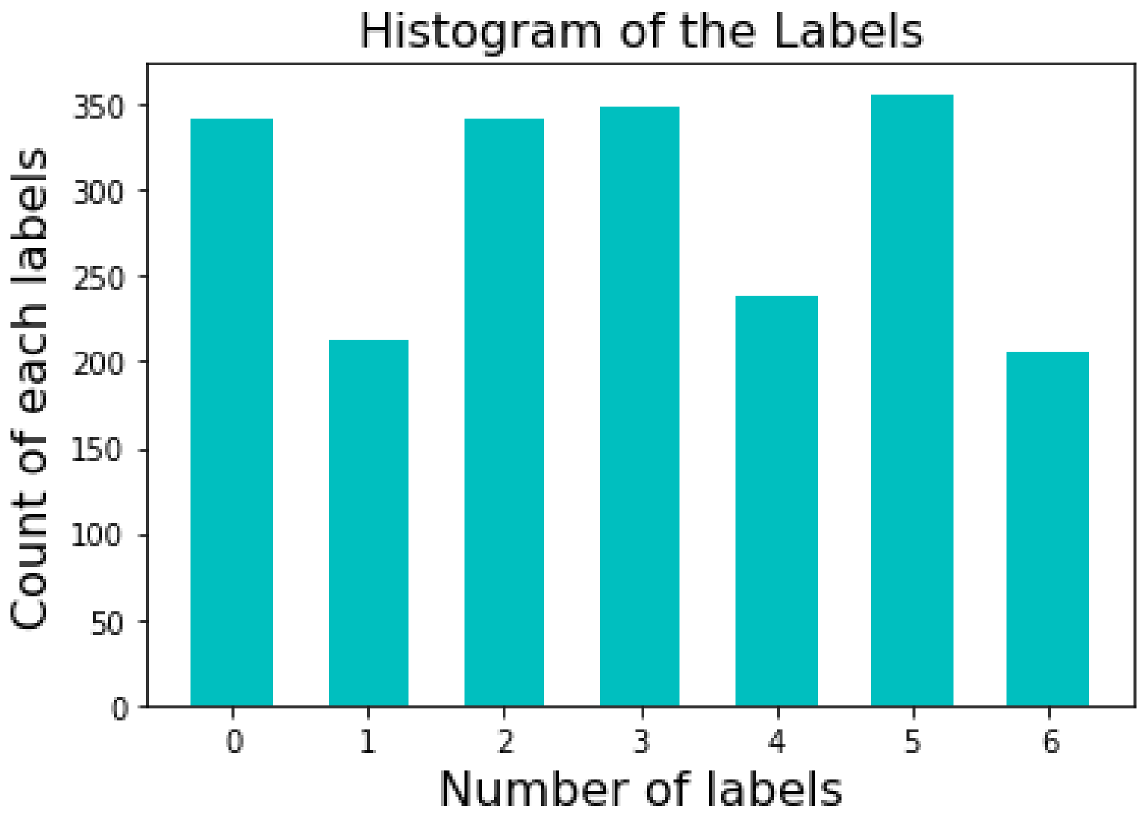


Fig 3.4.2.2: Histogram of speech dataset

Here is the number of these data:

Anger	: 436 samples
Disgust	: 252 samples
Fear	: 436 samples
Happy	: 436 samples
Neutral	: 308 samples
Sad	: 436 samples
Surprise	: 252 samples

Total : 2556 Samples

3.4.2.1 Plotting and Cleaning:

First of all, we created a directory named "Clean", so that we can keep the clean audio files in that directory, along with this we initialized 4 dictionaries, they are signals, FFT, Filter bank, and MFCCs. Where we used the number of filters = 26, FFT = 512 at the signal rate of 16000, Windows size of 25ms for short term Fourier transformation, sampling frequency = 1103, and finally the number of Cepstral = 13 for MFCC. Then we plotted the figure for corresponding dictionaries where we found a lot of dead spots. So we cleaned those audio files by removing dead spots and saved them in the clean directory to use for training purposes.

To remove it we calculated the envelope of a signal where the length or size of a window was 10^{th} of a second, periods per minute =1. Then we did downsampling to our audio by 16000. Downsampling was used to make sure we didn't have stuff in the high-frequency range so that we could discard it and create more compact data via downsampling and then we created a mask using the envelope function and passed the signal at the rate of 0.005 and put a mask over our audio. so this is why we created the clean directory we took everything form wave files, leaned all those up based on this noise floor detection, and did our downsampling then stored all of it inside the clean directory.

The corresponding figures are given in fig. 3.4.2.1.1, fig. 3.4.2.1.2, fig. 3.4.2.1.3, and fig. 3.4.2.1.4

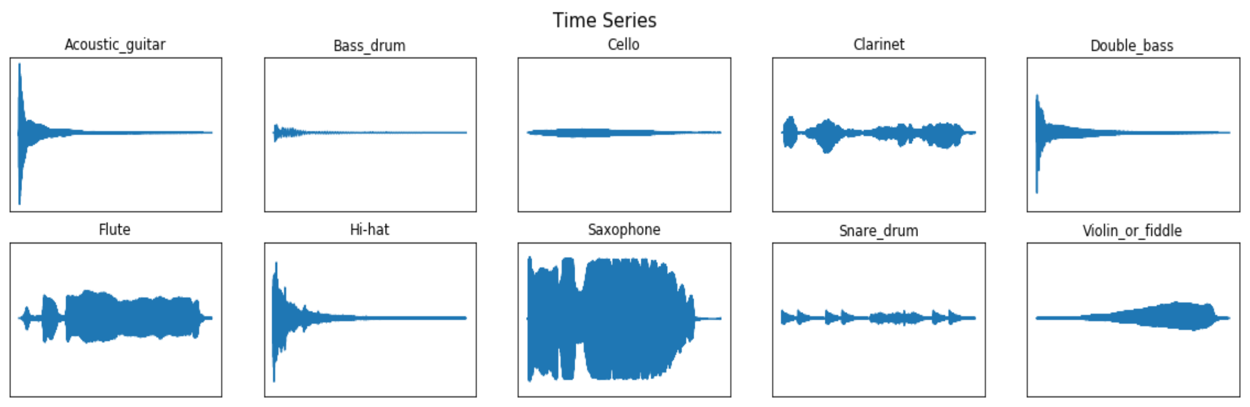


Fig 3.4.2.1.1: Time series plot of clean data

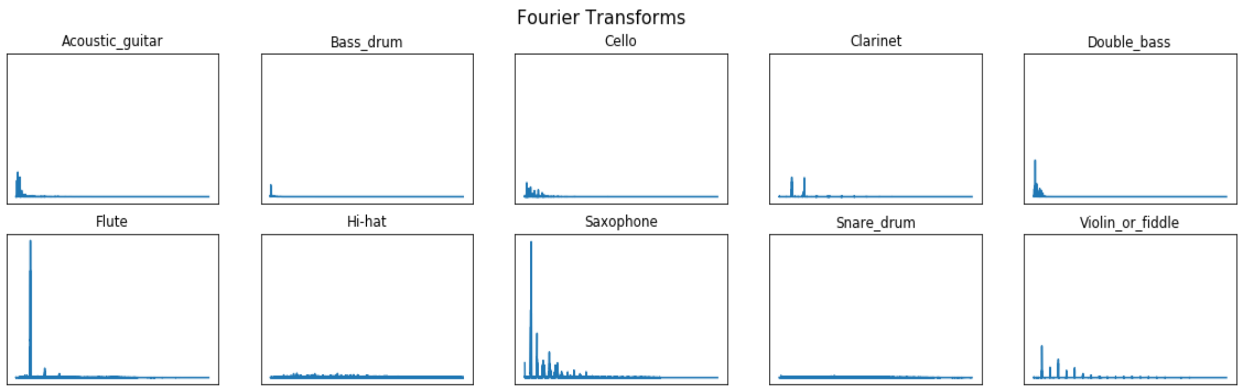


Fig. 3.4.2.1.2: FFT plot of clean data

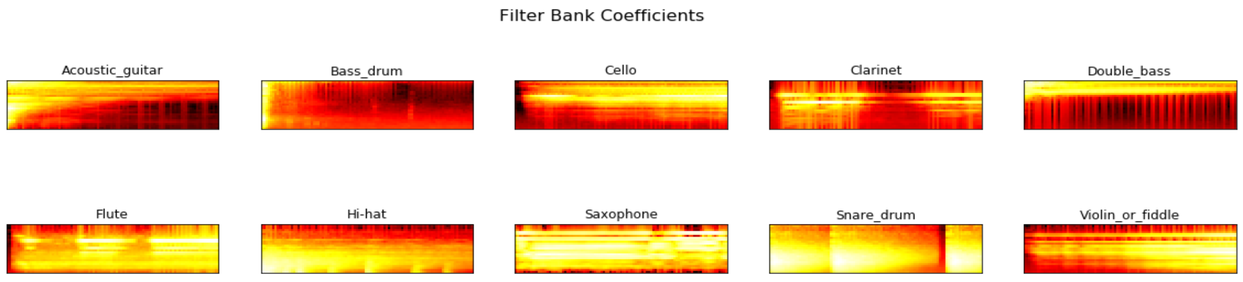


Fig. 3.4.2.1.3: Filter Bank plot of clean data

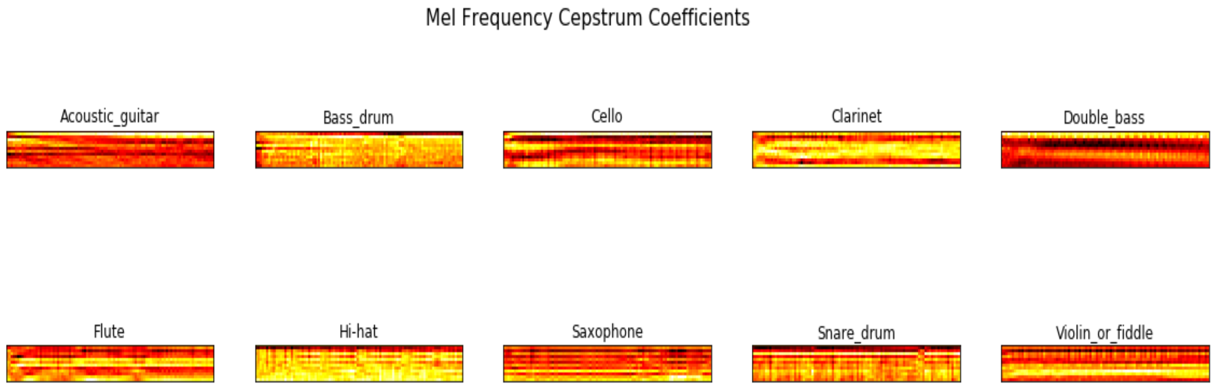


Fig. 3.4.2.1.4: MFCCs plot of clean data

3.4.2.1.1 Plotting the audio file's waveform and its spectrogram (Human speech):

We tested out one of the audio files to know its features by plotting its waveform and spectrogram. An example is given in fig. 3.4.2.1.1.1 and the Frequency Domain Plot of the Speech signal is given in Fig 3.4.2.1.1.2.

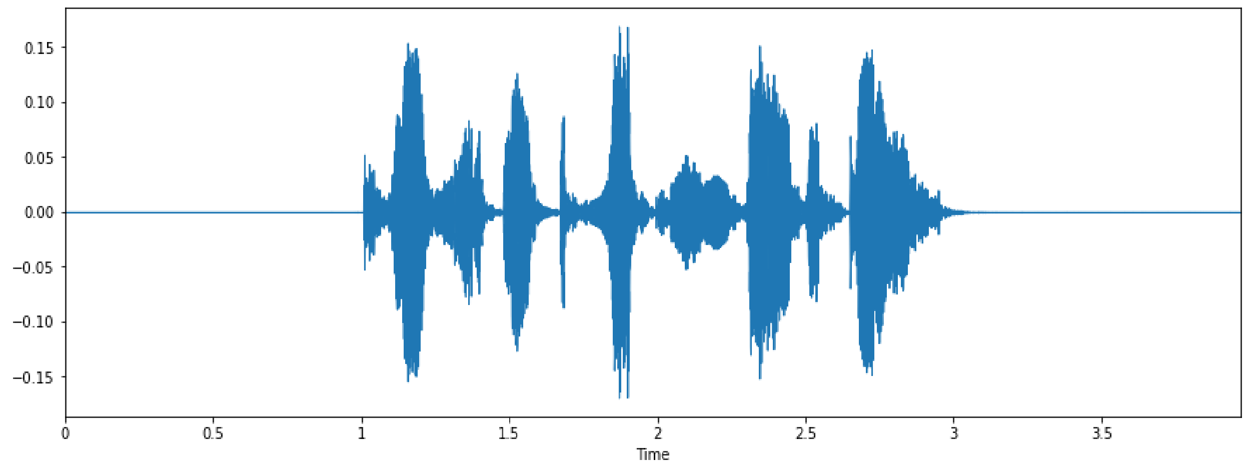


Fig 3.4.2.1.1.1: Time Domain Plot of the Speech signal

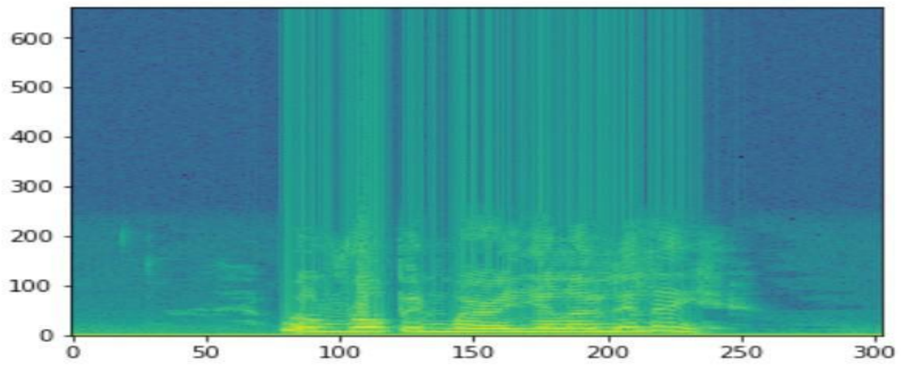


Fig 3.4.2.1.1.2: Frequency Domain Plot of the Speech signal

The next step involves organizing the audio files. Each audio file has a unique identifier that tells the emotion of the file which can be used to determine the label of the audio file. We have 7 different emotions in our dataset. We used the Librosa library in Python to process and extract features from the audio files. Librosa is a package of python for music or audio data analysis. It provides the facility of building blocks that are necessary to create musical information and for the retrieval systems.

Using the Libros library we were able to extract features i.e MFCC(Mel Frequency Cepstral Coefficient). In an automatic speech as well as speaker recognition the MFCCs are a feature widely used.

We also separated out the female's and male's voice by using the identifiers. This was because as an experiment we found out that separating male and female voices increased by 15%. It could be because of the pitch of the voice was affecting the results. Each audio file gave us many features which were basically an array of many values. These features were then appended by the labels which we created in the previous step.

3.4.2.2 Getting the features of audio files using Librosa

3.4.2.2.1 Calculating MFCC, Pitch, magnitude, Chroma features:

At this stage, we calculated MFCC features, Pitch, Magnitude, and the Chroma features of the human speech for both of the male and female voices. We did this in the same way as we did in the music instrumental sound classification described earlier. But for human speech Pitch plays a vital role since it can be different from person to person, unlike instrumental sounds. The pitch of a speech for the male is very much different than the pitch of a female for the exact same speech. So, it always can be confusing for the machine to recognize the emotions as soon as the gender gets changed. That’s why we included both of the male and female speeches in our dataset for various emotions like sad, happy, and etc.

We also used the Chroma features. Chroma feature is very well known nowadays to identify the difference among various pitches. It also can visualize a very high degree of robustness. That is why the Chroma feature is applied in musical or audio-based data. Examples of these successive stages have been given in Fig 3.4.2.2.1.1: Feature extraction.

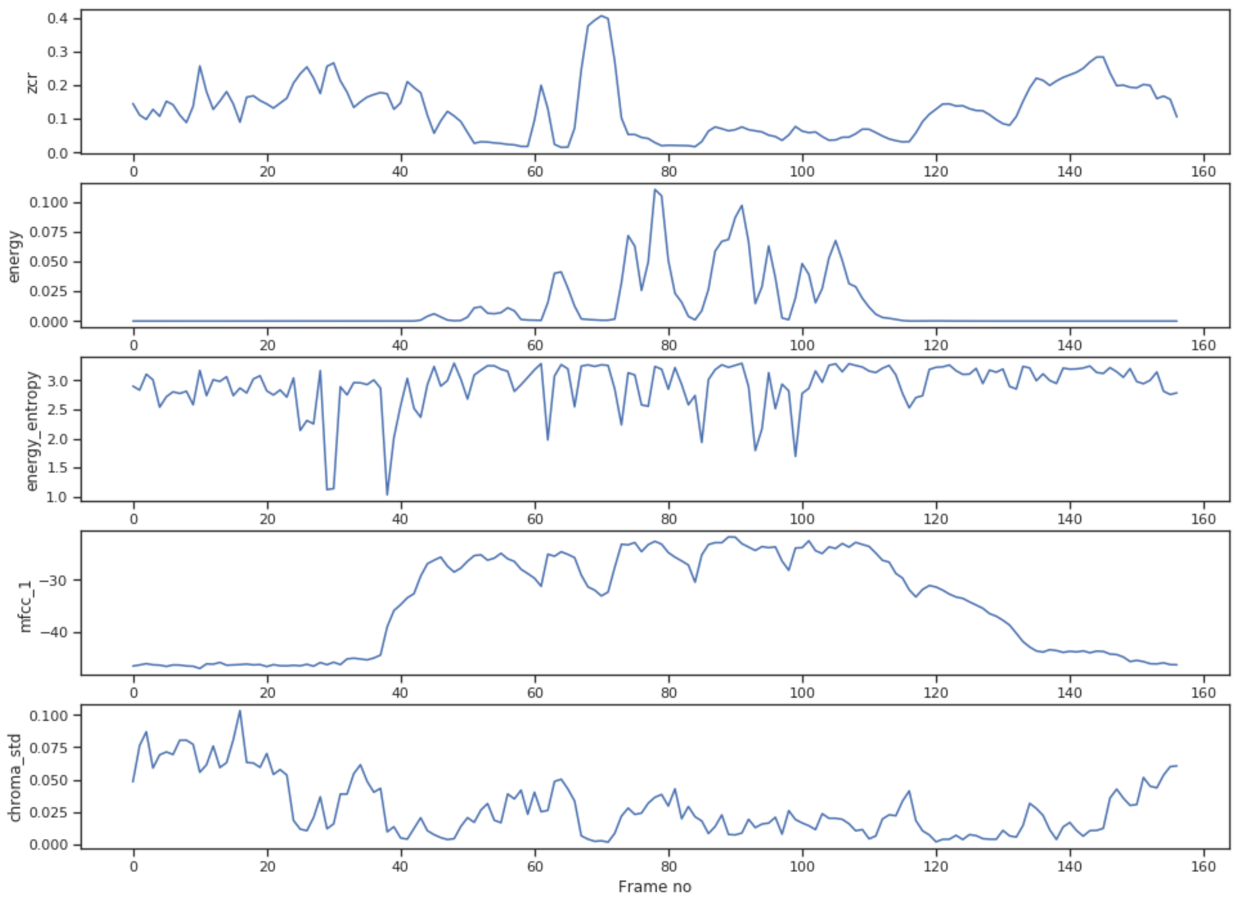


Fig 3.4.2.2.1.1: Feature extraction

3.4.2.2.2 Features Extraction details:

The representation of time-domain for sound can be very complex, and in for its original form, it can not provide a standard insight into the key characteristics for the signal. Due to having characteristics in sound signals of these types, we can easily map a better representation of time-domain into more types of their telling features. The very easy and pretty straightforward technique that gets involved by determining the average of the energy of the signals. The energy present in the signal and this metric which generally indicates various speaker of 'volume'. The duration offers insights into emotion, as a result following the estimated statistics like the mean, range, minimum, maximum, as well as the standard deviation for both of these spectrums and signals. This is how that may indicate the various fluctuations in the pitch or for the volume as well that can be very used to calculate emotion. For both of these signals and spectrums, we also did derive the skewness, and kurtosis, the measurement of the departure for horizontal symmetry, and the measure of sharpness as well as the height of the central peak in the signals that are very close to the standard curve and relevant to the type of bell curve.

We have also processed the signals in the domain of frequency through the FFT (Fast Fourier Transform). We also used the samples of windowed in order to get the accurate representations of the frequency on their contents of the signals for various points in time. For each window sample, the square value of the signals was taken. So that we could easily derive them in the expected power spectrum. We used the value of these power spectrums as their features, at the same time we also determined the frequencies that could have the maximum possible power. We had obtained three maximum peaks of the frequency for each and every single window then we added those with their feature vectors. That's how, we found the minimum and maximum frequencies with the substantial power for each and every single time frame and we used these value to determine the range of frequency for each frame. By mapping the spectrum of power, the auditory spectrum can very easily be derived to the axis of an auditory frequency from the combination of the FFT (Fast Fourier Transform) bins into almost equally spaced intervals.

The MFCC (Mel-frequency Cepstrum Coefficients) captures different characteristics from the frequency of the signals that are represented on the Mel scale, generally, that closely aligns with the nonlinear characteristics or nature of the humans hearing procedures. Elaborately, the MFCC (Mel-frequency Cepstrum Coefficients) represents the spectrum (spectrum of the spectrum). By mapping the powers of the frequency spectrum, MFCC's can be derived onto the Mel scale, and then by calculating the logs of these powers, followed by their DCT (Discrete Cosine Transform). So that's why MFCC's (Mel-frequency Cepstrum Coefficients) are widely used as features in various speech detection applications.

Based on both a fine time scale and a coarse time scale, the changes of pitch gets measured. The signal is divided into three (3) parts like the beginning, middle, and end, for the coarse measurement, and the maximum average pitch of the sound signal is used to detect whether the certain pitch rises or drops over time. The most dominant frequency for all the windowed samples to determine the fine measurement, are compared to the most dominant frequencies received from those windowed samples immediately following and preceding. At last, their difference is recorded in the feature vector of each time.

3.4.2.2.3 Features Selection:

In order to extract features, we had to process all the signals of the sounds, the algorithm that was used for high variance revealed that we were to filter a large number of features in order to detect it, which contributed mostly in the used or following classifier. We had approximately 72 windows for every audio sample, our input speech signals had been windowed and almost each of these generated windowed samples provided a total number of 577 features. In total, we had extracted 41,558 features. This huge number of features which is much larger than the number of examples that had resulted in such a variance which was very high. So, the very next duty for us was to extract the most useful and important features. Because of the large number of features, we used heuristics to score each feature, rather than the implementation of backward or forward brute-force search.

3.5 Implementation Requirements

- Language: Python (Version: 3.7.0)
- IDE : PyCharm / Open source web application: Jupyter Notebook
- Google Colab
- Library : Pandas (Data analysis)
- Library : Malplotlib (Data visualization)
- Library : Scipy (Signal Processing)
- Library : tqdm (For visualizing progress)
- Library : Keras (for adding layers)
- Library : Librosa (for audio analysis)
- Library : python_speech_features (to use MFCCs)
- Library : Sklearn (for classification)
- Fundamental package for computing: Numpy
- Microsoft Excel

- Microsoft Word
- Basic knowledge of computing
- Knowledge of Python

CHAPTER 4

Experimental Results and Discussion

4.1 Introduction

In this segment, we are going to describe the details of our developed model and their performances separately. Any audio classification is classified after extracting the features from these audio files but often collecting audio files take a lot of time again there may rise various issues like background noise and the pure length of the audios. We build a model in such a way so that the file works well for both of the CNN (Convolutional Neural Network) and RNN (Recurrent Neural Network).

4.2 Experimental Results

We divided this part into a few segments. The full experimental result will be analyzed gradually maintaining steps for the clarity of our work.

4.3 Descriptive Analysis:

4.3.1 Model preparation for audio classification:

Now we can read data from the clean directory, we down-sampled our audio then removed the dead spots in the audio as well. We also did the noise floor detection. Now at this stage, the bass drum went down by 1%, so we removed a little bit of data here and there. Model preparation is about how we might handle class and balance when we train a neural network. Here we created a function that generated the X matrix and Y matrix. We used only 1/10 of a second of audio for each sample. So, we randomly sampled along with the length of our audio files and then we took that one-second chunk out of that audio. Then we shaped our data and also our input matrix to predict our target variables Y.

Table 4.3.1.1: Parameters name and value

Properties Name	Value	Calculated value
Sampling Rate	16 kHz	-
Window Length	25 ms	400 samples

Step size	10 ms	160 samples
N FFT	512	-

4.3.2 CNN (Convolutional Neural Network) Model:

We need three basic components to define a basic convolutional network.

1. The convolutional layer

2. The Pooling layer[optional]

3. The output layer

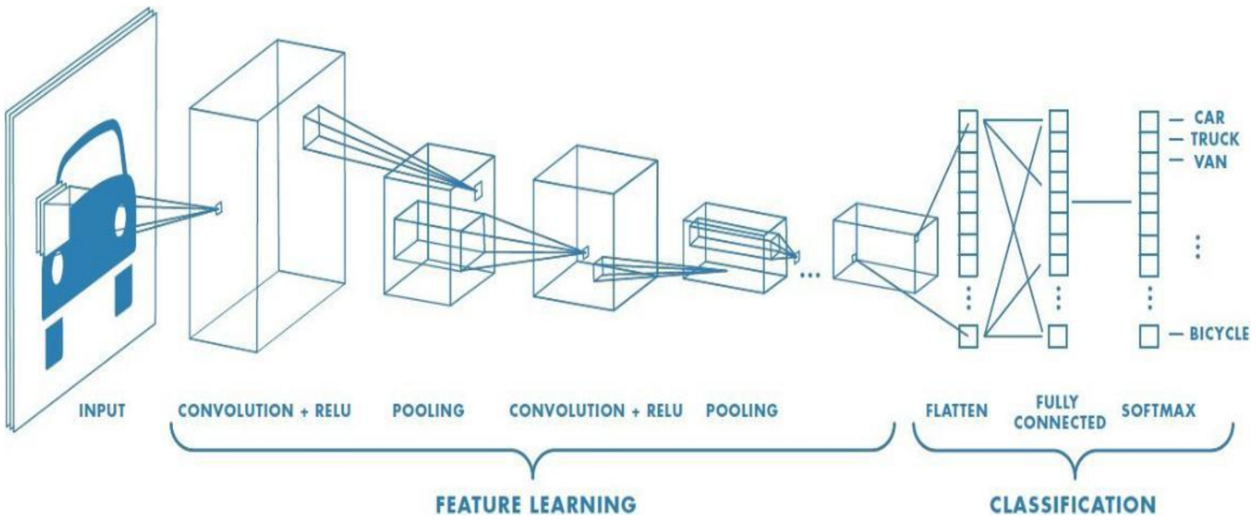


Fig 4.3.2.1: General Representation of CNN model

All the successive stages of a CNN model is given below:

- The input image/audio is passed to the first convolutional layer. The convoluted result is an output that is obtained as an activation map. After that, the filters are applied in the convolution layer to extract relevant features so that we can pass further from the input images.
- To detect the correct class during prediction, almost each and every filter will provide a different features. In case we need to retain the size of the image, we have to use the same

padding specifically zero paddings, otherwise, the valid padding is used since it helps to lessen the number of features.

- Pooling layers are also added for further reducing the number of parameters.
- Several pooling layers and convolution layers are added before the prediction is made. Convolutional layers help in extracting features largely. As we dig into the deeper of the network, the more and more specific features are extracted as compared to a shallow network where the features extracted are more generic.
- Generally, the output layer of CNN that was mentioned previously is a fully connected layer, the input taken from the other layers are flattened and sent to transform the output for the number of classes that are desired by the network.
- Next, the output is generated through the layers of output and it is compared to the output layers for error detection and generation. A loss function is also defined at that time in the fully connected output layer in order to compute loss of the mean square. The errors of the gradient are generally calculated then.
- After that, to update the bias values and the filter (weights), the error is backpropagated.
- In a single or per forward and backward pass, 1 (one) training cycle is completed.

4.3.3 CNN for Audio classification:

At this stage, we built our CNN model. We need to take the hot encoded Y matrix and then we needed to process them back into their original classes. To do so we used Numpy Argmax, hat mapped them back to the original column, then we set the input shape of our convolution, where we put, batch size = 32, epochs = 10, shuffle = true, class_weight = classes_class weight(provided by Scikit-learn), testing data = 10%, etc. We used the convolutional layer followed by the pooling layer, to stack those up over time and we would eventually pull down our data if our input space is really high then having a lot of pooling layers makes sense. However, the CNN architecture is given below,

Table 4.3.3.1: Sequential CNN Architecture

No.	Type	Size	Other

1	Convolution
	Relu
		16, 3x3
		-
			Strides= 1x1
			Padding = same

2	Convolution
	Relu
		32, 3x3
		-
			Strides= 1x1
			Padding = same

3	Convolution
	Relu
		64, 3x3
		-
			Strides= 1x1
			Padding = same

4	Convolution
	Relu
	Maxpooling
	Droupout
		128, 3x3
		-
		2x2
		0.5
			Strides= 1x1
			Padding = same

5	Fully connected Relu	128 -	- -
6	Fully connected Relu	64 -	- -
7	Fully connected Softmax	10 -	- -

4.3.4 RNN (Recurrent Neural Network) and LSTM:

LSTM has characteristics of RNN. It is called Long Short Term Memory networks –in the short form it is "LSTMs" – are a special type of RNN, having the capability to learn long-term dependencies. It was introduced by Hochreiter & Schmidhuber (1997), and were popularized and refined by many people mentioned here [30]. They worked quite well on huge types of problems and nowadays they have been widely used.

LSTMs have been designed to avoid the problem of long-term dependency. Practically their default behavior is remembering information for long periods of time is, that is not something they do struggle to learn!

Almost all the recurrent neural networks have the form like a chain for repeating modules of the specific neural network. In the standard RNNs, this repeating module will have a simple structure, like a tanh function of a single layer.

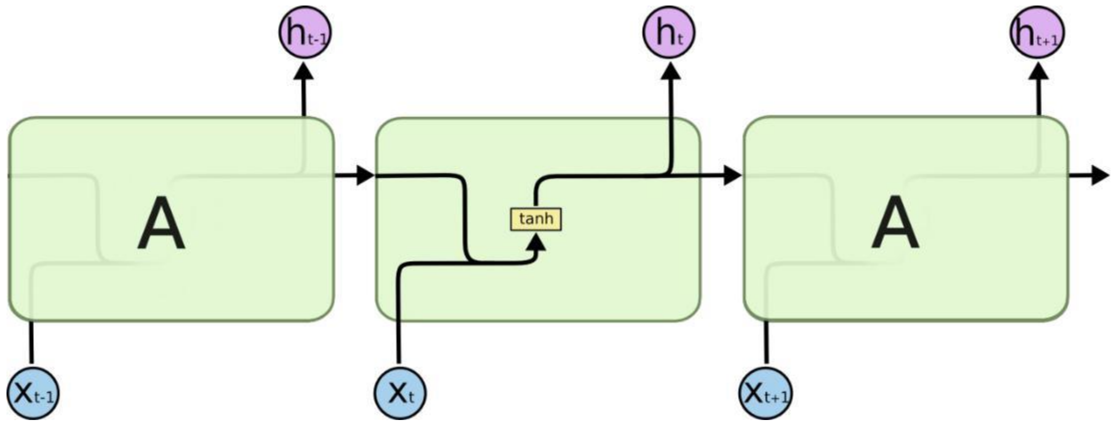


Fig 4.3.4.1 :The repeating module in a standard RNN contains a single layer.

The chain-like structures are also seen in LSTMs when the repeating module has a different structure. Instead of using a single (1) layer of the neural network, there are four, and they interact in a very special way.

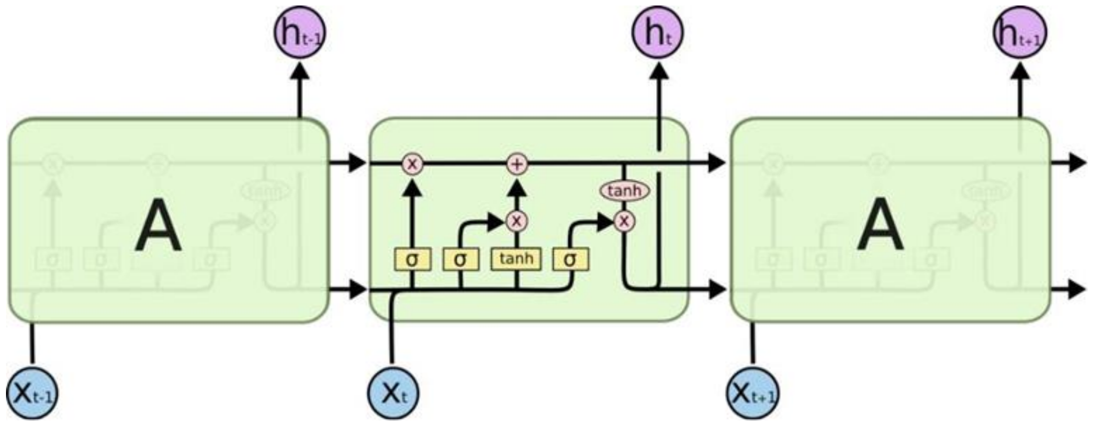


Fig 4.3.4.2: The repeating module in an LSTM contains four interacting layers.

The following figure Fig 4.3.3.2 is representing how does a LSTM network work. But for clear understanding, all the notations have been described below in detail.

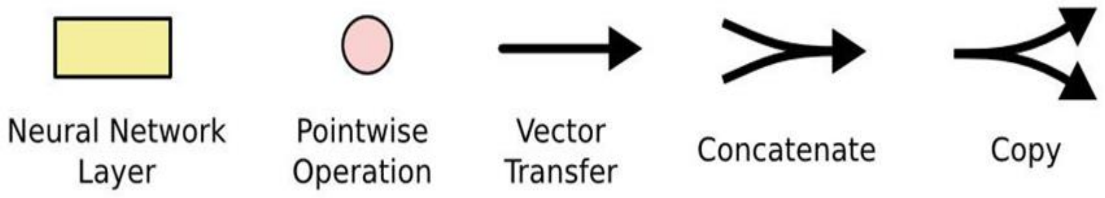


Fig 4.3.4.3: Descriptive Notations

The diagram here in Fig 4.3.4.3, generally each line carries almost an entire vector, from the most output to the very inputs of others. Here the pink color circles represent a pointwise operation of the process, for example, the vector addition, when the yellow boxes represent the learned layers of the neural network. Likewise, the lines merging denote a concatenation, and a line forking is used to denote its content being copied while the copies are going to different locations.

4.3.4.1 RNN (Recurrent Neural Network) model:

Next, we built an RNN model so that we can compare its performance with the CNN model. For recurrent model, we do not use convolution layers. Rather we use LSTM (Long Short Term Memory) unit. It’s kind of dense layer but it’s a long short term memory unit. The way it works like having a lot of neurons but these neurons have long short term memory components. However, we train our model for 10 epochs, when we took 10% data for testing purposes. More details description is given below,

Table 4.3.4.1.1: Sequential RNN Architecture

No.	Type	Size	Other
1	LSTM	128	Return sequences =

			True
2	LSTM	128	Return sequences = True
3	Time Distributed Relu	64 -	- -
4	Time Distributed Relu	32 -	- -
5	Time Distributed Relu	16 -	- -
6	Time Distributed Relu	8 -	- -
7	Fully connected Softmax	10 -	- -

Again we compiled and used the categorical cross-entropy for model's loss, Adam optimizer, and accuracy for matrix. We got fine accuracy here in RNN, but it really took a long time to train the model comparing with the CNN model. We got around 87.62% accuracy for the following RNN model.

4.4 CNN for Human Emotion Detection:

To make this operation we randomized the dataset and split it 80% for training and 20% for testing. Where the sampling rate was set to 20000.

Dataset samples: 2556
Training Samples: 2044
Testing Samples: 512

Primarily we tried several models and choose the best one. The very first one is given here below using the summary of the model.
Apart from this, during the compilation of the following model we set the following parameters as follows,

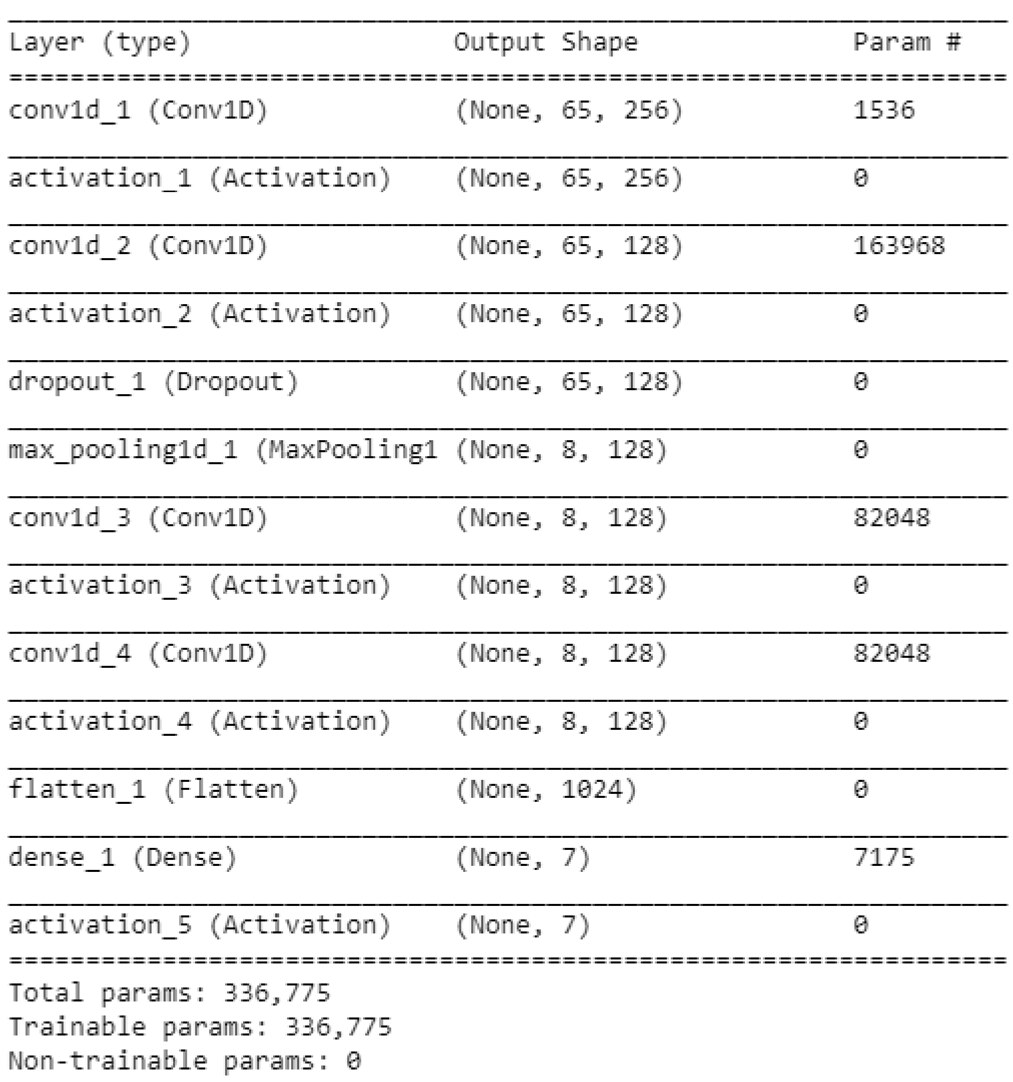
loss='categorical_crossentropy'

optimizer=opt,

metrics=['accuracy']

batch size=16

epochs =400



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_1 (Conv1D)            (None, 65, 256)           1536
_________________________________________________________________
activation_1 (Activation)    (None, 65, 256)           0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 65, 128)           163968
_________________________________________________________________
activation_2 (Activation)    (None, 65, 128)           0
_________________________________________________________________
dropout_1 (Dropout)          (None, 65, 128)           0
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 8, 128)            0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 8, 128)            82048
_________________________________________________________________
activation_3 (Activation)    (None, 8, 128)            0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 8, 128)            82048
_________________________________________________________________
activation_4 (Activation)    (None, 8, 128)            0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1024)              0
_________________________________________________________________
dense_1 (Dense)              (None, 7)                 7175
_________________________________________________________________
activation_5 (Activation)    (None, 7)                 0
=================================================================
Total params: 336,775
Trainable params: 336,775
Non-trainable params: 0
```

Fig 4.4.1: Model Summary of Emotion detection

So here total parameters were 336,775 and trainable parameters were the same as the total parameters. All the layers that we have added for the following model are also very visible in the Fig 4.4.1.

So now it’s time for training the model. We trained the model and got around 60% accuracy. Basically for any dataset which is complex like there are various types of data (both male and female) moreover the background of the dataset and the death spot may cause damages to the whole process. When the quality of the microphone or the recorder is another issue which may vary from device to device and bran to brand.

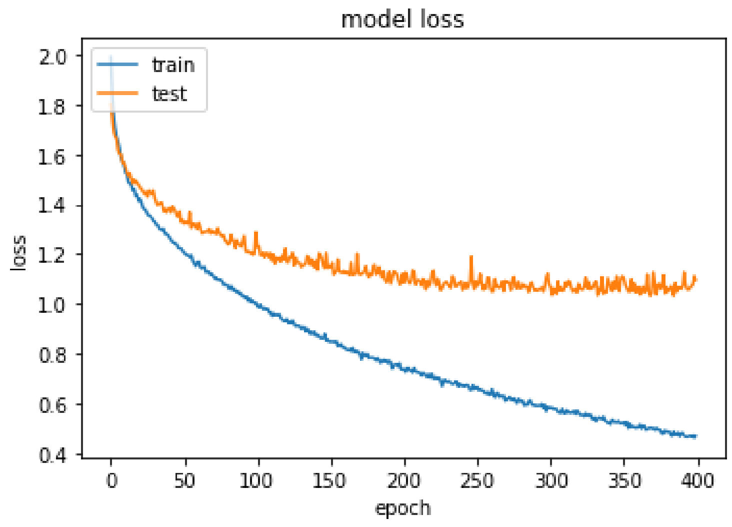


Fig 4.4.2: Loss VS Iterations

The accuracy for the following fig 4.4.2 was around 85% on the training dataset and 60% for new testing audios. The model is not seeming to perform well after 50 epochs but 50 epochs look too low for training audio datasets. On the other hand, the loss is high too.

That’s why we run several models to get rid of this situation.

4.4.1 CNN for Human Emotion Detection (another model):

Keeping everything almost the same we made changes in some parameters to reduce those problems. The following changes are given below here:

batch size=8

epochs =200

dropout rate = 0.3

Here we have increased the number of dropout rates since the last time we found that the loss was very high. The result for that model is visible in fig.4.4.1.1 where we can notice that the loss has decreased a lot. But the accuracy after the following changes were reduced by 2%.

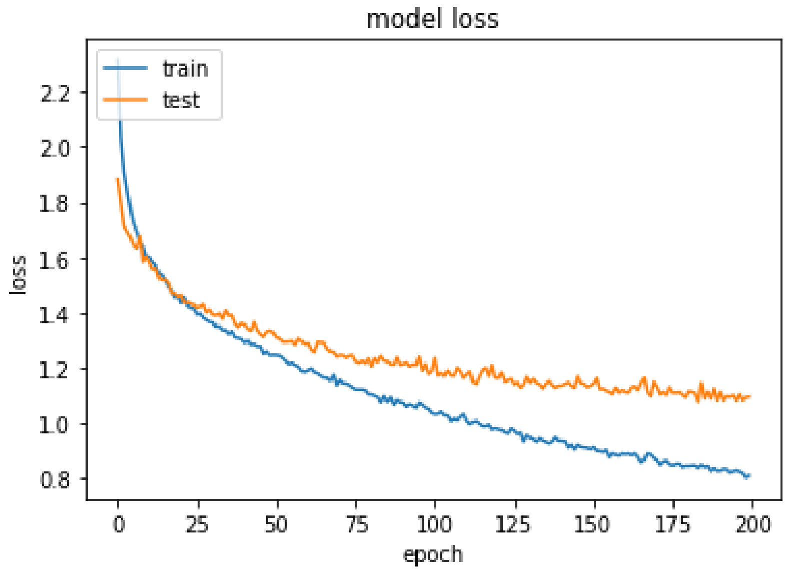


Fig 4.4.1.1 : Model VS Iterations(another model)

4.5 Summary

So far we have applied two different deep learning models and algorithms in order to classify audio related data. The algorithms were applied to the two separate datasets. Initially, we tried to realize which algorithms work better. So we build a model and then started the training for the music instrumental dataset. As soon as we finished training the model using a convolutional neural network the result was amazing. We know the convolutional neural network performs more accurately for image related datasets. As a matter of fact, we converted all of these music instrumental sounds to images using various tactics which is the sole purpose of this research. We applied the MFCCs algorithm to get the logarithm value since we could not determine the source or the types of audios just by seeing their time-domain series images. The FFT (Fast Fourier Transform) was not enough for the process. So the triangular bandpass filter helped us a lot and finally, the DCT (Discrete Cosine Transform) removed all the triangular filters of high frequency. That's how at the very last stage we got the images that could be used to feed into the convolutional neural network model. A simple periodogram can't help us to do so. After training the dataset with CNN we tried using RNN (Recurrent Neural Network). The recurrent neural network gave us a higher accuracy too but comparing with the convolutional network we came to a point that the CNN algorithms perform better than RNN. So after that, we were ready to train any dataset with CNN applying the same tactics. This time we tried another dataset to recognize the human voice. Human voice means the emotions of humans. We built another model using the algorithm of a convolutional neural network for the purpose. As we know human voice is full of various pitches, which gets changed as soon as the person gets changed. That's why probably a huge dataset could be helpful to get very accuracy but training a dataset that is more than several gigabytes takes a lot of time. So we trained the model using around 2500 samples of audio or speeches from several persons including both male and female. Now in the next segment, we shall implement our built model for human emotions detection.

CHAPTER 5
Implementation

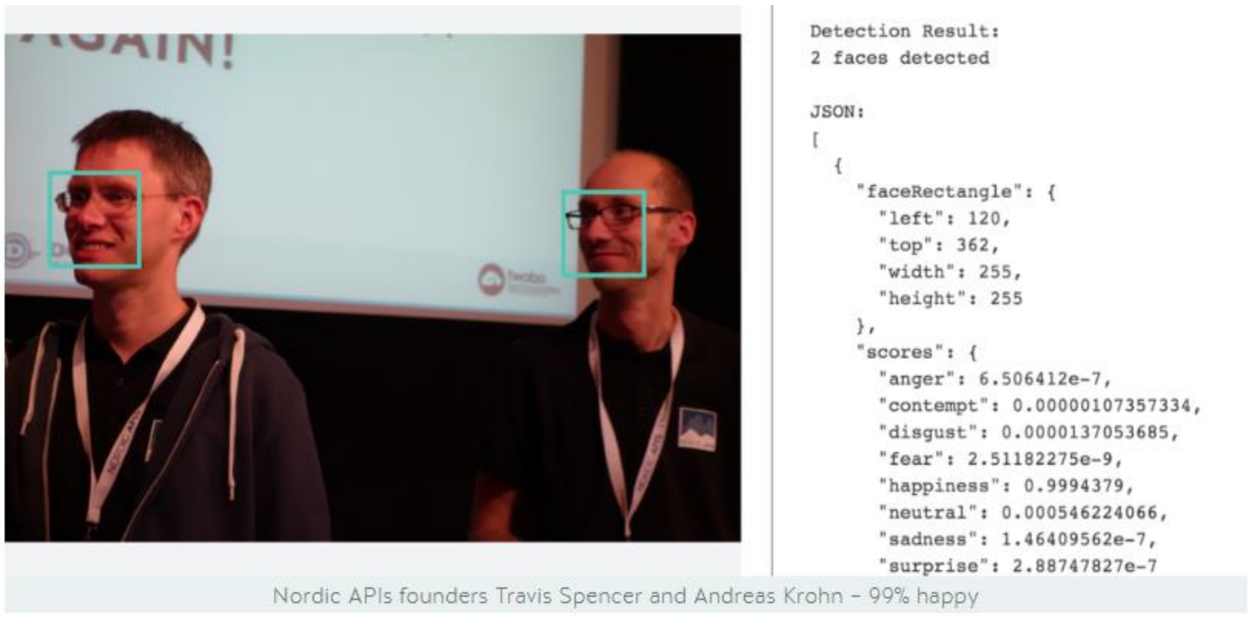
5.1 Introduction

Now we are living in an age where machines are being used for various purposes like machines can detect and identify images very easily using different algorithms. Even it's possible to track human face from live video. But among all of these possibilities, the fields of machine learning is pretty lagging behind from the present pace of modern technology. So we showed and described very clearly how we build models and chose the best one for audio classification. But we wanted to do something unique, we also wanted to give it a new dimension. We tried our best to develop a Desktop Application so that the user can record their voice and then try our model to detect how the model is performing. We used the Flask framework to implement our project. More details will be described in this chapter (Implementation). We developed this for human emotion detection instead of audio classification from music instrumental sounds since the source of audio may not present with the user. However, these scopes will be overcome in the future. So the application will work only for human emotions detection.

5.2 Comparative studies

At present, there are only a few apps and API's that can detect human emotions from images and video. But for audio, nothing found. So we built the software using one of the frameworks of python named 'Flask'. Despite having various drawbacks we are trying to upgrade the app. Now we are going to present a common API developed by 'Project Oxford by Microsoft'.

The following API focusses on the Computer Vision, language, and also speech analysis. But the emotion detection API works based on the photos. Initially, it determines the face and then responds in JSON. There the API could detect up to 7 emotions based on the photos and the percentage was also written on them where the best one could easily be selected. Here on the photo, it can be seen both of the people were in a happy mood and the API detected it with an accuracy of 99%. An example is given in Fig 5.2.1: API to detect human emotions.



```
Detection Result:
2 faces detected

JSON:
[
  {
    "faceRectangle": {
      "left": 120,
      "top": 362,
      "width": 255,
      "height": 255
    },
    "scores": {
      "anger": 6.506412e-7,
      "contempt": 0.00000107357334,
      "disgust": 0.0000137053685,
      "fear": 2.51182275e-9,
      "happiness": 0.9994379,
      "neutral": 0.000546224066,
      "sadness": 1.46409562e-7,
      "surprise": 2.88747827e-7
```

AGAIN!
Nordic APIs founders Travis Spencer and Andreas Krohn – 99% happy

Fig 5.2.1: API to detect human emotions

However, a few emotions detection technologies are described in the table:

Table 5.2.2: Comparative Applications

Sl. No	**App or API Name**	**Features**	**References**
1	NVisio	Video Analytics, 3D facial images	[31]
2	Kairos	Video Analytic	[31]
3	Sightcorp	Facial Points, Eyes, gaze, etc.	[31]
4	Face++	Photos, Tagged Photos from online	[31]
5	Crowd Emotion	Facial Points from real-time videos	[31]

5.3 Application Framework

To develop and host the app on the web we used the Flask framework and other things are described below:

Table 5.3.1: Application Framework's Details

Framework Name	Flask
Local Host Address	127.0.0.1:5000
App Name	Emotion Recognizer
IDE	Pycharm
Language	Python
Flask Version	1.1.2
Python Version	3.3.6
Environment and OS	Ubuntu 18.04 (Virtual)
Tested on	Mozilla Firefox

Our app was built using the environment mentioned in table 5.3.1 where we specifically showed necessary documents in detail. During running the developed app we came to know bugs of Flask like running a function on a single thread! So our planned app was changed a bit. However, in the future, we shall overcome this crisis.

5.4 Activity Diagram

The activity diagram of the developed app is given below in fig 5.3.1. Where it's shown that the very first step is to run the app. Now the application can be run from the command line using the working directory or it can be directly run from the Pycharm. Pycharm has its own built-in Terminal. So in order to run from PyCharm

Pycharm(Terminal Command):

- 1st run : $export FLASK_APP = app name run
- 2nd run: $ flask run

Within a few moments, the app will be hosted at 127.0.0.1:5000

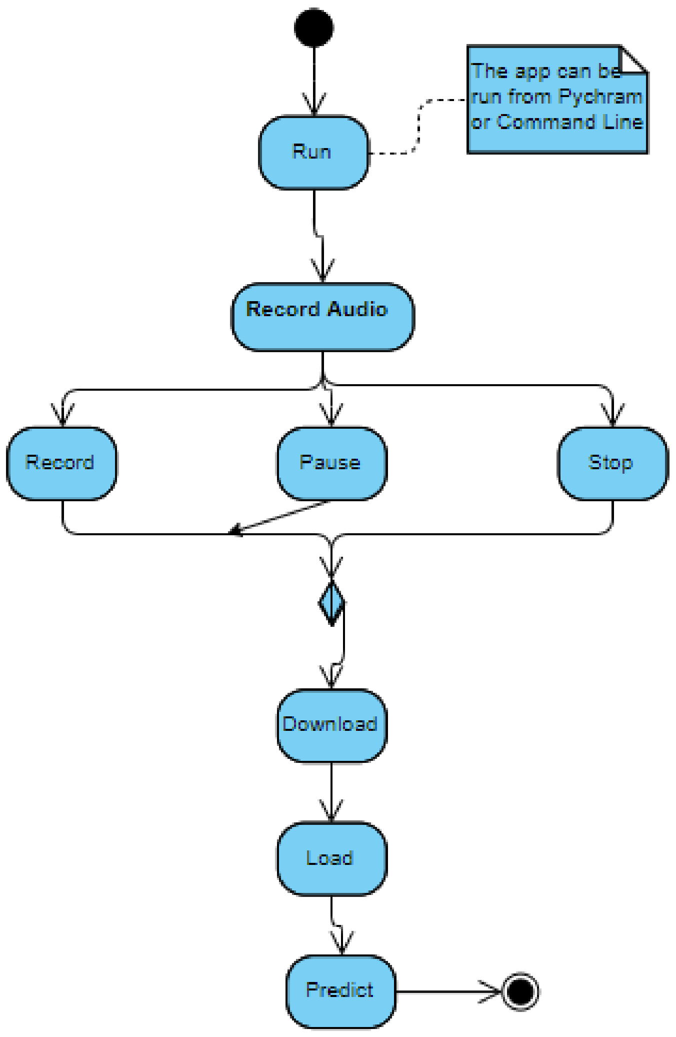


Fig 5.4.1: Application Framework

Another available option is running from the Ubuntu terminal.

To do so:

- Click on: "ctrl+alt+t", it will open the terminal
- Change the directory to the app directory
- Export the app using 'export FLASK_APP' command
- Run: 'Flask run' command

The app will be hosted at the localhost: 127.0.0.1:5000

Here the 5000 is the port address for running flask application. From the activity diagram, it is visible that the user will record their audio then download the audio file, and then they should click on the predicted app to predict. But the problem is that predicting here using that could not be built dynamically because of the bug in Flask 1.1.2 version which is the latest version. We did not degrade the current version else the problem could have been solved. It’s a possibility but in the next version of the flask, they will serve it removing these bugs.

5.5 Challenges

Despite having bugs in the current version of the flask (which we only came to know during running our program), we tried our best to develop and design a user-friendly application where user can easily record, play their audio, and then detect their emotions very easily. The challenges are mentioned below:

- Creating a virtual environment, we had run the app in a newly created virtual environment on Ubuntu 18.04
- Working with the requests like ‘POST’ or ‘GET’ was not easy because we did not find a better way to send variables from one request to another request, even sometimes using the python class could not help it out.
- Installing various dependency libraries and their peer libraries gave us a nightmare experience.
- We failed to upload the file on a live session because the necessary file uploading process could not be run because the flask used it on a single thread (that was a bug of the Flask).

5.6 Why Flask?

Researching on the web we came to know that machine learning models are very easy to deploy on desktop or android devices but running a Deep Learning Models is not that easy. But it is possible since there is a framework available like ‘Flask’ or ‘Django’. We also came to know that flask can be handier and user friendly than Django, so we selected FLASK over Django. But the deployment of the project could also be done using Django too. Maybe the problem would have gone if we tried that way.

5.7 Work schedule both for research and development

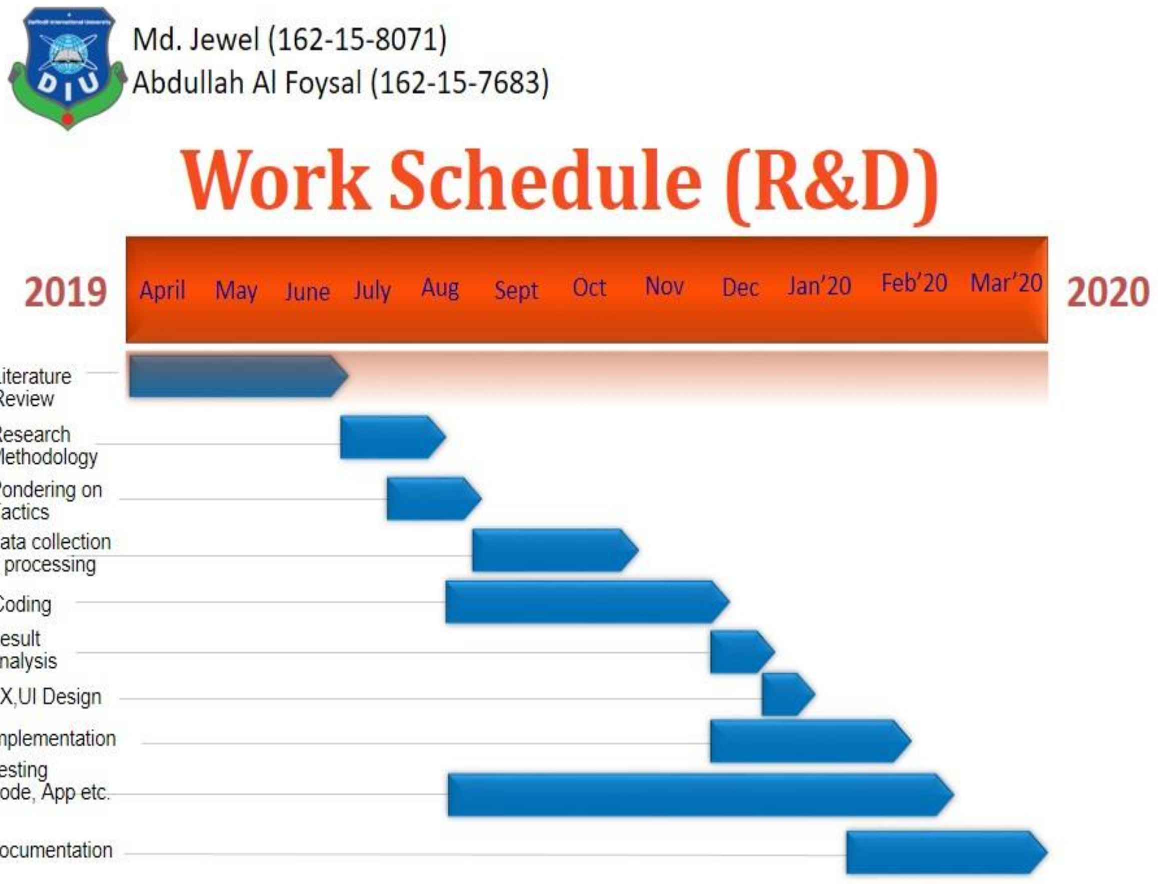


Figure 5.7.1: Gantt chart of the full thesis (Research and Development)

The following Gantt chart is for both research and development where different time schedules have been given so that the proper management of the thesis and the interaction with our supervisor can easily be understood.

In the Gantt chart, we can see the testing time was huge it took almost 6 months (Aug'2019 to February'2020) because we started testing as soon as we inaugurated coding and throughout the rest of the work we had to test various things which include designing the app, running the app, checking its accuracy, etc.

5.8 Business process modelling

Business process modelling generally helps to understand the analytical view of depiction or the main illustration for any company or organization. That’s why it is a very crucial element for any organization.

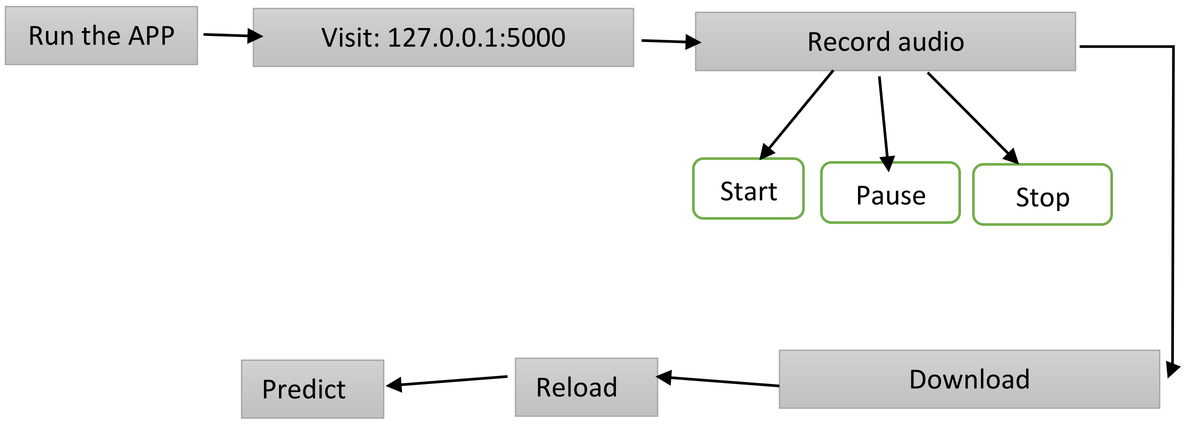


Figure 5.8.1: Business Process Model

Fig 5.8.1 shows that the business process modeling is very straight forward to understand the whole project in a nutshell.

5.9 Requirement collection and analysis

This stage, requirement collection, and analysis are for having a better knowledge of how the app can be run on which preconditions since without the fulfillment of any dependencies libraries and utilities the app may not run let alone giving accurate prediction.

- Installation of everything mentioned on table 5.3.1
- Having python installed. (>= 3.6)
- Flask Installed
- Available port number:5000 (the app will use that port)
- Use a better microphone to record your audio
- Try to record in a silent room for better accuracy
- We trained the model using the English dataset so it’s better if you could use English while testing.

Tools requirements to develop the projects:

Tools and the requirements of the project are clearly given on the Application Framework segment Table no: 5.3.1. Where we also mentioned the proper version number so that the users do not have any confusion. But here we would like to give some dependency libraries to run the application and we had run the application on Ubuntu 18.04.

- Tensorflow 2.0
- Keras
- Numpy
- Sklearn
- Matplotlib
- Pandas
- Librosa

Apart from these libraries we had installed more than a dozen of libraries during the implementation of the project.

5.10 Use case modelling and Description

5.10.1 Use case modelling

The Use Case Model defines the suggested feature of the new system. A Use Case is a discrete unit of user interaction with the system (human or machine). Cases of use are typically linked to 'actors.' An actor is a human or machine entity that executes a meaningful job that interacts with the scheme. Here is the use case model for our application as shown in Figure 5.10.1.1.

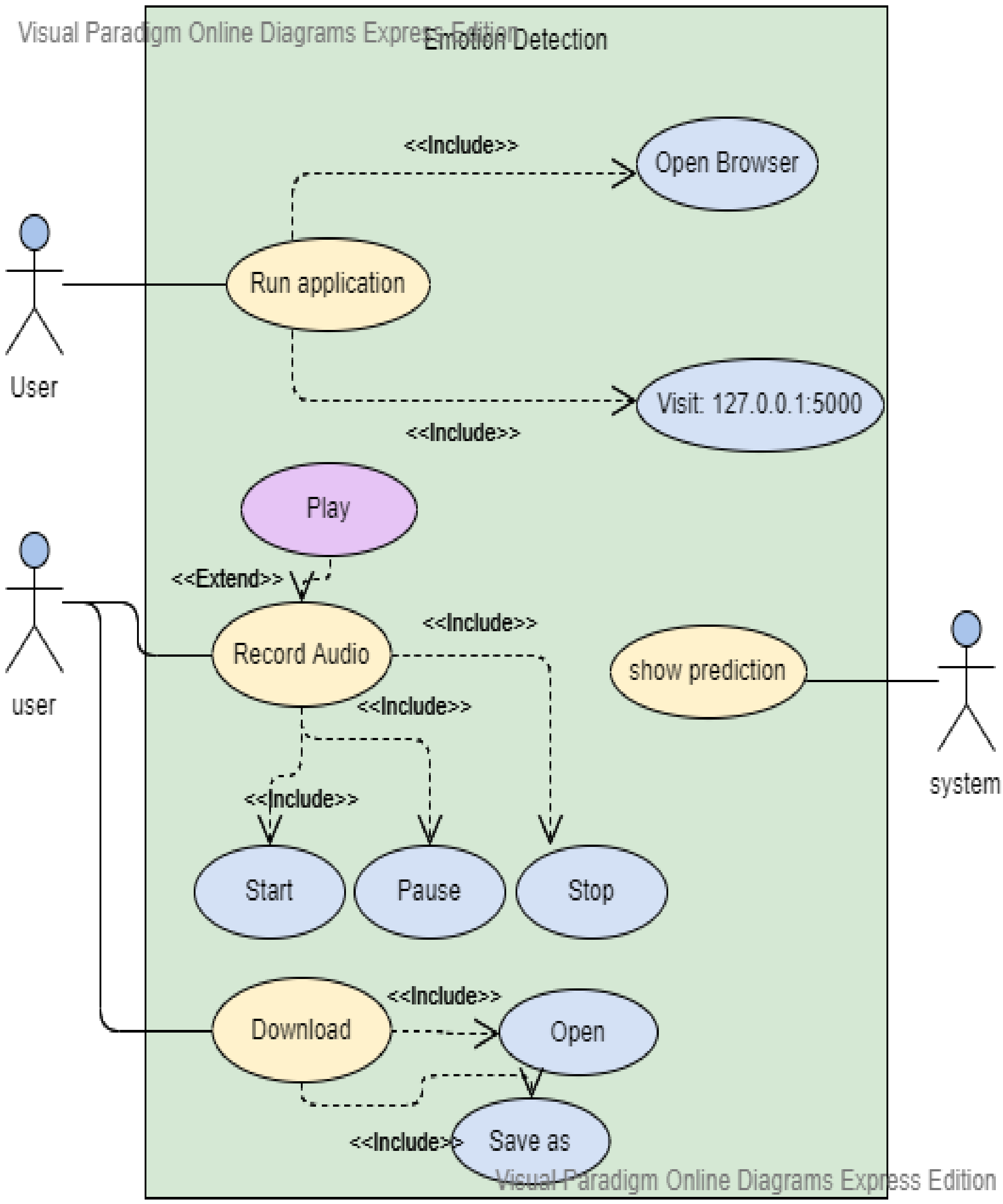


Figure 5.10.1.1: Use case modeling

5.10.2 Use case Description

Table 5.10.2.1: Use Case Description of 'Run Application'

Use Case ID:	01
Use Case Name:	Run Application
Created by:	Md. Jewel
Date Created:	03/03/2020
Description:	The user will run the application either using the command line of the project directory or from any IDE that supports python, we used Pycharm.
Primary Actor:	User
Secondary Actor:	None
Preconditions:	The user has to successfully run the application installing all dependency libraries.

Table 5.10.2.2: Use Case Description of 'open browser'?

Use Case ID:	02
Use Case Name:	Open browser
Created by:	Md. Jewel
Date Created:	03/03/2020
Description:	At this stage, the user will open any browser (we tested it on Mozilla Firefox)
Primary Actor:	User

Secondary Actor:	None
Preconditions:	Users must run the application.

Table 5.10.2.3: Use Case Description of ‘visit the page’?

Use Case ID:	03
Use Case Name:	Visit the page
Created by:	Md. Jewel
Date Created:	03/03/2020
Description:	Here, the user will type on the URL bar: 127.0.0.1.:5000 and hit enter which will prompt the desired window!
Primary Actor:	User
Secondary Actor:	None
Preconditions:	Users must open the browser.

Table 5.10.2.4: Use Case Description of ‘Record Audio’?

Use Case ID:	04
Use Case Name:	Record Audio
Created by:	Md. Jewel
Date Created:	03/03/2020
Description:	At this stage, the user has to record his speech or audio in order to classify the emotions.

Primary Actor:	User
Secondary Actor:	None
Preconditions:	User must visit the page

Table 5.10.2.5: Use Case Description of ‘Download’?

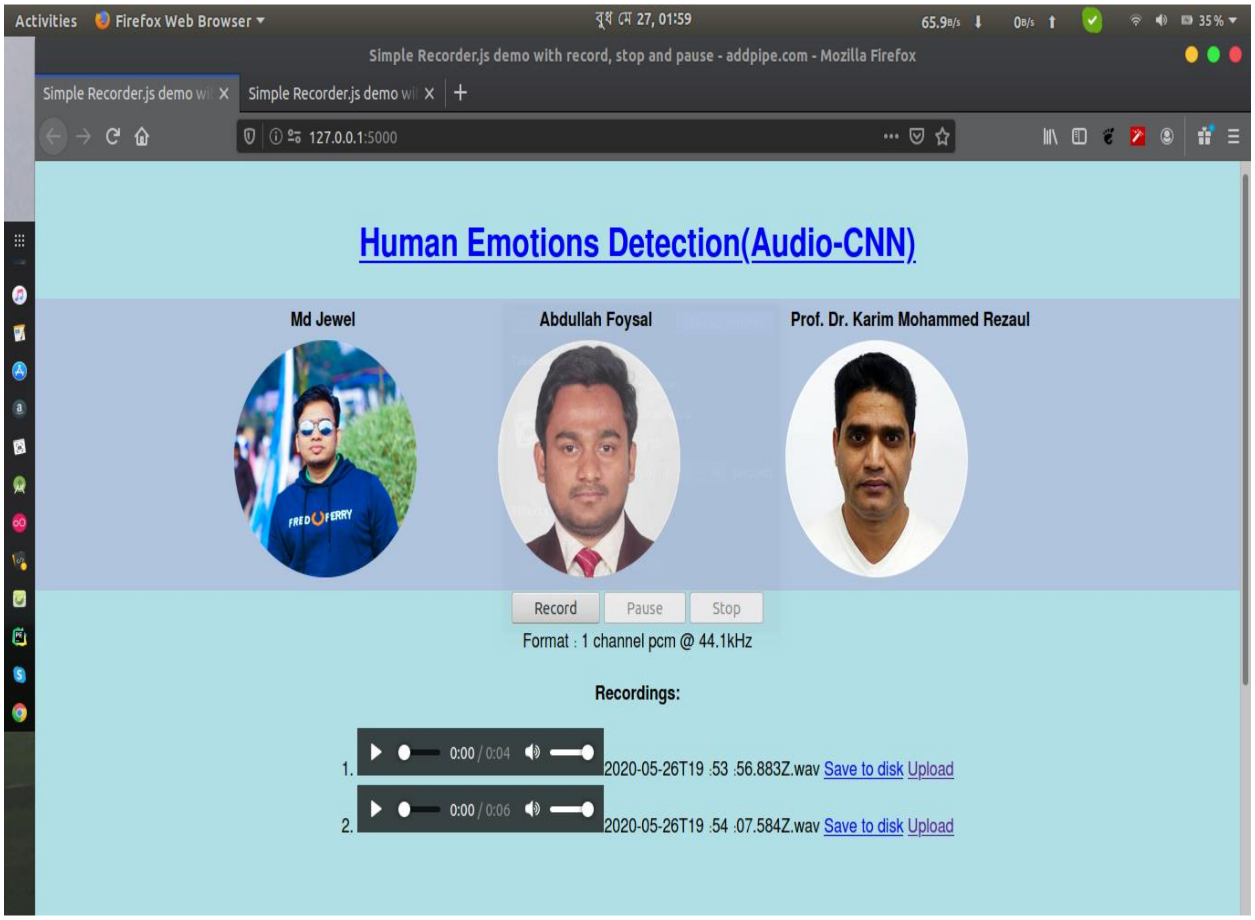
Use Case ID:	05
Use Case Name:	Download
Created by:	Md. Jewel
Date Created:	03/03/2020
Description:	Here at this stage, the user will download all of the recorded audio clips for future predictions on them.
Primary Actor:	User
Secondary Actor:	None
Preconditions:	Users must record the audio.

Table 5.10.2.6: Use Case Description of ‘Predict’?

Use Case ID:	06
Use Case Name:	Predict
Created by:	Md. Jewel
Date Created:	03/03/2020
Description:	This is the very last stage of the developed application. Here the

	user will only click on the button called predict and the system will show the result on the screen.
Primary Actor:	User
Secondary Actor:	None
Preconditions:	Users must fulfill all of the previous stages mentioned earlier, for example running the application was the first condition then, visiting the URL mentioned earlier, then record and download all of the audio clips.

5.11 A Full View of the Application Diagram


Activities
Firefox Web Browser
Simple Recorder.js demo with record, stop and pause - addpipe.com - Mozilla Firefox
127.0.0.1:5000
Human Emotions Detection(Audio-CNN)
Md Jewel
Abdullah Foysal
Prof. Dr. Karim Mohammed Rezaul
Record
Pause
Stop

Format : 1 channel pcm @ 44.1kHz

Recordings:

1. 2020-05-26T19 :53 :56.883Z.wav Save to disk Upload
2. 2020-05-26T19 :54 :07.584Z.wav Save to disk Upload

Fig 5.11.1: An overview of the application

Here in figure 5.11.1, we have given a basic view of the application which will be visible to the user as soon as he/she runs the application from the terminal or IDE built-in terminal. The user needs to also open the browser and visit 127.0.0.1:5000 where the application is hosted. Another way is that the user may click on the link which will be appeared when he/she runs the application typing: $flask run

5.12 Front end Design

Basically, we developed the application only for the desktop since running deep learning model or the deployment of any deep learning is not very feasible to run on android based application. But we have the plan to run the application on android in the future. We started designing the front end when we completed our testing on the testing dataset.

Designing using Html and jinja template which is generally used in python can make an application user friendly.

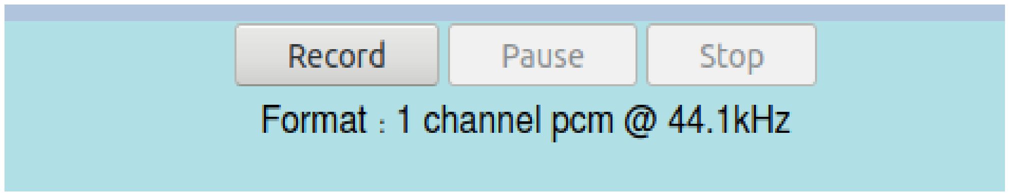


Fig 5.12.1: Simple audio recorder using Javascript

Here on fig 5.12.1, we showed the core module of the speech recording process, this module was build using Javascript (JS). JS provides very clear definitions and guidance to develops this sort of tool in their official documentation. That is how the recorder was made to work. It has 3 basic components that work individually as a whole.

- Record: If the user clicks on that button the speech or audio recording gets started, but here something must be done carefully that the audio can be recorded using external headphones or the internal built-in microphone of the desktop or device. So the system should be error-free on the contrary the recorded audio will work better if the user records English clips.
- Pause: This is just a simple way to pause during recording the audio. This option helps

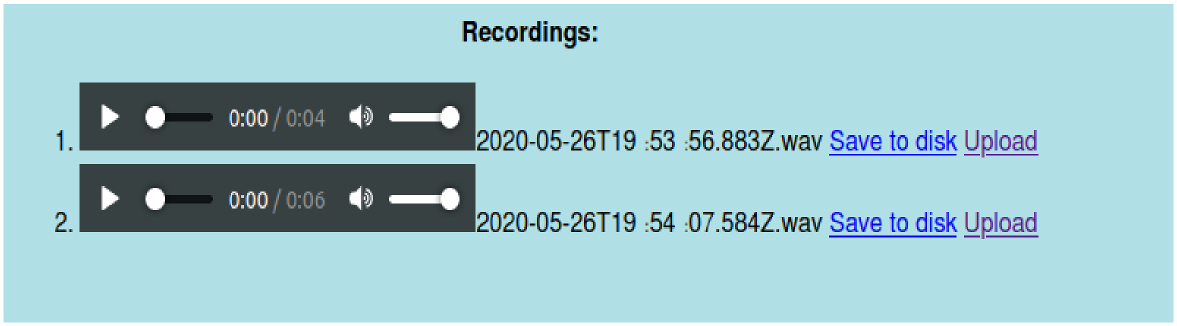
the user to record a better, clear, and specific speech for emotion detection.

- Stop: This button is given to stop the recording process. As soon as the user clicks on that button the recorded speech gets displayed on the screen for further processing like the user may play his/her own voice before downloading it.

When the recording is done the format will be displayed just underneath these buttons. Which is visible on the following figure.

Format: @xx. KHz

Here xx is the number of Kilo Hertz which expresses the frequency of the audio. Basically, that was developed in the JavaScript documentation site for audio recording.


Recordings:

1. 0:00 / 0:04 2020-05-26T19 :53 :56.883Z.wav Save to disk Upload
2. 0:00 / 0:06 2020-05-26T19 :54 :07.584Z.wav Save to disk Upload

Fig 5.12.2: Recording Samples

The user can record multiple speech or audios at a time. All of them will be displayed on the page. Figure 5.12.2 is showing how the user can take the benefit of recording multiples files at a time. When the recording is finished then they are given chronological order on the web page like 1,2,3 and so on.

The user has also the option to play the recorded audio for more clarity so that he/she can decide whether it should be downloaded or not. If they think that the sound was ok for prediction then they may click on the button available at the right side.

- Save to disk: If the user clicks on that button a new window will prompt to get permission from the user. The window will ask to know whether to save the file on the disk or to open it with any browser available on the desktop. Here if the user clicks on save then the recorded file will be saved on the default directory that was set by the user for his browser earlier.

- Upload: We had no plan to upload directly from here, as a result, we made this button disabled.

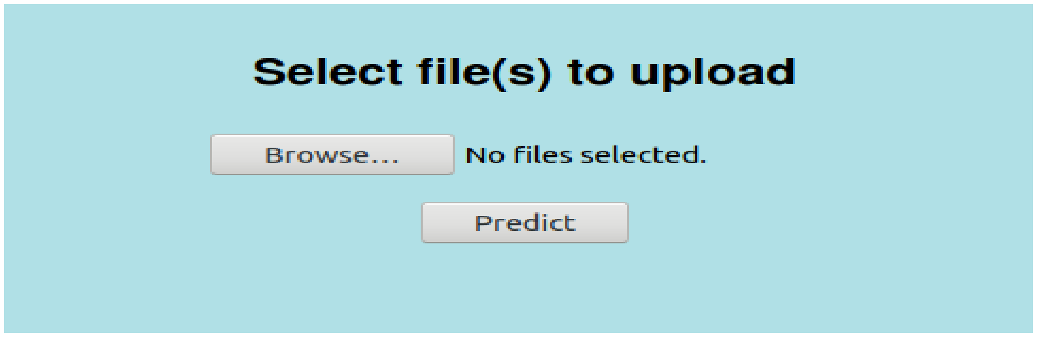


Fig 5.12.3: Uploading Files

That was the most critical and tricky part of our project, the file can be uploaded on the temporary server but here a request if also thrown which is ‘POST’. So we checked when the request is POST the directory of the uploaded file will be saved in order to predict in future. For predicting we need an audio file more specifically the path or location of the downloaded audio file. The file was downloaded earlier.

But the button ‘Predict’ throws another POST request, and for sending variables from one request to another, there is no direct way available in FLASK, so we to make it work we made a global variable after many difficulties and then new bugs arrived. From StackOverflow, we came to know that the particular program of the flask was running on a single thread, which is a bug of Flask for the current version and they will upgrade it as soon as possible. So we had to manually do the job!

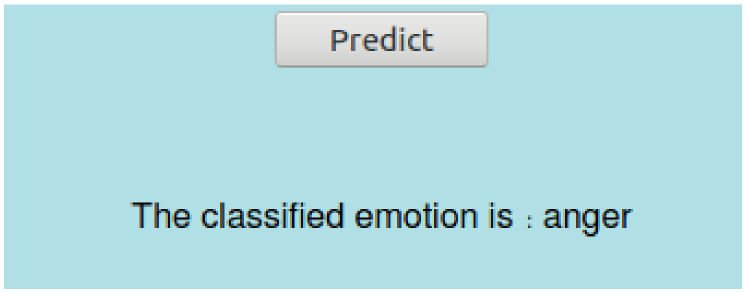
The specific task of the following button showing on figure 5.12.3 is given below:

- Browse: Browse button is for browsing file for prediction and uploading it to the present working directory. As soon as the user clicks on that button a new window will prompt to load or select the file that will be used for testing purposes. Then the user should mark the file and click on the select button available on the screen. But the user cannot select multiple files at a time nor he can mark any files other than .wav extension.

  Format: xx.wav

  Here, xx is the filename, and .wav is the extension of the file that we recorder using the simple javascript recorder.

- Predict: The predict button is for showing the final result after the application of various algorithms. As soon as the user clicks on the predict button the result gets displayed on the screen. The result is showing the same as showing on figure 5.12.4,



The classified emotion is : anger

Fig 5.12.4: Predicting the result

Here, when the user clicks on predict then the following text gets displayed,

Prediction: The classified emotion is: ‘anger’

The classified emotion can be any of the classes mentioned in the ‘**Table 3.4.2.2: Labels of the dataset’.** There we mentioned that we had 7 classes available. So the final result can be one of the following 7 classes.

Back-end:

We had integrated the Jinja template with flask and Html for the front design. But to load the file and to train the model a dataset was required initially and it is a total loss of time to train and test the model for the same piece of the dataset. So we saved the trained weights and then re-used it in for future prediction.

5.13 Implementation Issues:

While uploading the file a ‘POST’ request is sent which will be used on another ‘POST’ request at the time of prediction. But flask does it on a single thread! As a result, it causes an internal server error issue. So the uploading file was totally unsuccessful and we had to find out other solutions for the rest of the job.

5.13.1 Solutions of the Issue:

In order to make it work, we have to manually do the job. The sequential order of these task has been described below:

- Go to the directory of the downloaded files
- Cut it to the present working directory where the path is set, in our case we set it at: /demo_audio directory.
- Rename and run the file, renaming should be like this:
  Rename: ./demo_audio/demo_audio.wav

Here, demo_audio is the directory, and demo_audio.wav is the file name after the renaming. That's how we made it ready for the deployment. As soon as the Flask version will be upgraded we shall change it too.

5.14 Implementation Requirements

We had to install various dependency libraries. A few of them is given below

- Render_template
- Wave
- Request from flask
- Secure_filename
- url_for from flask
- io from skimage
- model_from_json from keras.models
- get_audio_features from utils.feature_extraction

Those are some major and important libraries that we needed to merge with the Flask framework during the implementation of the project.

5.15 Testing Implementation and Reports

A test case is a set of circumstances or variables to determine whether a test scheme meets or operates correctly. The test case development method can also assist define problems in the demands or design of an application. Here Table 5.15.1 shows that the Human Emotion Detection Test Cases.

Table 5.15.1: Test case table for Human Emotion Detection

No	Tested Case	Test Input	Expected Outcome	Actual Outcome	Result	Tested on
1	Run the application	Tested on 2 platforms, 1. Mozilla Firefox. 2. PyCharm IDE was also used for managing files easily.	Successfully run the application.	Ran the app successfully	Passed	1 February, 2020
2	Run the URL	Ran using 2 methods 1. On Mozilla: 127.0.0.1.:5000 2. From Pycharm click on the URL	Load the page	Successfully the page was loaded.	Passed	1 February, 2020

3	Record the Speech	Tested on these platforms, 1. Mozilla Firefox. 2. PyCharm IDE. Environment: Ubuntu 18.04	Record and play the recorded clips.	The recorded clips were played successfully.	Passed	2 February, 2020
4	Download the clips	Tested on these platforms, 3. Mozilla Firefox. 4. PyCharm IDE. Environment: Ubuntu 18.04	Download the clips on a particular path	Successfully downloaded the clips	Passed	2 February, 2020
5	Make Prediction	Tested on these platforms, 1. Mozilla Firefox. 2. PyCharm IDE. Environment: Ubuntu 18.04	Predict the loaded audio or speech and classify its emotion	Successfully the emotion detection system worked	Passed	3 February, 2020

5.11 Future Scope

Implementation of any deep learning model is always welcomed in the modern era of science and technology because this is something which is the prime target of the researcher. Without implementing any machine learning or deep learning models, it cannot give the human race a way to deploy the advantages of science. What we showed and implemented in this thesis is a unique task so far. Despite having some issues during the implementation of the project, it was so far so good. As soon as the flask version will be upgraded fixing the issues of running a request on a single thread, we shall make the next version of the developed application too. At the same time, we must confess regarding the quality of the dataset. We shall add more clear data for both of the music instrumental sound classification and human emotion detection process in order to get more accuracy and less percentage of loss.

CHAPTER 6

SUMMARY, CONCLUSION, RECOMMENDATION, AND IMPLICATION FOR FUTURE RESEARCH

6.1 Summary of the study

Machine or bots nowadays can perform several tasks that are generally performed by a human. The sole purpose of our thesis was to pave a way and hasten the process of development in the field of machine learning and deep learning. At the very primary stage of this thesis, we classify sample audio from different sources and we predicted using the audio classification techniques where we used both of the options available for audio classification, which were CNN and RNN. Generally, CNN is used to classify images that's why we could not apply the algorithm directly on the dataset we had preprocessed it in many ways. So we made those audio clips suitable for the CNN model (proposed model) and we also preprocessed it for the RNN model (proposed model). We initially expected that the RNN would perform better than CNN but with a great surprise, we noticed that CNN performed much better than CNN. That's how we concluded that CNN will perform better than RNN. Then we build another CNN model for human emotion detection. Human emotions have been detected using various tactics but based on voice, there were only a handful of jobs done. So we were very curious about that research. But the big problem was the dataset since recording audios when there is background noise and the quality of the recording device or the headphone is another issue. Again language made it more difficult since in Bengali (or any other languages) the expression varies for both males and females. The voice changing has a deep impact on the accuracy of the model. Even a very good model does not show better accuracy when the dataset is not handled or processed properly. We removed the death spot from our dataset during the Instrumental Sound Classification but for human detection, we did not do this because of the interaction between background noise and human speeches. After all, we integrate and made a secondary dataset. The dataset was in English. At last, we trained our proposed model on Google Colab and saved the weights to use in the future, it is because the training of most of the deep learning model takes huge time. So the trained weights might give us a solution to this issue.

After that when the coding stages were done we keep working for the implementation because it was a research and development thesis and we build our model that can differentiate among various expressions of humans without seeing their faces. For the deployment of the project, we elected the Flask framework over Django. Flask is very easy to use moreover we had experience working with Flask. So we hosted the app on the localhost (address: 127.0.0.1:5000). Where the user can record or play their voice using the external or internal microphone, then they can download the file, and then a prediction can be performed for a single audio clip only.

6.2 Conclusions

From this research, we have come to a point where we can say that the Convolutional Neural Network (CNN) works better than Recurrent Neural Network (RNN) based on LSTM (Long Short Term Memory). Since for Music Instrumental Sound Classification, we got around 95% accuracy on CNN and 87% accuracy on LSTM (a type of RNN). So that's why for human emotion detection we applied the CNN algorithm. The accuracy for this operation was not high but this was standard comparing other CNN or RNN models built for classification on the audio dataset. During feature extraction, various things arrive and hinder to achieve good accuracy for audio and speech classification. Again we worked to classify emotions so the same piece of sound can be different from person to person. We got around 60% accuracy after the application of the CNN algorithm for the purpose of human emotion detection. One of the main things of emotion detection was the pitch of the speeches. The same piece of speech may give a different prediction when the pitch is different. Again for males and females, the recorded audio shows different pitch as a result the outcome might be different. That's why working with audio is not easy and we successfully developed our proposed model with good accuracy compared with existing models based on audio datasets. One of the main attractions of our thesis is its implementation part. Very few deep learning models can be deployed but nowadays Tensorflow officially has provided platforms like Tensorflow.js to host various models on the web. However, we hosted our developed model of human emotion detection on the local host using Flask. Where users can record and detect their emotion types.

6.3 Limitations

- For music instrumental audio classification, small dataset was used.
- Background noice may cause wrong prediction.
- Changes in the accent may cause wrong prediction.
- The quality of the microphone during recording may create issues.
- Changes in the voice for respective genders may create issue despite of using both types of data.
- Issues created by the current FLASK version.

6.4 Implication for Further Study

We have a lot of things to do when the next version of the flask will be published. We are expecting this will arrive on public very soon, but in case if they don’t upgrade the current version then we shall degrade the installed Flask version in order to make it work, most of the previous versions of flask support multiple processes on different threads. Apart from this, we shall update our dataset and we will also make our primary dataset for the Bengali language to detect emotions when we’ll try to add more features in the upcoming days. In a nutshell, we shall try our best to upgrade the accuracy of the Human Emotion Detection model.

CONTRIBUTION FROM THIS R&D PROJECT

1. An Application has been developed for human emotion detection on a live session.

2. International paper-1: Scopus indexed

   Md. Jewel, Abdullah Al Foysal and Karim Mohammed Rezaul, “Music Instrumental Sounds classification using CNN and RNN”, Submitted *to* Journal of *International Journal of Innovative Technology and Exploring Engineering (IJITEE) in India*, March 2020. (Accepted)

3. International paper-2: Scopus indexed

   Md. Jewel, Abdullah Al Foysal and Karim Mohammed Rezaul, “Music Instrumental Sounds classification using CNN and RNN”, Submitted *to* conference of *“8th Medinah International Conference on Multidisciplinary Issues (ICMI-2020)”*, April 2020. (Accepted)

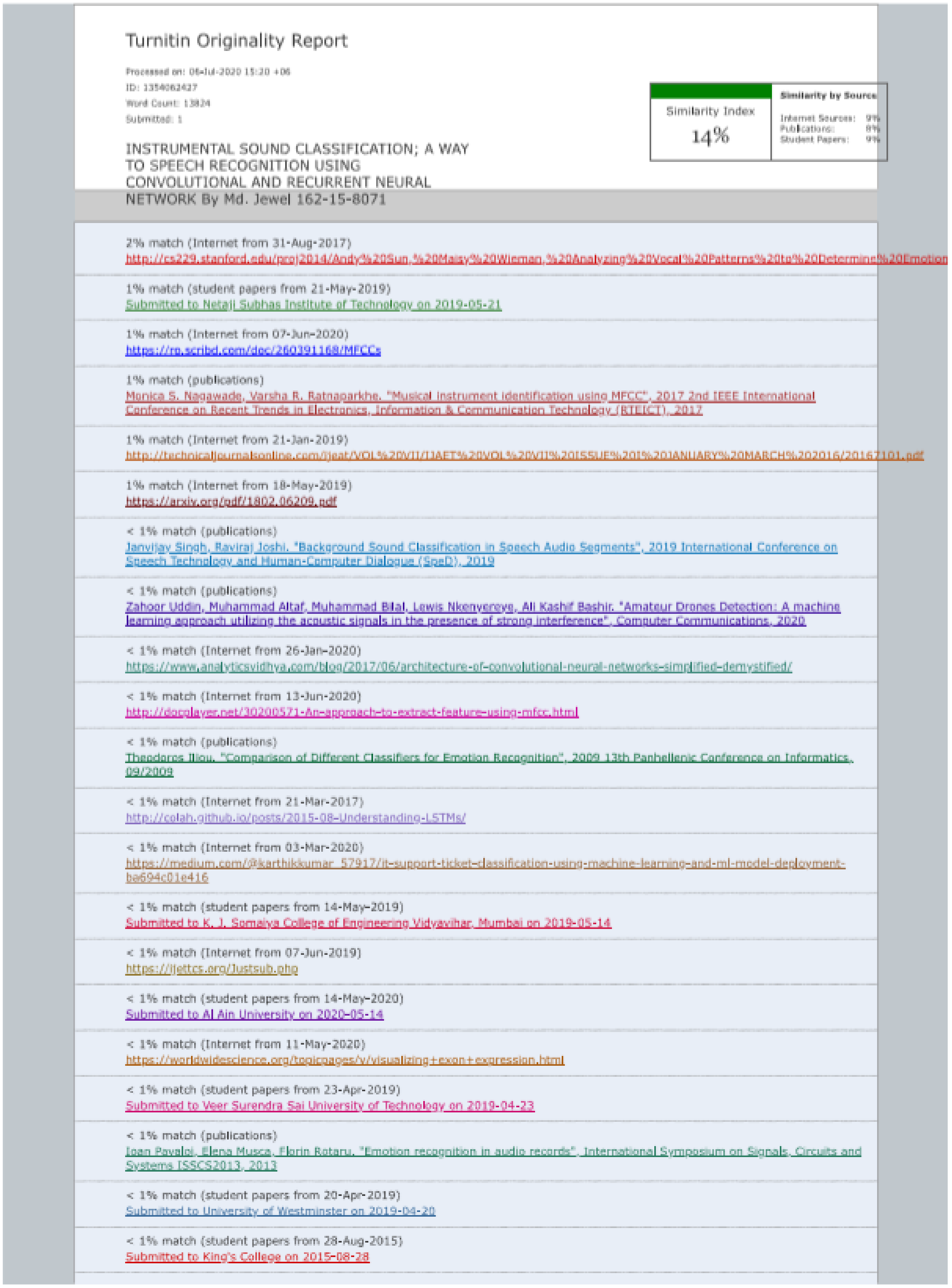
REFERENCES

1. D. G. Bhalke, C. B. Rama Rao, D. S. Bormane, “Automatic musical instrument classification using fractional fourier transform based- MFCC features and counter propagation neural network”, Journal of Intelligent Information Systems, 2016, Volume 46, Number 3, Page 425
2. Monica S. Nagawade, Varsha R. Ratnaparkhe, “Musical Instrument Identification using MFCC”, 2017 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT), May 19-20, 2017, India.
3. T. Virtanen, M. D. Plumbley, and D. Ellis, Computational Analysis of Sound Scenes and Events. Springer, 2018.
4. Sharath Adavanne and Tuomas Virtanen. Sound event detection using weakly labeled dataset with stacked convolutional and recurrent neural network. In DCASE Workshop, 2017.
5. Sharath Adavanne, Konstantinos Drossos, Emre Çakır, and Tuomas Virtanen. Stacked convolutional and recurrent neural networks for bird audio detection. In EUSIPCO, 2017.
6. Miroslav Malik, Sharath Adavanne, Konstantinos Drossos, Tuomas Virtanen, Dasa Ticha, and Roman Jarina. Stacked convolutional and recurrent neural networks for music emotion recognition. In Sound and Music Computing Conference (SMC), 2017.
7. Jordi Pons, Thomas Lidy, and Xavier Serra. Experimenting with musically motivated convolutional neural networks. In Content-Based Multimedia Indexing (CBMI) Workshop, pages 1–6, 2016.
8. Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In IEEE ICASSP, pages 6964–6968, 2014.
9. Tara N Sainath, Ron J Weiss, Andrew Senior, Kevin W Wilson, and Oriol Vinyals. Learning the speech front-end with raw wave form cldnns. In Sixteenth Annual Conference of the International Speech Communication Association, 2015.
10. Wei Dai, Chia Dai, Shuhui Qu, Juncheng Li, and Samarjit Das. Very deep convolutional neural networks for raw waveforms. In IEEE ICASSP, pages 421–425, 2017.
11. Meinard Müller, Member, IEEE, Daniel P. W. Ellis, Senior Member, IEEE, Anssi Klapuri, Member, IEEE, and Gaël Richard, Senior Member, IEEE. “Signal processing for music analysis”. IEEE Journal of selected topics in signal processing, VOL. 5, NO. 6, OCTOBER 2011
12. Priyanka S. Jadhav, “ Classification of Musical Instruments sounds by Using MFCC and Timbral Audio Descriptors”IJRITCC, ISSN: 23218169, Volume: 3 Issue: 7, 5001 – 5006
13. Slim Essid, Ga¨el Richard, and Bertrand David “Efficient musical instrument recognition on solo performance music using basic features”, AES 25th international conference, london, united kingdom, 2004 june 17–19
14. M. Erdal Ozbek , Nalan Ozkurt and F. Acar Savaci, ” Wavelet ridges for musical instrument classification”,J Intell Inf Syst (2012) 38:241–256, DOI 10.1007/s10844-011-0152-9
15. Farbod Foomany and Karthikeyan Umapathy, “Classification of music instruments using wavelet-based time-scale features”, 2013 IEEE ICMEW.

16. Chauhan, P. M., & Desai, N. P. (2014). Mel Frequency Cepstral Coefficients (MFCC) based speaker identification in noisy environment using wiener filter. 2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE).
17. Karpagavalli S and Chandra E, " A Review on Automatic Speech Recognition Architecture and Approaches", International Journal of Signal Processing, Image Processing and Pattern Recognition Vol.9, No.4, (2016), pp.393-404
18. C. Poonkuzhali, R. Karthiprakash, S. Valarmathy and M. Kalamani, An Approach to feature selection algorithm based on Ant Colony Optimization for Automatic Speech Recognition, International journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, 11(2), and 2013.
19. Yuan Meng, Speech recognition on DSP: Algorithm optimization and performance analysis, The Chinese university of Hong Kong, July 2004, pp. 1-18.
20. Lindasalwa Muda, Mumtaj Begam and I. Elamvazuthi, Voice recognition algorithm using MFCC & DTW techniques, Journal Of Computing, Volume 2, Issue 3, March 2010, ISSN 2151-9617, pp. 138-143.
21. Fredric J. Harris, Mexber, IEEE, "On then Use of Windows for Harmonic Analysis with the Discrete Fourier Transform"in Proceeding of the IEEE, January 1978.
22. Parwinder Pal Singh, Pushpa Rani," An Approach to Extract Feature using MFCC", IOSR Journal of Engineering (IOSRJEN) ISSN (e): 2250-3021, ISSN (p): 2278-8719 Vol. 04, Issue 08 (August. 2014), ||V1|| PP 21-25
23. Siddhant C. Joshi, Dr. A.N.Cheeran, "MATLAB Based Feature Extraction Using Mel Frequency Cepstrum Coefficients for Automatic Speech Recognition", International Journal of Science, Engineering and Technology Research (IJSETR), Volume 3, Issue 6, June 2014
24. Yuan Meng, Speech recognition on DSP: Algorithm optimization and performance analysis, The Chinese university of Hong Kong, July 2004, pp. 1-18.
25. Sirko Molau, Michael Pitz, Ralf Schl¨uter, and Hermann Ney, Computing Mel-frequency cepstral coefficients on the power spectrum, University of Technology, 52056 Aachen, Germany
26. Gaurav, Devanesamoni Shakina Deiv, Gopal Krishna Sharma, Mahua Bhattacharya, Development of Application Specific Continuous Speech Recognition System in Hindi, Journal of Signal and Information Processing, 2012, 3, pp. 394-401.
27. E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S.Oramas, A.Porter, and X.Serra, "Free sound datasets: a platform for the creation of open audio datasets,"in Proceedings of the 18th International Society for Music Information RetrievalConference(ISMIR2017), Suzhou, China,2017,pp. 486–493.
28. Eduardo Fonseca1∗, Manoj Plakal2, Frederic Font1, Daniel P. W. Ellis2, Xavier Favory1, Jordi Pons1, Xavier Serra1, "General-purpose tagging of free sound audio with audio set labels: task description, dataset, and baseline", Detection and Classification of Acoustic Scenes and Events 2018.
29. M.M.H.E. Ayadi, M.S. Kamel, F. Karray, "Survey on speech emotion recognition:Features classification schemes and databases", Pattern Recognition, pp. 572-587, 2011.
30. Liberman, Mark, et al. Emotional Prosody Speech and Transcripts LDC2002S28. Web Download.

Philadelphia: Linguistic Data Consortium, 2002.

31. To know about existing audio API's <https://nordicapis.com/20-emotion-recognition-apis-that-will-leave-you-impressed-and-concerned/>



Turnitin Originality Report

Processed on: 06-Jul-2020 15:20 +06
ID: 1354062427
Word Count: 13824
Submitted: 1

INSTRUMENTAL SOUND CLASSIFICATION; A WAY TO SPEECH RECOGNITION USING CONVOLUTIONAL AND RECURRENT NEURAL NETWORK By Md. Jewel 162-15-8071

Similarity Index	Similarity by Source	
14%	Internet Sources:	9%
	Publications:	8%
	Student Papers:	9%

2% match (Internet from 31-Aug-2017)
http://cs229.stanford.edu/proj2014/Andy%20Sun,%20Maisy%20Wieman,%20Analyzing%20Vocal%20Patterns%20to%20Determine%20Emotion

1% match (student papers from 21-May-2019)
Submitted to Netaji Subhas Institute of Technology on 2019-05-21

1% match (Internet from 07-Jun-2020)
https://ro.scribd.com/doc/260391168/MFCCs

1% match (publications)
Monica S. Nagawade, Varsha R. Ratnaparkhe. "Musical instrument identification using MFCC", 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017

1% match (Internet from 21-Jan-2019)
http://technicaljournalsonline.com/ijeat/VOL%20VII/IJAET%20VOL%20VII%20ISSUE%20I%20JANUARY%20MARCH%202016/20167101.pdf

1% match (Internet from 18-May-2019)
https://arxiv.org/pdf/1802.06209.pdf

< 1% match (publications)
Janvijay Singh, Raviraj Joshi. "Background Sound Classification in Speech Audio Segments", 2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD), 2019

< 1% match (publications)
Zahoor Uddin, Muhammad Altaf, Muhammad Bilal, Lewis Nkenyereye, Ali Kashif Bashir. "Amateur Drones Detection: A machine learning approach utilizing the acoustic signals in the presence of strong interference", Computer Communications, 2020

< 1% match (Internet from 26-Jan-2020)
https://www.analyticsvidhya.com/blog/2017/06/architecture-of-convolutional-neural-networks-simplified-demystified/

< 1% match (Internet from 13-Jun-2020)
http://docplayer.net/30200571-An-approach-to-extract-feature-using-mfcc.html

< 1% match (publications)
Theodoros Iliou. "Comparison of Different Classifiers for Emotion Recognition", 2009 13th Panhellenic Conference on Informatics, 09/2009

< 1% match (Internet from 21-Mar-2017)
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

< 1% match (Internet from 03-Mar-2020)
https://medium.com/@karthikkumar_57917/it-support-ticket-classification-using-machine-learning-and-ml-model-deployment-ba694c01e416

< 1% match (student papers from 14-May-2019)
Submitted to K. J. Somaiya College of Engineering Vidyavihar, Mumbai on 2019-05-14

< 1% match (Internet from 07-Jun-2019)
https://ijettcs.org/Justsub.php

< 1% match (student papers from 14-May-2020)
Submitted to Al Ain University on 2020-05-14

< 1% match (Internet from 11-May-2020)
https://worldwidescience.org/topicpages/v/visualizing+exon+expression.html

< 1% match (student papers from 23-Apr-2019)
Submitted to Veer Surendra Sai University of Technology on 2019-04-23

< 1% match (publications)
Ioan Pavaloi, Elena Musca, Florin Rotaru. "Emotion recognition in audio records", International Symposium on Signals, Circuits and Systems ISSCS2013, 2013

< 1% match (student papers from 20-Apr-2019)
Submitted to University of Westminster on 2019-04-20

< 1% match (student papers from 28-Aug-2015)
Submitted to King's College on 2015-08-28

APPENDICES

Appendix A: Our Dataset links

Human Speech:

https://drive.google.com/open?id=1qAjCDYzeDd8bZpLMFf0aLY8XsSFTVt3Q

Instrumental Sounds:

https://github.com/NeoZian/Sound-Audio-Classification-with-CNN-RNN/tree/master/wavfiles

Appendix B: Our implemented Code Links

Human Speech:

https://github.com/NeoZian/Audio_emotion_detection/blob/master/audio_emotion.ipynb

Instrumental Sounds:

https://github.com/NeoZian/Sound-Audio-Classification-with-CNN-RNN