VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANASANGAMA, BELAGAVI – 590018

Final Project Report
On

SPEECH EMOTION RECOGNITION USING MACHINE LEARNING

Submitted in partial fulfillment for the award of degree of
Bachelor of Engineering
In
Computer Science and Engineering

Submitted by
1BG17CS051- Monish Goutham K
1BG18CS414- Nayan Surya N
1BG18CS406- Gishnu Govind
1BG18CS422- Santhosh A

Internal Guide
Mrs.Sajitha N
Assistant Professor, Dept of CSE, BNMIT

Vidyayāmruthamashnuthe

B.N.M. Institute of Technology

Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.
All UG branches – CSE, ECE, EEE, ISE & Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021
Post box no. 7087, 27th cross, 12th Main, Banashankari 2nd Stage, Bengaluru- 560070, INDIA
Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

Department of Computer Science and Engineering
2020-2021

B.N.M. Institute of Technology

(Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.
All UG branches – CSE, ECE, EEE, ISE & Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021
Post box no. 7087, 27th cross, 12th Main, Banashankari 2nd Stage, Bengaluru- 560070, INDIA
Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

Department of Computer Science and Engineering

Vidyayāmruthamashnuthe

CERTIFICATE

Certified that the project work entitled **Speech Emotion Recognition Using Machine Learning** carried out by **Mr.Monish Goutham K (1BG17CS051), Mr.Nayan Surya N (1BG18CS414), Mr.Gishnu Govind (1BG18CS406), Mr.Santhosh A (1BG18CS422)** bonafide students of VIII Semester, BNM Institute of Technology in partial fulfillment for the award of Bachelor of Engineering in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the year 2020-21. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said Degree.

Mrs. Sajitha N	**Dr. Sahana D Gowda**	**Dr. Krishnamurthy G N**
Assistant Professor	**Professor and HOD**	**Principal**
Department of CSE	**Department of CSE**	**BNMIT, Bengaluru**
BNMIT, Bengaluru	**BNMIT, Bengaluru**	

Name & Signature:

Examiner 1:

Examiner 2:

ABSTRACT

Communication is the key to express one’s thoughts and ideas clearly. Amongst all forms of communication, speech is the most preferred and powerful form of communications in human. The era of the Internet of Things (IoT) is rapidly advancing in bringing more intelligent systems available for everyday use. These applications range from simple wearables and widgets to complex self-driving vehicles and automated systems employed in various fields. Intelligent applications are interactive and require minimum user effort to function, and mostly function on voice-based input. This creates the necessity for these computer applications to completely comprehend human speech. A speech percept can reveal information about the speaker including gender, age, language, and emotion. Several existing speech recognition systems used in IoT applications are integrated with an emotion detection system in order to analyze the emotional state of the speaker. The performance of the emotion detection system can greatly influence the overall performance of the IoT application in many ways and can provide many advantages over the functionalities of these applications. This research presents a speech emotion detection system with improvements over an existing system in terms of data, feature selection, and methodology that aims at classifying speech percepts based on emotions, more accurately.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned my effort with success.

We would like to thank **Shri Narayan Rao R. Maanay,** Secretary, BNMIT, Bengaluru for providing excellent academic environment in the College.

We would like to sincerely thank **Prof. T. J. Rama Murthy**, Director, BNMIT, Bengaluru for having extended his support and encouraging me during the course of the work.

We would like to express my gratitude to **Prof. Eishwar N. Maanay**, Dean, BNMIT, Bengaluru for his relentless support, guidance and assistance.

We would like to thank **Dr. Krishnamurthy G.N,** Principal, BNMIT, Bengaluru for his constant encouragement.

We would like to thank **Dr. Sahana D. Gowda,** Professor and Head of the Department of Computer Science and Engineering, BNMIT, Bengaluru who has shared her opinions and thoughts which helped me in completion of my Project successfully.

We express our in-depth, heartfelt, sincere gratitude to **Prof. Raghavendra C K**, Project Coordinator, Associate Professor, BNMIT, Bengaluru, for her valuable suggestions and support.

We would also like to thank our project guide **Mrs. Sajitha N**, Assistant Professor, Department of Computer Science and Engineering, BNMIT, Bengaluru for guiding in a systematic manner.

MonishGouthamK(1BG17CS051)

NayanSuryaN(1BG18CS414)

GishnuGovid(1BG18CS406)

SanthoshA(1BG18CS422)

.

Table of Contents

List of Figures

Chapter 1

INTRODUCTION

For several years now, the growth in the field of Artificial Intelligence (AI) has been accelerated. AI, which was once a subject understood by computer scientists only, has now reached the house of a common man in the form of intelligent systems. The advancements of AI have engendered to several technologies involving Human-Computer Interaction (HCI). Aiming to develop and improve HCI methods is of paramount importance because HCI is the front-end of AI which millions of user's experiences. Some of the existing HCI methods involve communication through touch, movement, hand gestures, voice and facial gestures. Among the different methods, the voice-based intelligent devices are gaining popularity in a wide range of applications. In a voice-based system, a computer agent is required to completely comprehend the human's speech percept in order to accurately pick up the commands given to it. This field of study is termed as Speech Processing and consists of three components:

- Speaker Identification
- Speech Recognition
- Speech Emotion Detection

Speech Emotion Detection is challenging to implement among the other components due to its complexity. Furthermore, the definition of an intelligent computer system requires the system to mimic human behavior. A striking nature unique to humans is the ability to alter conversations based on the emotional state of the speaker and the listener. Speech emotion detection can be built as a classification problem solved using several machine learning algorithms. This project discusses in detail the various methods and experiments carried out as part of implementing a Speech Emotion Detection system. Reduced development time building applications that use database

1.1 Motivation

The recognition of emotional speech aims to recognize the emotional condition of an individual utterer by applying his/her voice automatically. Speech emotion recognition is mostly beneficial for applications, which need human-computer interaction such as speech synthesis, customer service, education, forensics and medical analysis. To solve the challenge of Speech Emotion Recognition we will be using a speech dataset consisting of Audio samples. Which are labelled as Angry, happy, sad, fearful, calm and neutral this dataset is utilized to validate the proposed methods in speech emotion recognition.

1.2 Problem Statement

Design and develop an application for Speech Emotion Recognition using machine learning technique. The main elements for speech emotion recognition system are same as they are for any classic pattern recognition system. It has speech with emotions as input, then features are extracted and then classification is done by taking algorithm.

The efficiency of the speech emotion recognition is highly depending upon the database. The collected speech signals will be extracted and are provided to the classification algorithm. Then file under the test is classified and recognizes the emotion of the test data. This reality motivates many researchers to consider speech signals as a quick and effective process to interact between computer and human. It means the computer should have enough knowledge to identify human voice and speech.

Although there is a significant improvement in speech recognition, researchers are still away from natural interplay between computer and human, since computers are not capable of understanding human emotional state. The recognition of emotional speech aims to recognize the emotional condition of an individual utterer by applying his/her voice automatically

1.3 Objectives

The objectives are:

- The primary objective of SER is to improve man-machine interface. It can also be used to monitor the psycho physiological state of a person in lie detectors.
- In recent time, speech emotion recognition also finds its applications in medicine and forensics.
- Characteristics of a human vocal tract and hearing system is represented by different features of speech signal.
- Speech emotion recognition is mostly beneficial for applications, which need human computer interaction such as speech synthesis, customer service, education, forensics and medical analysis.

1.4 Summary

Communication is the key to express one’s thoughts and ideas clearly. Amongst all forms of communication, speech is the most preferred and powerful form of communications in human. The era of the Internet of Things (IoT) is rapidly advancing in bringing more intelligent systems available for everyday use. These applications range from simple wearables and widgets to complex self-driving vehicles and automated systems employed in various fields. Intelligent applications are interactive and require minimum user effort to function, and mostly function on voice-based input. This creates the necessity for these computer applications to completely comprehend human speech. A speech percept can reveal information about the speaker including gender, age, language, and emotion. Several existing speech recognition systems used in IoT applications are integrated with an emotion detection system in order to analyze the emotional state of the speaker. The performance of the emotion detection system can greatly influence the overall performance of the IoT application in many ways and can provide many advantages over the functionalities of these applications. This research presents a speech emotion detection system with improvements over an existing system in terms of data, feature selection, and methodology that aims at classifying speech percepts based on emotions, more accurately

Chapter 2

LITERATURE SURVEY

A literature survey in a project report is that section which shows the various analyses and research made in the field of your interest and the results already published taking into account the various parameters of the project and the extent of the project. A Literature survey refers to getting the content from the books which are related to the topic or a given project. It should be referred from some research paper that is related to the topic. Any materials which are related to the project from the internet which is valuable for the student and has helped the student to enhance the report status as well as the calculation, analysis and tabulation majorly reflect in the survey. So, in this way, one can select the literature survey.

It is necessary to emphasize that it is the most important part in the project report. It is the most important part of the report as it gives the students a direction in the area of their research. It helps the students to set a goal for analysis - thus giving them their problem statement.

When one writes a literature review in respect of the project, they have to write the researches made by various analysts - their methodology (which is their abstract) and the conclusions they have arrived at. One should also give an account of how this research has influenced their thesis.

Literature surveys are needed for:

- To see what has and has not been investigated.
- To identify data sources that other researchers have used.
- To learn how others have defined and measured key concepts.
- To develop alternative research projects.
- To put one's perspective into work.
- To contribute to the field by moving research forward. Reviewing the literature lets one see what came before, and what did and didn't work for other researchers.
- To demonstrate one's understanding, and ability to critically evaluate research in the field.
- To provide evidence that may be used to support your own findings.

1. Speech based human emotion recognition using MFCC “by M. S. Likitha, S. R. R. Gupta, K. Hasitha and A. U. Raju, International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, 2017.

- Facial expression-based emotion classification enhances the fluency, accuracy, and genuineness of an interaction. This type of classification approach is very useful in interpreting human-computer interaction.
- Feature modeling and classification accuracy are strongly related to each other.

Advantages

• Detecting faces based on the relationships between neighboring regions. • For videos, investigation and combination are done by both spatial and temporal features into a unified model.

Disadvantage

• Some features would be taken more than once and some would be negligible

2. Speaker Verification Using MFCC and Support Vector Machine” by Chen, S.H. and Y.R. Luo in Proceedings of the International Multiconference of Engineers and Computer Scientists, March 18 - 20, 2009.

- The speech signal was first stored as a 10000-sample vector. Based on observation, the actual uttered speech came up to about 2500 samples.
- The feature extraction is done by using MFCC (Mel Frequency Cepstral Coefficients) and Vector Quantization. The accuracy of VQ technique is seen to be higher than the MFCC approach and hence, better identification is carried out using the VQ technique.

Advantages

- It uses two approaches, if Single user approach fails, it transfers to multiple user approach card

Disadvantages

- because of the prominence given to energy, this approach sometimes failed to recognize the same word uttered with different energy. The accuracy of Vector Quantization ranges from 90.2%-98% and may be lesser when tested on a larger database

3. "The Ryerson Audiovisual Database of Emotional Speech and Song (RAVDESS)", by Livingstone SR, Russo FA in PLOS ONE 2018.

• In this application, we described the construction and validation of the RAVDESS, a set of emotional expressions that are dynamic and multimodal.

• Validation of the RAVDESS was performed with 247 raters from North America. Validity referred to the accuracy with which participants correctly identified the actors' intended emotions.

Advantages

• Participant testing involving untrained research participants revealed high rates of emotional validity and test-retest reliability.

Disadvantages

• Data storage is requirement is high.

4. An Introduction to Sampling Theory." by Thomas Zawistowski & Paras Shah

• The signals we use in the real world, such as our voices, are called "analog" signals. To process these signals in computers, we need to convert the signals to "digital" form.

• When the signal is converted back into a continuous time signal, it will exhibit a phenomenon called aliasing.

Advantages

- It is easy to see the effects of changing the sampling frequency by looking at these transform plots.

Disadvantages

- sampling frequency do not improve the quality of the reconstructed signal.
- higher sampling frequencies require faster converters and more storage.

5. “Fraud Detection Emotion Recognition Using Hybrid Gaussian Mixture Model and Deep Neural Network by ISMAIL SHAHIN, ALI BOU NASSIF, AND SHIBANI HAMSA Department of
Electrical and Computer Engineering, University of Sharjah, Sharjah 27272, United Arab Emirates

- The project aims at recognizing emotions for a text-independent and speaker-independent emotion recognition system based on a novel classifier, which is a hybrid of a cascaded Gaussian mixture model and deep neural network (GMM-DNN).. These results demonstrate that the hybrid classifier significantly gives higher emotion recognition accuracy than SVMs and MLP classifiers. Also, the performance of the classifier has been tested using two distinct emotional databases and in normal and noisy talking conditions.

Advantages

- Neural networks help to group unlabeled data according to similarities among the samples. In some situations, in order to get a more precise classification, the features extracted by neural networks may be processed by other algorithms or vice versa.

Disadvantages

- The performance of ‘‘GMM-DNN, SVM, and MLP’’ classifiers in the recognition of the disgust emotion is not as good as in the recognition of other emotions

2.2 Methodologies

The speech emotion detection system is implemented as a Machine Learning (ML) model. The steps of implementation are comparable to any other ML project, with additional fine-tuning procedures to make the model function better. The flowchart represents a pictorial overview of the process . The first step is data collection, which is of prime importance. The model being developed will learn from the data provided to it and all the decisions and results that a developed model will produce is guided by the data. The second step, called feature engineering, is a collection of several machine learning tasks that are executed over the collected data. These procedures address the several data representation and data quality issues. The third step is often considered the core of an ML project where an algorithmic based model is developed. This model uses an ML algorithm to learn about the data and train itself to respond to any new data it is exposed to. The final step is to evaluate the functioning of the built model.

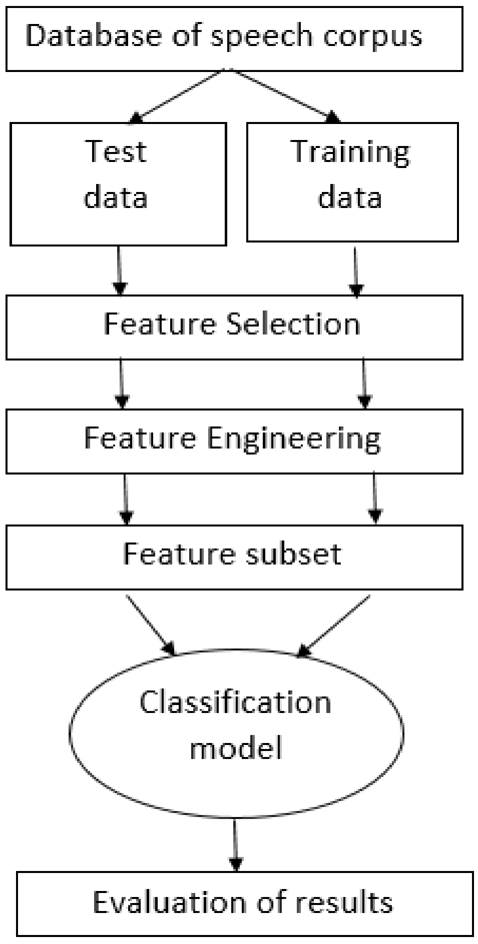


Figure 2.1 System Model

2.3 Summary

Very often, developers repeat the steps of developing a model and evaluating it to compare the performance of different algorithms. Comparison results help to choose the appropriate ML algorithm most relevant to the problem.

The two algorithm we have implemented in this project are MLP and Neural Network so we have taken the better algorithm which gives us more accuracy and have implemented the same for our live audio testing

Chapter 3

SYSTEM REQUIREMENTS

3.1 Introduction

Requirement analysis is an essential part of product development. It plays an important role in determining the feasibility of an application. Requirement analysis defines the software and hardware necessities that are required to develop the product or application. Requirement analysis mainly consists of software requirements, hardware requirements and functional requirements.

Software requirements: This mainly refers to the needs that solve the end user problems using the software. The activities involved in determining the software requirements are:

1. **Elicitation:** This refers to gathering of information from the end users and customers.
2. **Analysis:** Analysis refers to logically understanding the customer needs and reaching a more precise understanding of the customer requirements.
3. **Specification:** Specification involves storing the requirements in the form of use cases, user stories, functional requirements and visual analysis.
4. **Validation:** Validation involves verifying the requirements that has been specifies.
5. **Management:** During the course of development, requirements constantly change and these requirements have to be tested and updated accordingly.

Hardware requirements: Hardware requirements are the physical aspects that are needed for an application. All the software must be integrated with hardware in order to complete the development process. The hardware requirements that are taken into account are:

1. Processor Cores and Threads
2. GPU Processing Power
3. Memory
4. Secondary Storage
5. Network Connectivity

3.2 Software and Hardware Requirements

SOFTWARE REQUIREMENTS: Listed below are the software requirements for performing time series analysis on the fraud data:

1. **Operating System:** Operating system acts as the interface between the user programs and the kernel. Windows 8 and above (64 bit) operating system is required.
2. **Anaconda**: Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. Package versions are managed by the package management system conda.
3. **Jupyter Notebook**: The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.
4. **Data Set**: The dataset contains 1440 sound files of speech utterances in six basic emotional categories, namely: Happy, Sad, Angry, Calm, Fear and Neutral. It is an acted recording, where actors from two age groups of Old (64-year-old) and Young (26-year-old) had performed the dictation

HARDWARE REQUIREMENTS

1. **Processor:** Intel i5 2.5 Ghz upto 3.5Ghz (or AMD equivalent)
2. **GPU (preferred)**: dedicated GPU from NVIDIA or AMD with 4GB VRAM
3. **Memory**: minimum 8GB RAM
4. **Secondary Storage**: minimum 128GB SSD or HDD
5. **Network Connectivity**: bandwidth ~ 10 Mbps – 75 Mbps

3.2.1 Functional and Non-Functional Requirements

FUNCTIONAL REQUIREMENTS: Functional requirements are those which represent the functions of the system. Functional requirements may involve calculations, data manipulation, technical details and data processing. The functional requirements are:

1. **Data Pre-processing**: This process involves cleaning, transforming and reducing data to convert raw data in a useful manner.
2. **Training**: Initially, the system has to train based on the data set given. The training period is when the system learns how to perform the required task based on the inputs given through the dataset.
3. **Forecasting**: Forecasting is the process of making predictions of the future based on past and present data and most commonly by analysis of trends.
4. **Evaluation**: In order to know whether the system is working efficiently, the song that has been generated is again passed into the classifier to determine the kind of song that it has generated. This is necessary in order to determine the accuracy of the system.

NON-FUNCTIONAL REQUIREMENTS: Non-functional requirements are the specifics that can be used to measure the operation of the system. These include:

1. **Accuracy**: This refers to the number of correct outputs to the total number of outputs.
2. **Openness**: The system must be guaranteed to work efficiently for a certain period of time.
3. **Portability**: The system must be designed in a way which is independent of the platform on which it is run. This makes it possible to run on many systems without making a lot of changes.
4. **Reliability**: The system has to produce fast and accurate results.

3.3 Summary

Requirement analysis is an essential part of product development. It plays an important role in determining the feasibility of an application. By analyzing the different requirements for a system, a concrete plan of action can be designed. The software and hardware requirements can limit the system if not carefully listed down. They have to be compatible with each other to complete integration of the system to deliver the final product. These requirements are a quantitative measure of the system. The user demand is met by breaking down the high-level tasks to requirements, which help in the task of system designing providing clear goals.

The functional and non-functional requirements help measure the system operations. The functional requirements describe the operations that a system has to perform. These can include tasks such as pre-processing, data extraction and evaluation. The non-functional requirements perform the task of measuring of how well the system executes these operations. It assesses the system based on how reliable, accurate and user friendly it is.

Requirement analysis is hence an important task before the start of any project. It allows the developer to find out the feasibility level and complexity of the project. It further helps in building an execution plan for the project to proceed along.

Chapter 4

SYSTEM DESIGN

4.1 Introduction

System Design is the way towards portraying the components of a framework, for instance, interfaces, algorithms, UML diagrams and information sources or databases utilized for a framework to fulfill determined prerequisites. It is characterized so as to fulfill the necessities and prerequisites of a business or association through the building of a lucid and well-running framework.

Frauds are known to be dynamic and have no patterns, hence they are not easy to identify. Fraudsters use recent technological advancements to their advantage. They somehow bypass security checks, leading to the loss of millions of dollars. Analyzing and detecting unusual activities using data mining techniques is one way of tracing fraudulent transactions. More and more companies are looking to invest in machine learning to improve their services. Machine learning is a combination of various computer algorithms and statistical modeling to allow the computer to perform tasks without hard coding. The acquired model would be learning from the "training data". Predictions can be made or actions can be performed from stored experiential knowledge.

4.2 Proposed System

The proposed system consists 3 stages – speech signal acquisition, emotion pattern classification and the calculation of affection index with mobile agent. A query speech signal is first picked up by the single microphone of the cellular phone and then transmitted to the emotion recognition server. Then the queried speech is classified with confidence with probability and the classification result will be reported and transmitted to the mobile agent, based on the probability of emotion classification.

Raw speech data set is randomly divided into two sets of data set – training set and test set. At the training phase, feature set is extracted from pre-processed speeches in training set At the testing phase, a test input speech is picked up by a cellular phone and preprocessed as before, and the same indexed feature set from the SFS optimization stage is extracted.

Each signal from both the sets is pre-processed to make it suitable for data gathering and analysis. The test signal is tested with every model in order to classify and detect its emotion.

1. Sampling

Sampling is the first and important step of signal processing. Signals which we used normally, are all analog signals i.e., continuous time signals. Therefore, for processing purpose in computer, discrete signals are better. In order to convert these continuous time signals to discrete time signals, sampling is used.

2. Pre-emphasis

The input signal often has certain low frequency components which will result in samples similar to their adjacent samples. These parts represent a slow variation with time and hence are not of any importance while extracting signal information. Therefore, we are performing pre-emphasizing by applying a high pass filter on the signal in order to emphasize the high frequency components which represent the rapidly changing signal.

3. De-silencing

Audio signals often contain regions of absolute silence occurring at the beginning or at the end and sometimes in between higher frequencies. It is required to remove this unwanted part from the signal and hence de-silencing is performed.

4. Framing

For the purpose of analysis, observable stationary signal is preferable. If we observe the speech signal on a short time basis, we are able to get a stationary signal. We divide the input signal into small constituent frames of a specific time interval.

5. Windowing

Most of the digital signals are large and infinite that they cannot be analyzed entirely at the same time. For better statistical calculations and processing, values of signal at each point should be available. In order to convert large digital signal into a small set for processing and analyzing, and for smoothing ends of signal, windowing is performed

4.3 Data Flow Diagram

4.3.1 Description

A data flow (DFD) is a graphical representation of the "flow" of data through information through information system, modeling its prospects. DFDs are also used for the visualization of data processing. A DFD shows that what kind of information will be input to the system and output from the system, where the data will come from and go to, and where the data will be stored. It does not show about the timing of the processors, or information about whether processes will operate in sequence or in parallel. Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation. Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

4.3.2 Uses of DFD's

- Data flow diagram is useful for establishing the boundary of the business or system
- domain (the sphere of analysis activity) under investigation.
- It identifies external entities along with their data interfaces that interact with the processes of interest.
- A DFD can be useful tool for helping secure stake holder agreement (sign–off) on the project scope.
- It is also a useful tool for breaking down a process into its sub processes for closer analysis.
- It helps in the logical information flow of the system.
- It determines of physical system construction requirements.
- Simplicity of notation.
- It establishes the manual and automated systems requirements.

4.3.3 Levels of Abstraction

In order to fashion an accurate picture of anything, which need to employ various levels of abstraction. This need arises from the basic relationship between an object and an observer, or a subject and its student. There are qualitative changes, as one move from one level of abstraction to the next, and seemingly a progression of from and/or formative principle.
In Software engineering DFD (data flow diagram) can be drawn to represent the system of different levels of abstraction. Higher level DFDs are partitioned into low levels-hacking more information and functional elements. Levels in DFD are numbered 0, 1, 2 or beyond. Here, we will see mainly 2 levels in data flow diagram, which are: 0-level DFD and 1-level DFD

Level 0:

Level 0 is also called a Context Diagram. It is a basic overview of the whole system or process being analyzed or modeled. It is designed to be an at-a-glance view showing the system as a single high-level process with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

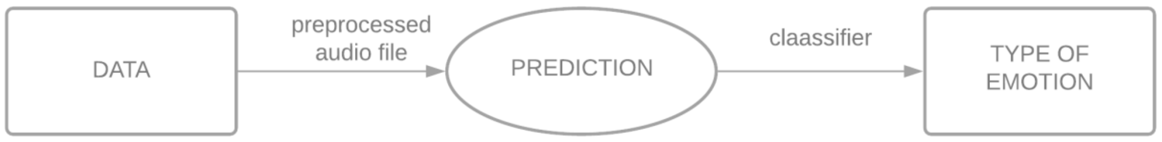


Figure 4.1 Level 0 Dataflow Diagram

Fig 4.1 describes the overall process of the project. We input the audio file data and preprocess the audio file before prediction using different classifier to predict the type of emotion

Level 1:

This context-level data flow diagram (DFD) is next explored, to produce a level 1data flow diagram (DFD) that shows some of the detail of the system being modeled. The level 1 data flow diagram (DFD) shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provides all functionality of the system as a whole. It also identifies the internal data stores that must be presenting order for the system to do its job, and flow of data between the various inputs of the system.

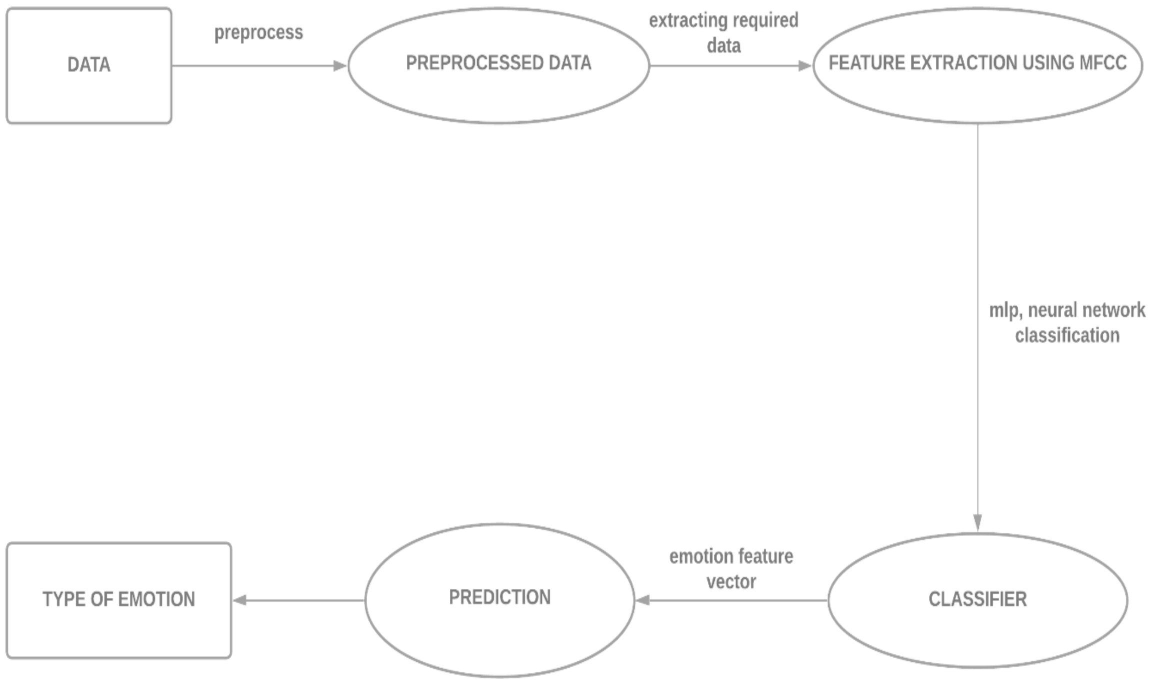


Figure 4.2 Level 1 Dataflow Diagram

Fig 4.2 describes the preprocessed audio file is being extracted where features like MFCC ,Pitch and other important features which determines a audio and using these features we can classify the audio file by using classifiers like MLP and Neural Network which predict the type of emotion

4.3 SUMMARY

This chapter gives a brief introduction to the system design process, its methodologies requires for developing a system. It deals with different types of design processes used in real world and system architectures. Proposed model mainly describes the how exactly the system works. Data flow diagram for the proposed model is designed in three different levels of abstraction. Dataflow diagram (DFD's) offers a graphical representation for summarizing the movement of data flows in the different levels of processes. It is mainly discussed about what is the proposed system that is implementing with that of the existing system. It describes the how the complexity is reduced, reduced in the cost and about the performance of the system.

Chapter 5

IMPLEMENTATION

5.1 Introduction

Implementation is the process of transforming a mental plan in familiar terms into one compatible with the computer. It is the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy. Computer programming is a key element in the process of implementation which is the process of designing, writing, testing, debugging / troubleshooting, and maintaining the source code of computer programs. The source code is written in a programming language. The purpose of programming is to create a program that exhibits a certain desired behavior. The process of writing source code often requires expertise in many different subjects, including knowledge of the application domain, specialized algorithms and formal logic. Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements.

Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. The architectural design of a system emphasizes the design of the system architecture that describes the structure, behavior and more views of that system and analysis. The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. It is often conducted via modelling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems, designs are included. The physical design relates to the actual input and output processes of the system. It is explained in terms of how data is input into a system, how it is verified or authenticated, how it is processed, and how it is displayed.

5.2 System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. System design is used to solve internal problems, improve efficiency and expand opportunities

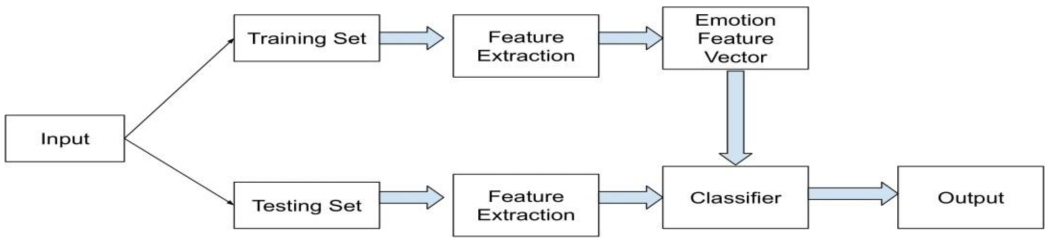


Figure 5.1 System design.

The flowchart represents a pictorial overview of the process.

- The first step is data collection, which is of prime importance. The model being developed will learn from the data provided to it and all the decisions and results that a developed model will produce is guided by the data.
- The second step, called feature engineering, is a collection of several machine learning tasks that are executed over the collected data. These procedures address the several data representation and data quality issues.
- The third step is often considered the core of an ML project where an algorithmic based model is developed. This model uses an ML algorithm to learn about the data and train itself to respond to any new data it is exposed to.
- The final step is to evaluate the functioningof the built model.

.

5.2.1 Methodology

Language used: Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It’s high-level built-in data structures combined with dynamic typing and dynamic binding make it very attractive for Rapid Application Development as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

GUI used: Jupyter Notebook

Jupyter is a project and community whose goal is to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages**.** Jupyter Notebook is an open-source web application that allows data scientists to create and share documents that integrate live code, equations, computational output, visualizations, and other multimedia resources, along with explanatory text in a single document.

5.3 Algorithm

Multilayer Perceptron

A multilayer perceptron (MLP) is a class of feedforward artificial neural network (ANN). The term MLP is used ambiguously, sometimes loosely to any feedforward ANN, sometimes strictly to refer to networks composed of multiple layers of perceptrons (with threshold activation); see § Terminology. Multilayer perceptrons are sometimes colloquially referred to as "vanilla" neural networks, especially when they have a single hidden layer.

An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

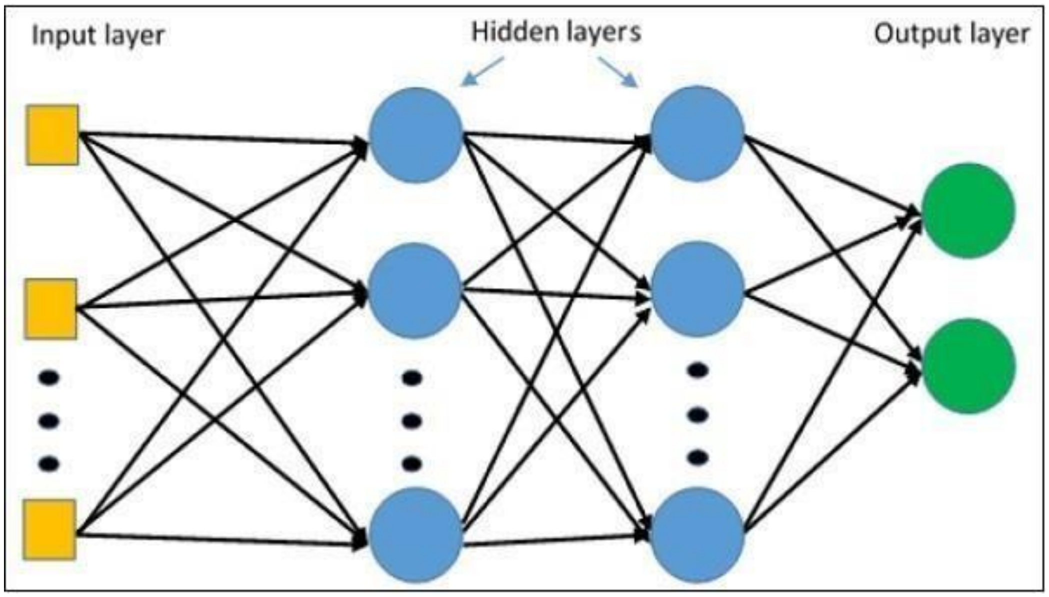


Figure 5.2 MLP classifier layers

Training the Model

There are basically three steps in the training of the model.

- Forward pass
- Calculate error or loss
- Backward pass

1. **Forward pass**

In this step of training the model, we just pass the input to model and multiply with weightsand add bias at every layer and find the calculated output of the model.

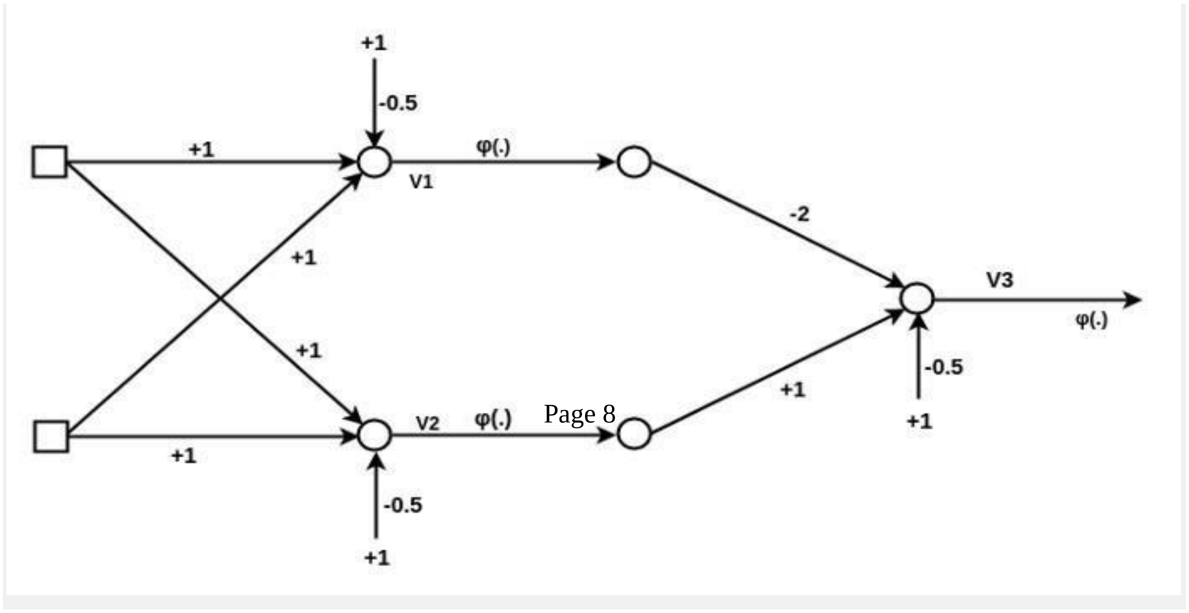


Figure 5.3 Forward pass diagram

2. **Loss Calculate**

When we pass the data instance (or one example) we will get some output from the model that is called Predicted output(pred_out) and we have the label with the data that is real output or expected output (Expect_out). Based upon these both we calculate the loss that we have to backpropagate (using Backpropagation algorithm). There is various Loss Function that we use based on our output and requirement.

3. **Backward Pass**

After calculating the loss, we backpropagate the loss and updates the weights of the model by using gradient. This is the main step in the training of the model. In this step, weights will adjust according to the gradient flow in that direction.

Neural Networks

Artificial neural networks (ANNs) as “Biologically inspired computing code with the number of simple, highly interconnected processing elements for simulating (only an attempt) human brain working & to process information model”. It’s way different than computer program though. There are several kinds of Neural Networks in deep learning. Neural networks consist of input and output layers and at least one hidden layer.

- Multi-Layer Perceptron
- Radial Basis Network
- Recurrent Neural Networks
- Generative Adversarial Networks
- Convolutional Neural Networks.

Neural Network Algorithms are based on radial basis function with can be used for strategic reasons. There are several other models of the neural network including what we have mentioned above. For an introduction to the neural network and their working model continue reading this post. You will get a sense of how they work and used for real mathematical

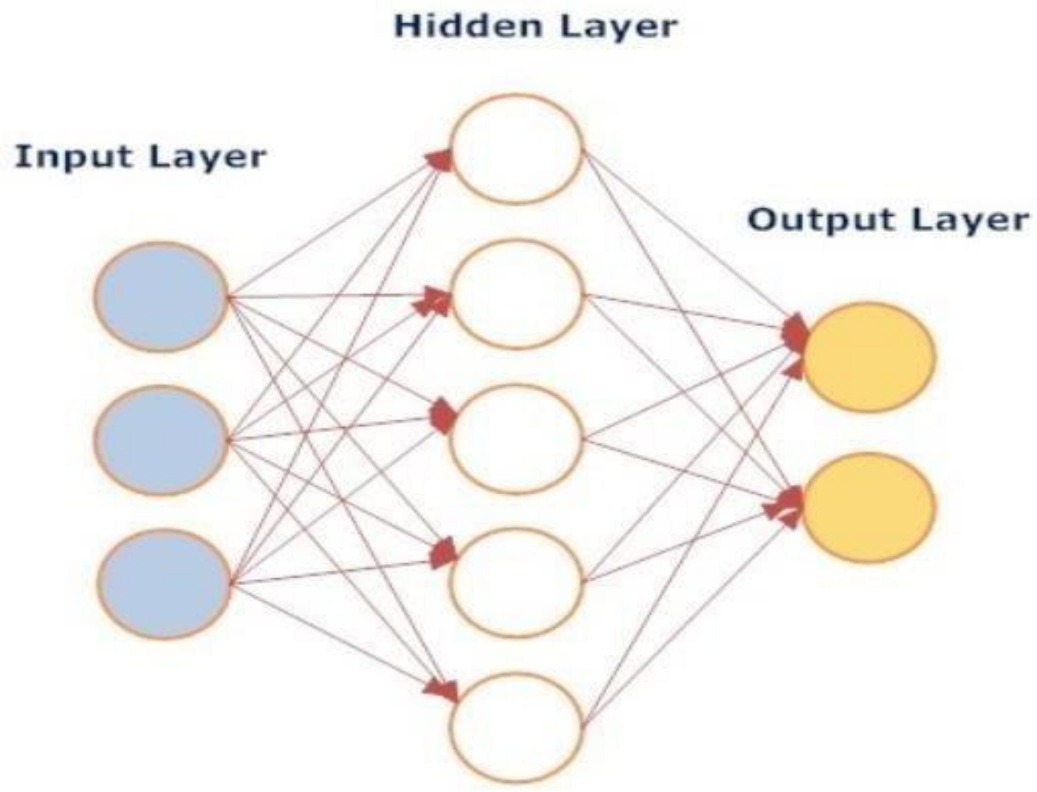


Figure 5.4 Neural Network Classifier

5.4 Architectural Components

- **Input Layers, Neurons, and Weights** – The basic unit in a neural network is called as the neuron or node. These units receive input from the external source or
- some other nodes. The idea here is to compute an output based associated weight. Weights to the neuron are assigned based on its relative importance compared with other inputs. Now finally function is applied to this for computations.
- Let's assume our task to it to make tea so our ingredients will represent the "neurons" or input neurons as these are building blocks or starting points. The amount of each ingredient is called a "weight." After dumping tea, sugar, species, milk and water in a pan and then mixing will transform it another state and color. This process of transformation can be called an "activation function".
- **Hidden Layers and Output Layers** – The hidden layer is always isolated from the external world hence it's called as hidden. The main job of the hidden layer to take inputs from the input layer and perform its job i.e., calculation and transform the result to output nodes. Bunch of hidden nodes can be called a hidden layer.
- Continuing the same example above – In our tea making task, now using the mixture of our ingredients coming out of the input layer, the solution upon heating (computation process) starts changing color. The layers made up by the intermediate products are called "hidden layers". Heating can be compared with the activation process at the end we get our final tea as output.

The network described here is much simpler for ease of understanding compared to the one you will find in real life. All computations in the forward propagation step and backpropagation step are done in the same way (at each node) as discussed before. Neural Network Algorithms

.

5.5 Feature Extraction

THE PROCESS

Speech is a varying sound signal. Humans are capable of making modifications to the sound signal using their vocal tract, tongue, and teeth to pronounce the phoneme. The features are a way to quantify data. A better representation of the speech signals to get the most information from the speech is through extracting features common among speech signals. Some characteristics of good features include:

- The features should be independent of each other. Most features in the feature vector are correlated to each other. Therefore, it is crucial to select a subset of features that are individual and independent of each other.
- The features should be informative to the context. Only those features that are more descriptive about the emotional content are to be selected for further analysis.
- The features should be consistent across all data samples. Features that are unique and specific to certain data samples should be avoided.
- The values of the features should be processed. The initial feature selection process can result in a raw feature vector that is unmanageable. The process of Feature Engineering will remove any outliers, missing values, and null values.

SPEECH EMOTION DETECTION.

The features in a speech percept that is relevant to the emotional content can be grouped into two main categories:

1. Prosodic features
2. Phonetic features.

The prosodic features are the energy, pitch, tempo, loudness, formant, and intensity. The phonetic features are mostly related to the pronunciation of the words based on the language. Therefore, for the purpose of emotion detection, the analysis is performed on the prosodic features or a combination of them. Mostly the pitch and loudness are the features that are very relevant to the emotional content.

5.6 Packages/Libraries Used:

Keras

Keras is a powerful and easy-to-use free open-source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code.

• Simple - but not simplistic. Keras reduces developer cognitive load to free you to focus on the parts of the problem that really matter.

• Flexible - Keras adopts the principle of progressive disclosure of complexity: simple workflows should be quick and easy, while arbitrarily advanced workflows should be possible via a clear path that builds upon what you've already learned.

• Powerful -- Keras provides industry-strength performance and scalability: it is used by organizations and companies including NASA, YouTube, or Waymo.

Tensoflow

TensorFlow is an open-source framework developed by Google researchers to run machine learning, deep learning and other statistical and predictive analytics workloads.

The framework includes sets of both high-level and low-level APIs.

1. Voice/Sound Recognition

One of the most well-known uses of TensorFlow are Sound based applications. With the proper data feed, neural networks are capable of understanding audio signals. These can be:

• Voice recognition – mostly used in IoT, Automotive, Security and UX/UI
• Voice search – mostly used in Telecoms, Handset Manufacturers
• Sentiment Analysis – mostly used in CRM
• Flaw Detection (engine noise) – mostly used in Automotive and Aviation

Librosa

Librosa is a Python package for music and audio analysis. Librosa is basically used when we work with audio data like in music generation (using LSTM's), Automatic Speech Recognition. It provides the building blocks necessary to create the music information.

Soundfile

SoundFile is an audio library based on libsndfile, CFFI and NumPy. File reading/writing is supported through libsndfile, which is a free, cross-platform, open-source (LGPL) library for reading and writing many different sampled sound file formats that runs on many platforms including Windows, OS X, and Unix

Data can be written to the file using soundfile.write(), or read from the file using soundfile.read(). SoundFile can open all file formats that libsndfile supports, for example WAV, FLAC, OGG and MAT files (see Known Issues below about writing OGG files).

Here is an example for a program that reads a wave file and copies it into an FLAC file:

```
import soundfile as sf

data, samplerate = sf.read('existing file.wav')

sf.write('new file.flac', data, samplerate)
```

NumPy

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices.

Pandas

Pandas Data Frame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). Indexing and Selecting Data. Working with Missing Data. Iterating over rows and columns

Seaborn

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. For a brief introduction to the ideas behind the library.

Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

5.7 Summary

Implementation is the stage of the project when the theoretical design is turned out into a working system. The implementation stage involves careful planning, investigation of the existing system and the constraints on implementation, designing of methods to achieve change over and evaluation of changeover methods.

It also seems that the Neural Network Algorithm performs better than MPL algorithm during the implementation and so we go ahead with Neural Network for our live audio testing where a real time audio will be inputted using a microphone and the type of emotion will be predicted and will be 80% accurate

.

CHAPTER 6

SYSTEM TESTING

6.1 Introduction

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. A primary purpose of testing is to detect software failures so that defects may be discovered and corrected. Testing cannot establish that a product functions properly under all conditions, but only that it does not function properly under specific conditions. The scope of software testing often includes the examination of code as well as the execution of that code in various environments and conditions as well as examining the aspects of code: does it do what it is supposed to do and do what it needs to do. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.

6.1.1 Testing Approaches

Static, dynamic and passive testing

There are many approaches available in software testing. Reviews, walkthroughs, or inspections are referred to as static testing, whereas executing programmed code with a given set of test cases is referred to as dynamic testing. Static testing is often implicit, like proofreading, plus when programming tools/text editors check source code structure or compilers (pre-compilers) check syntax and data flow as static program analysis. Dynamic testing takes place when the program itself is run. Dynamic testing may begin before the program is 100% complete in order to test particular sections of code and are applied to discrete functions or modules.

Typical techniques for these are either using stubs/drivers or execution from a debugger environment. Static testing involves verification, whereas dynamic testing also involves validation. Passive testing means verifying the system behaviour without any interaction with the software product. Contrary to active testing, testers do not provide any test data but look at system logs and traces. They mine for patterns and specific behaviour in order to make some kind of decisions. This is related to offline runtime verification and log analysis.

Exploratory approach

Exploratory testing is an approach to software testing that is concisely described as simultaneous learning, test design and test execution. CemKaner, who coined the term in 1984, defines exploratory testing as "a style of software testing that emphasizes the personal freedom and responsibility of the individual tester to continually optimize the quality of his/her work by treating test-related learning, test design, test execution, and test result interpretation as mutually supportive activities that run in parallel throughout the project.

The "box" approach

Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that the tester takes when designing test cases. A hybrid approach called grey-box testing may also be applied to software testing methodology. With the concept of grey-box testing—which develops tests from specific design elements—gaining prominence, this "arbitrary distinction" between black- and white-box testing has faded somewhat.

White-box testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) verifies the internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing, an internal perspective of the system (the source code), as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

Black-box testing

Black-box testing (also known as functional testing) treats the software as a "black box," examining functionality without any knowledge of internal implementation, without seeing the source code. The testers are only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, allpairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing, and specification-based testing. Specification-based testing aims to test the functionality of software according to the applicable requirements. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behavior), either "is" or "is not" the same as the expected value specified in the test case. Test cases are built around specifications and requirements, i.e., what the application is supposed to do. It uses external descriptions of the software, including specifications, requirements, and designs to derive test cases. These tests can be functional or non-functional, though usually functional.

Component interface testing

Component interface testing is a variation of black-box testing, with the focus on the data values beyond just the related actions of a subsystem component. The practice of component interface testing can be used to check the handling of data passed between various units, or subsystem components, beyond full integration testing between those units. The data being passed can be considered as "message packets" and the range or data types can be checked, for data generated from one unit, and tested for validity before being passed into another unit. One option for interface testing is to keep a separate log file of data items being passed, often with a timestamp logged to allow analysis of thousands of cases of data passed between units for days or weeks. Tests can include checking the handling of some extreme data values while other interface variables are passed as normal values. Unusual data values in an interface can help explain unexpected performance in the next unit.

Visual testing

The aim of visual testing is to provide developers with the ability to examine what was happening at the point of software failure by presenting the data in such a way that the developer can easily find the information she or he requires, and the information is expressed clearly. At the core of visual testing is the idea that showing someone a problem (or a test failure), rather than just describing it, greatly increases clarity and understanding. Visual testing, therefore, requires the recording of the entire test process – capturing everything that occurs on the test system in video format. Output videos are supplemented by real-time tester input via picture-in-a-picture webcam and audio commentary from microphones.

6.1.2 Testing Levels

Broadly speaking, there are at least three levels of testing: unit testing, integration testing, and system testing.

Unit testing

Unit testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors. These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to ensure that the building blocks of the software work independently from each other. Unit testing is a software development process that involves a synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development life cycle. Unit testing aims to eliminate construction errors before code is promoted to additional testing; this strategy is intended to increase the quality of the resulting software as well as the efficiency of the overall development process.

Integration testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed. Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system. Integration tests usually involve a lot of code, and produce traces that are larger than those produced by unit tests. This has an impact on the ease of localizing the fault when an integration test fails. To overcome this issue, it has been proposed to automatically cut the large tests in smaller pieces to improve fault localization.

System testing

System testing tests a completely integrated system to verify that the system meets its requirements. For example, a system test might involve testing a login interface, then creating and editing an entry, plus sending or printing results, followed by summary processing or deletion (or archiving) of entries, then logoff.

6.2 Test cases

A test case is a documentation which specifies input values, expected output and the preconditions for executing the test. Test case document is also a part of test deliverables, by reading test case document stakeholders get an idea about the quality of test cases written and the effectiveness of those test cases. Stakeholders can also provide inputs about the current set of test cases as well as suggest some more missing test cases.

A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. Test cases for software helps to guide the tester through a sequence of steps to validate whether a software application is free of bugs and working as required by the end user. The expected result tells the tester what they should experience as a result of the test performed. In this way we can determine if the test case is a “pass” or “fail”.

Test case 1:

TEST CASE	UNIT TESTING
TEST NAME	PREDICTION
ITEMS BEING TESTED	PREDICT
SAMPLE INPUT	ACCURATE EMOTIONS(AUDIO)
EXPECTED OUTPUT	EMOTION RECOGNITION IS DONE BASED ON THE INPUT FILES.
ACTUAL OUTPUT	SUCCESSFUL SPEECH EMOTION RECOGNITION
REMARKS	PASS

Figure 6.1 Unit Testing Table

Test case 2:

TEST CASE	**INTEGRATION TESTING**
TEST NAME	PREDICTION
ITEMS BEING TESTED	PREDICT
SAMPLE INPUT	INCORRECT AUDIO FILES
EXPECTED OUTPUT	INACCURATE EMOTIONS DISPLAYED ON INVALID AUDIO FILES
ACTUAL OUTPUT	EMOTION RECOGNITION UNSUCCESSFULL
REMARKS	PASS

Figure 6.2 Integration Testing Table

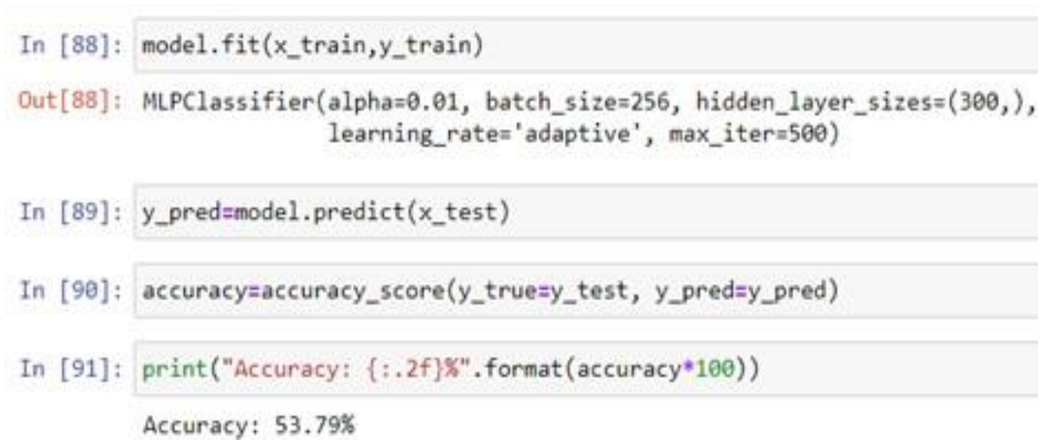
Test case 3:

TEST CASE	**SYSTEM TESTING**
TEST NAME	SYSTEM TESTING IN DIFFERENT OS
SAMPLE INPUT	EXECUTE THE PROGRAM IN WINDOWS AND macOS
EXPECTED OUTPUT	EQUALL PERFORMANCE IN BOTH WINDOWS AND macOS
ACTUAL OUTPUT	SAME AS EXPECTED OUTPUT
REMARKS	PASS

Figure 6.3 System Testing Table

6.3 Results

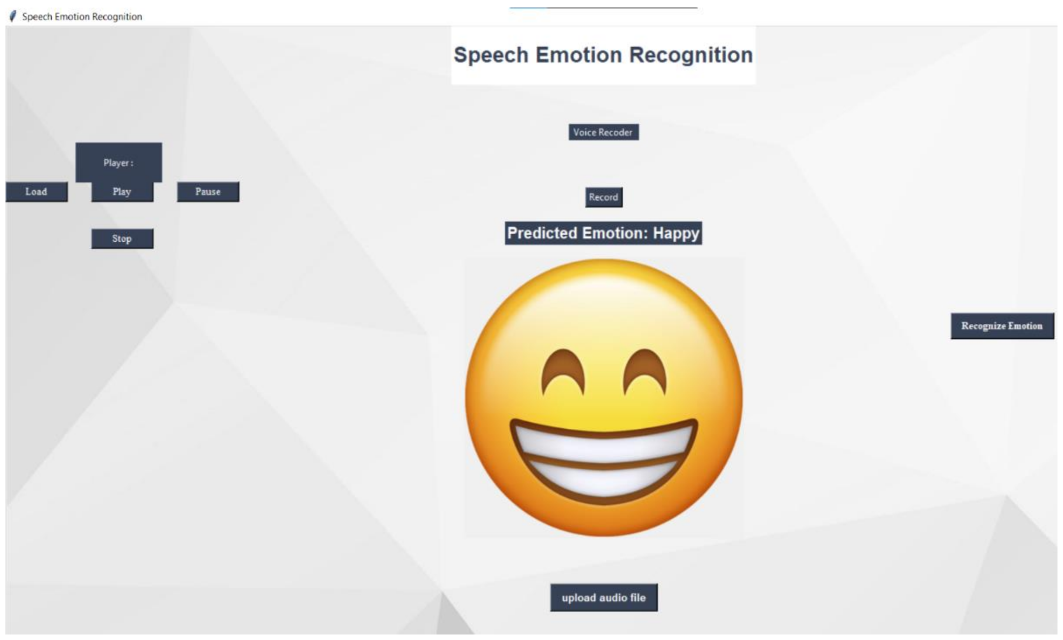
A result is the final consequence of actions or events expressed qualitatively or quantitatively. Performance analysis is an operational analysis, is a set of basic quantitative relationship between the performance quantities.



```
In [88]: model.fit(x_train,y_train)

Out[88]: MLPClassifier(alpha=0.01, batch_size=256, hidden_layer_sizes=(300,),
                       learning_rate='adaptive', max_iter=500)

In [89]: y_pred=model.predict(x_test)

In [90]: accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)

In [91]: print("Accuracy: {:.2f}%".format(accuracy*100))

         Accuracy: 53.79%
```

Figure 6.4 MLP Accuracy Score

MLP Classifier Algorithm had a accuracy score of 54% when emotions were predicted on the test dataset using the train dataset for training the Model . (Refer Fig 6.5)



,

Figure 6.6 Happy Emotion Prediction

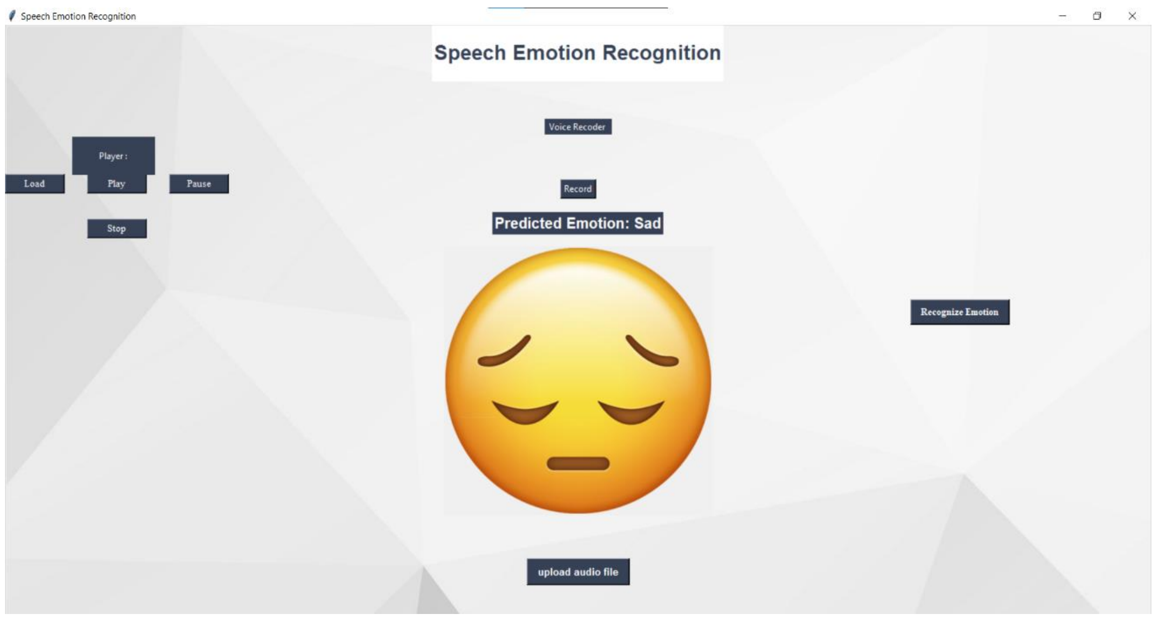


Figure 6.7 Sad Emotion Prediction

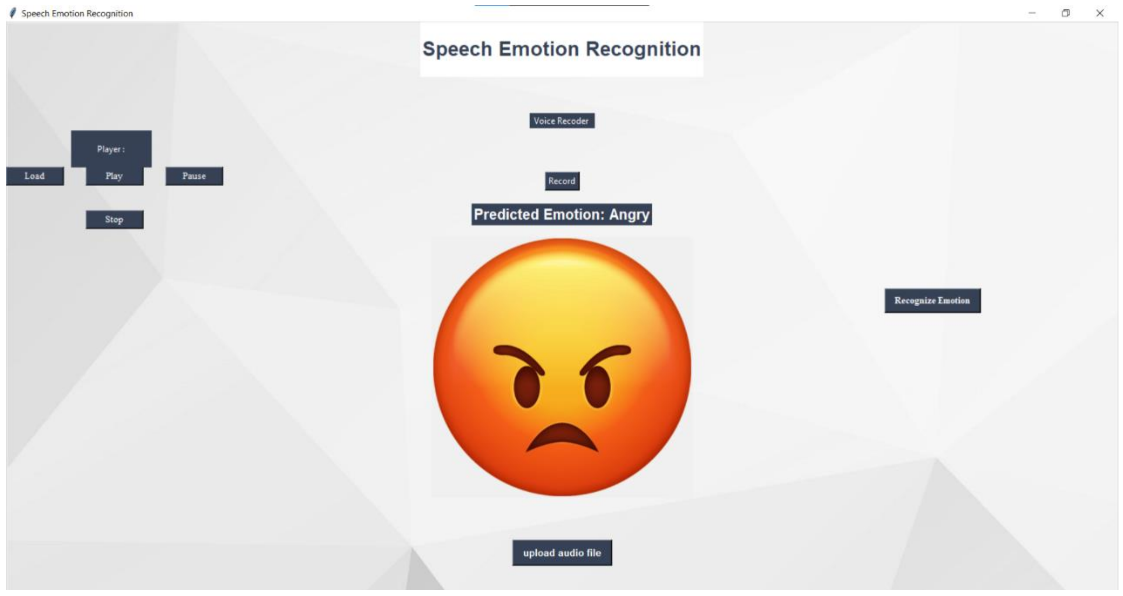


Figure 6.8 Angry Emotion Prediction

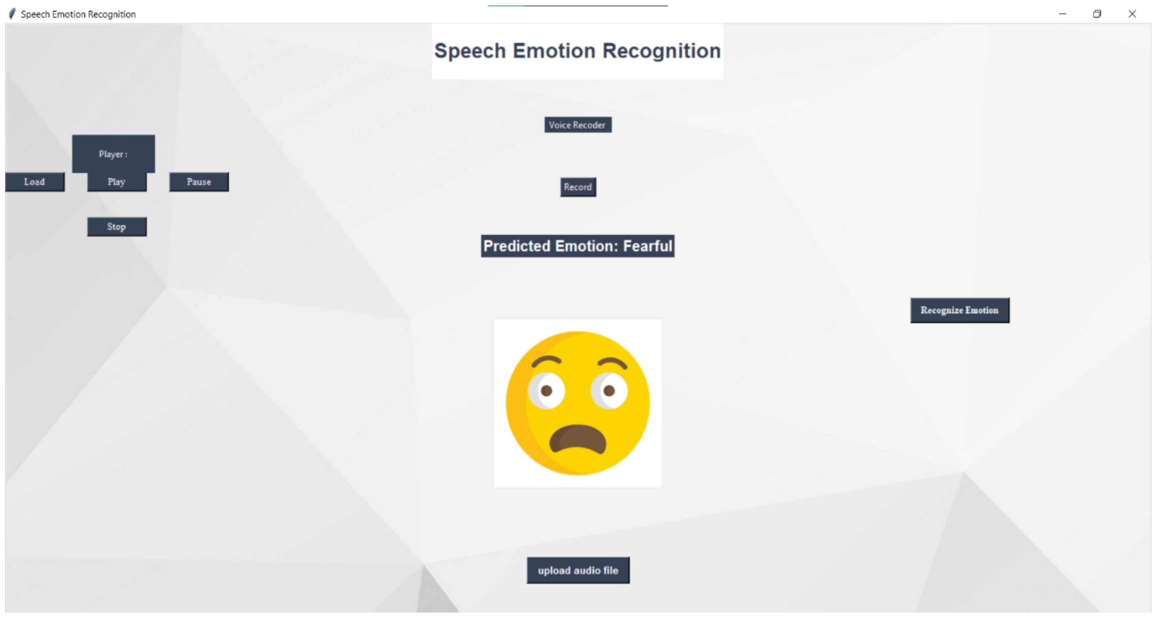

Speech Emotion Recognition
Speech Emotion Recognition
Voice Recoder
Player :
Load
Play
Pause
Stop
Record
Predicted Emotion: Fearful
Recognize Emotion
upload audio file

Figure 6.9 Fearful Emotion Prediction

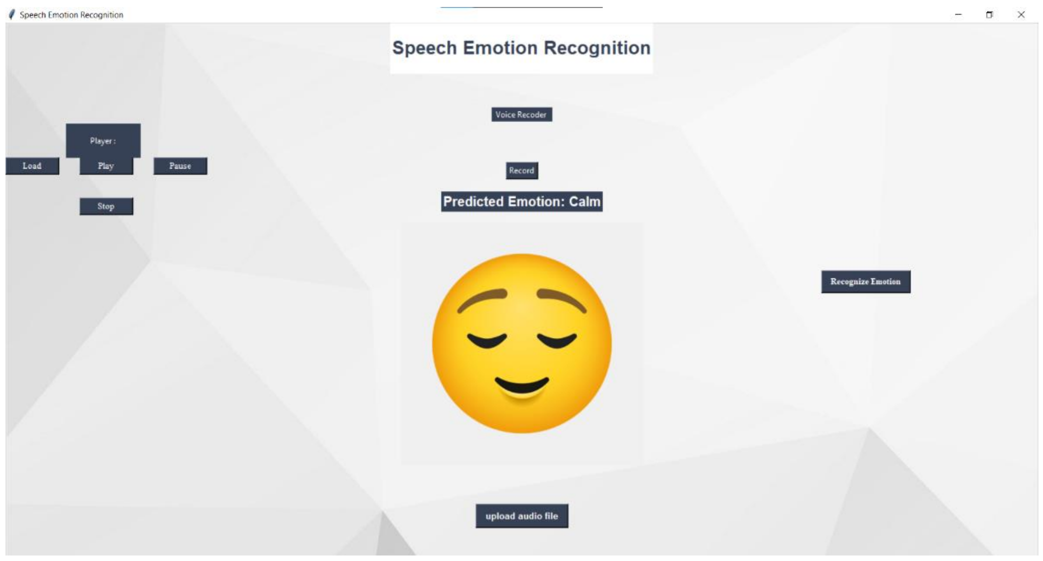


Figure 6.10 Calm Emotion Prediction

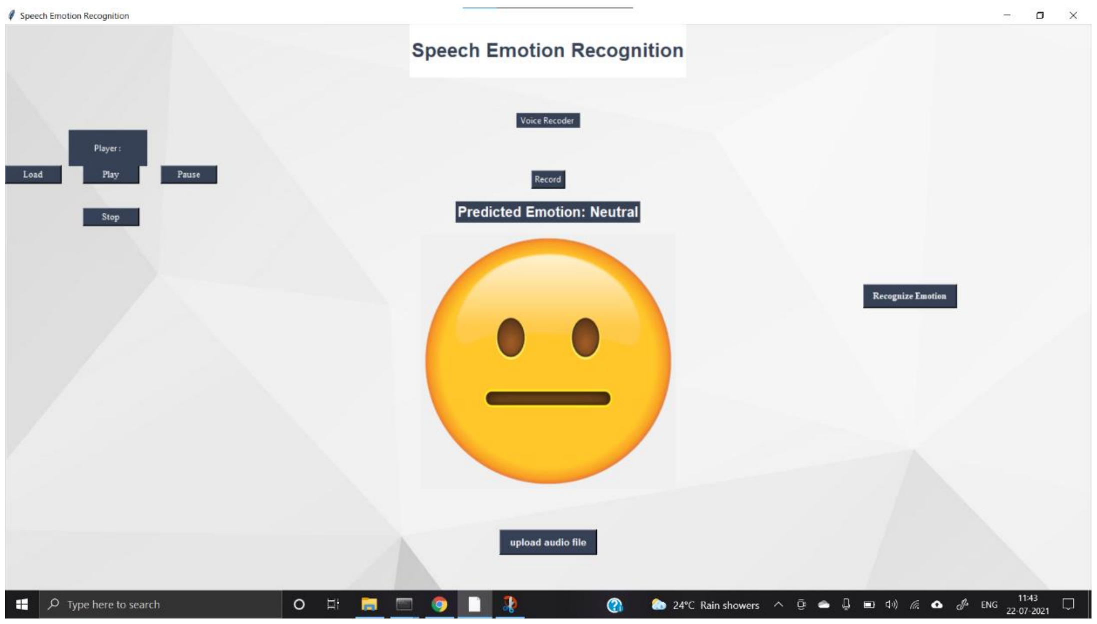
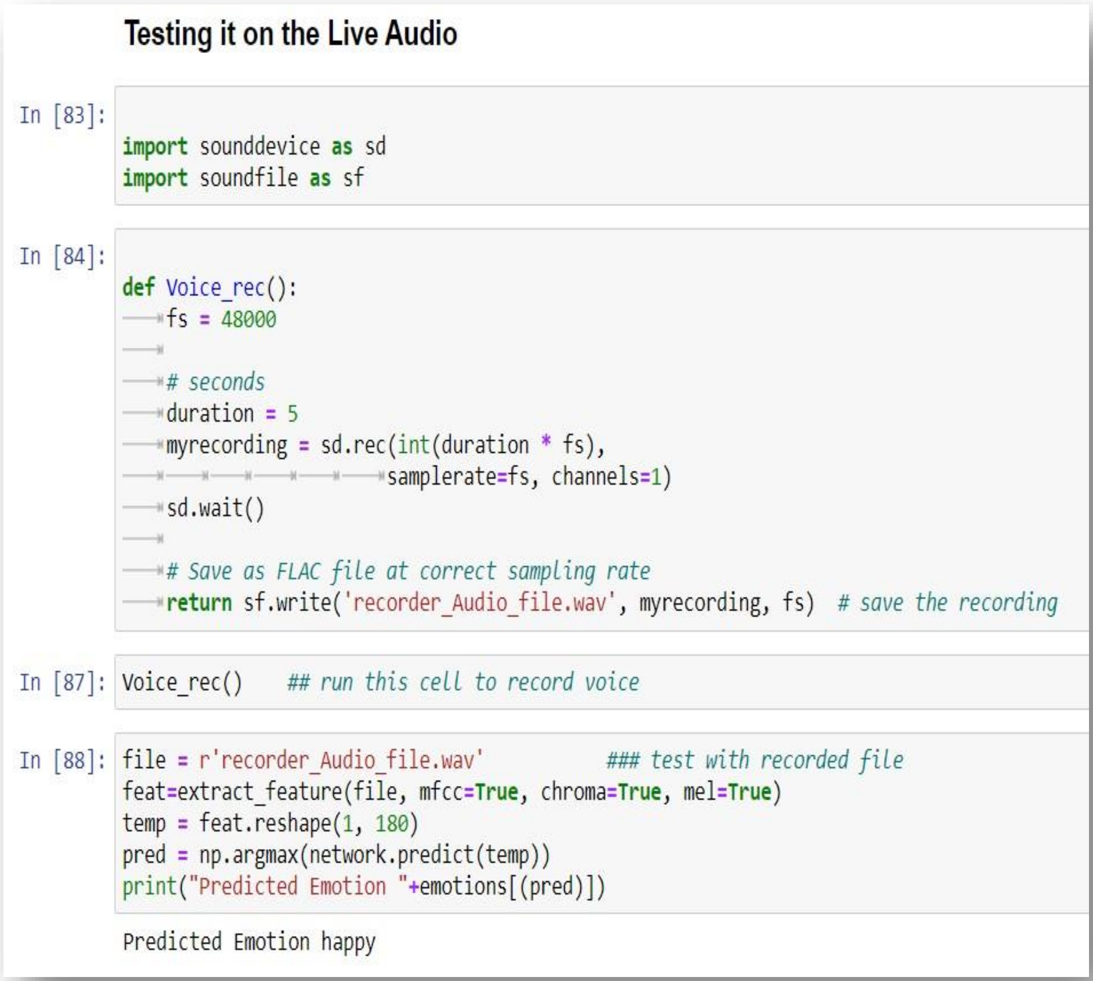


Figure 6.11 Neutral Emotion Prediction



Testing it on the Live Audio

```
In [83]:
import sounddevice as sd
import soundfile as sf
```

```
In [84]:
def Voice_rec():
    fs = 48000

    # seconds
    duration = 5
    myrecording = sd.rec(int(duration * fs),
                        samplerate=fs, channels=1)
    sd.wait()

    # Save as FLAC file at correct sampling rate
    return sf.write('recorder_Audio_file.wav', myrecording, fs)  # save the recording
```

```
In [87]: Voice_rec()    ## run this cell to record voice
```

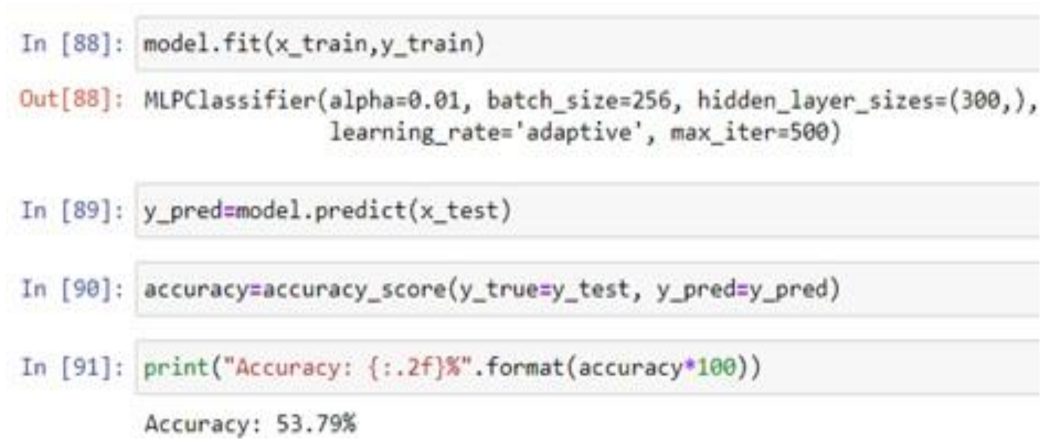
```
In [88]: file = r'recorder_Audio_file.wav'          ### test with recorded file
         feat=extract_feature(file, mfcc=True, chroma=True, mel=True)
         temp = feat.reshape(1, 180)
         pred = np.argmax(network.predict(temp))
         print("Predicted Emotion "+emotions[(pred)])

         Predicted Emotion happy
```

Figure 6.12 Live Audio Emotion Prediction

Since Neural Network Algorithm gave us an accuracy of 80% during the testing phase. So we have gone for the same Algorithm to be used for our Live audio test where the input is audio directly from the users microphone Our live audio demo had accuracy of 80% where 4 out of 5 times it predicted the right emotion

6.4 Performance Evaluation

After training the 2 models that is MLP and N e u r a l N e t w o r k ,the performance evaluation done for test data, by using evaluation matrics such as accuracy. The accuracy obtained from MLP Classifier is 53.79%.(Refer Fig 6.11)



```
In [88]: model.fit(x_train,y_train)

Out[88]: MLPClassifier(alpha=0.01, batch_size=256, hidden_layer_sizes=(300,),
                       learning_rate='adaptive', max_iter=500)

In [89]: y_pred=model.predict(x_test)

In [90]: accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)

In [91]: print("Accuracy: {:.2f}%".format(accuracy*100))

         Accuracy: 53.79%
```

Figure 6.13 MLP Accuracy

MLP Classifier and neural network algorithm has an accuracy score of 54% and 80% respectively for emotion prediction

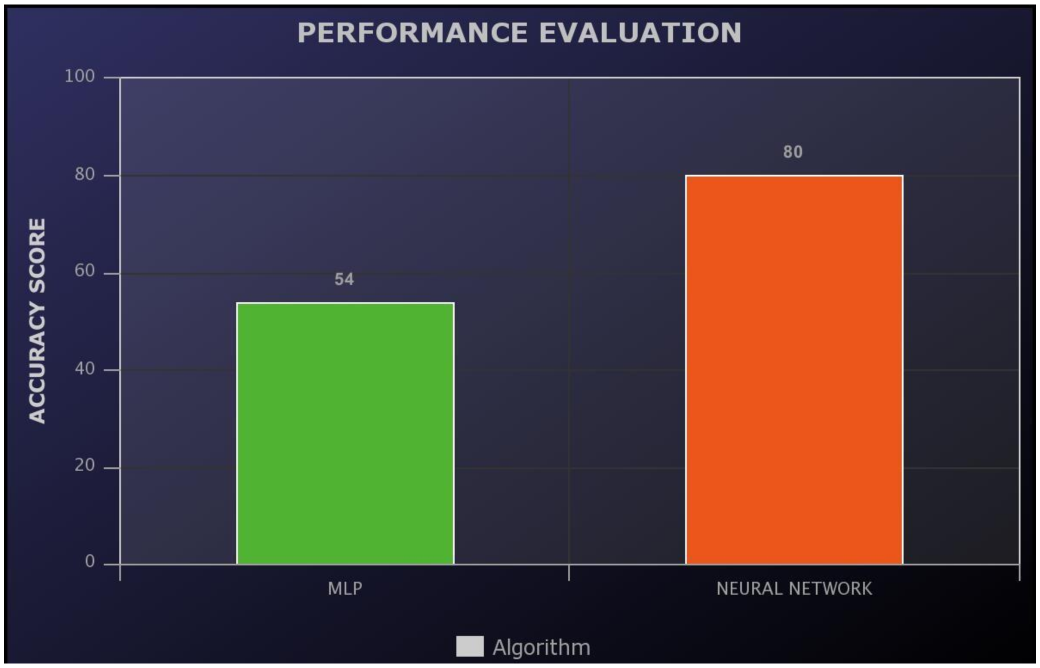


Figure 6.14 Performance Evaluation

.

6.5 Summary

In the proposed system, the main objective is to predict the type of emotion found in audio file input by employing a supervised learning approach. After training the models (MLP and Neural Network) the performance evaluation done for test data, by using evaluation matrics such as accuracy. The accuracy obtained from MLP Classifier is 53.79% and the accuracy obtained from Neural Network is 80.0%. Hence, we can conclude that the Neural Network gives the higher accuracy and use them to predict our live audio testing which is done by microphone audio input by the user.

CONCLUSION

The emerging growth and development in the field of AI and machine learning have led to the new era of automation. Most of these automated devices work based on voice commands from the user. Many advantages can be built over the existing systems if besides recognizing the words, the machines could comprehend the emotion of the speaker (user). Some applications of a speech emotion detection system are computer-based tutorial applications, automated call center conversations, a diagnostic tool used for therapy and automatic translation system.

In this thesis, the steps of building a speech emotion detection system were discussed in detail and some experiments were carried out to understand the impact of each step. Initially, the limited number of publically available speech database made it challenging to implement a well-trained model. Next, several novel approaches to feature extraction had been proposed in the earlier works, and selecting the best approach included performing many experiments. Finally, the classifier selection involved learning about the strength and weakness of each classifying algorithm with respect to emotion recognition. At the end of the experimentation, it can be concluded that an integrated feature space will produce a better recognition rate when compared to a single feature.

For future advancements, the proposed project can be further modeled in terms of efficiency, accuracy, and usability. Additional to the emotions, the model can be extended to recognize feelings such as depression and mood changes. Such systems can be used by therapists to monitor the mood swings of the patients. A challenging product of creating machines with emotion is to incorporate a sarcasm detection system. Sarcasm detection is a more complex problem of emotion detection since sarcasm cannot be easily identified using only the words or tone of the speaker. A sentiment detection using vocabulary, can be integrated with speech emotion detection to identify a possible sarcasm. Therefore,in the future, there would emerge many applications of a speech-based emotion recognition system.

REFERENCES

[1] Soegaard, M. and Friis Dam, R. (2013). The Encyclopedia of Human-Computer Interaction. 2nd ed.

[2] Developer.amazon.com. (2018). Amazon Alexa. [online] Available at: https://developer.amazon.com/alexa

[3] Store.google.com. (2018). Google Home Tips & Tricks – Google Store. [online] Available at: https://store.google.com/product/google_home_learn

[4] Apple. (2018). iOS - Siri. [online] Available at: https://www.apple.com/ios/siri/

[5] The Official Samsung Galaxy Site. (2018). What is S Voice?. [online] Available at: http://www.samsung.com/global/galaxy/what-is/s-voice/ [Accessed 2 May 2018].

[6] Gartner.com. (2018). Gartner Says 8.4 Billion Connected. [online] Available at: https://www.gartner.com/newsroom/id/3598917.

[7] H. Cao, R. Verma, and A. Nenkova, "Speaker-sensitive emotion recognition via ranking: Studies on acted and spontaneous speech," Comput. Speech Lang., vol. 28, no. 1, pp. 186–202, Jan. 2015.

[8] L. Chen, X. Mao, Y. Xue, and L. L. Cheng, "Speech emotion recognition: Features and classification models," Digit. Signal Process., vol. 22, no. 6, pp. 1154–1160, Dec. 2012.

[9] T. L. Nwe, S. W. Foo, and L. C. De Silva, "Speech emotion recognition using hidden Markov models," Speech Commun., vol. 41, no. 4, pp. 603–623, Nov. 2003.

[10] J. Rong, G. Li, and Y.-P. P. Chen, "Acoustic feature selection for automatic emotion recognition from speech," Inf. Process. Manag., vol. 45, no. 3, pp. 315–328, May 2009.

[11] S. S. Narayanan, "Toward detecting emotions in spoken dialogs," IEEE Trans. Speech Audio Process., vol. 13, no. 2, pp. 293–303, Mar. 2005.

[12] Dupuis, K. and Pichora-Fuller, M. (2010). [Collection] University of Toronto, Psychology Department, Toronto emotional speech set (TESS). Toronto.

[13] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera. KEEL Data- Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17:2-3 (2011) 255-287.