SPEECH EMOTION RECOGNITION USING MACHINE LEARNING IN PYTHON

MINI PROJECT

Submitted by

LAKSHMIPRIYA BM	**(211519205085)**
RACHEAL SWARNA KUMAR S	**(211519205119)**
YAMINI M	**(211519205180)**

In partial fulfillment for the award of the degree

Of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

PANIMALR INSTITUTE OF TECHNOLOGY

ANNA UNIVERSITY: CHENNAI-600 025

JUNE 2022

PANIMALAR INSTITUTE OF TECHNOLOGY

ANNA UNIVERSITY: CHENNAI-600025

BONAFIDE CERTIFICATE

Certified that this project report titled **“SPEECH EMOTION RECOGNITION USING MACHINE LEARNING IN PYTHON”** is the bonafide work of “**LAKSHMIPRIYA BM (211519205085), RACHEAL SWARNA KUMAR S(211519205119) and YAMINI M (211519205180)”** who carried out the project work under my supervision.

SIGNATURE

Dr.KavithaSubramani,M.E,P.hD.,
HEAD OF THE DEPARTMENT

Department of Information Technology,
Panimalar Institute of Technology,
Poonamallee, Chennai 600123.

SIGNATURE

G.Dhanalakshmi M.E.,(P.hD.),
SUPERVISOR
ASSOCIATE PROFESSOR

Department of Information Technology,
Panimalar Institute of Technology,
Poonamallee, Chennai 600123.

Certified that the candidates were examined in the university project viva-voce Examination held on ___________ at Panimalar Institute of Technology, Chennai 600123.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

A project of this magnitude and nature enquires kind co-operation and support from many, for successful completion. We wish to express our sincere thanks to all those who were involved in the completion of this project.

We seek the blessing from the **FOUNDER** of our institution **DR. JEPPIAAR, M.A., Ph.D.,** for having been a role model who has been our source of inspiration behind our success in education in his premier institution.

Our sincere thanks to the Honorable Chairman of our prestigious institution **Mrs. REMIBAI JEPPIAAR** for her sincere endeavor in educating us in her premier institution.

We would like to express our deep gratitude to our beloved **Secretary and Correspondent Dr. P. CHINNADURAI, M.A., Ph.D.,** for his kind words and enthusiastic motivation which inspired us a lot in completing this project.

We also express our sincere thanks and gratitude to our dynamic Directors **Mrs. C. VIJAYA RAJESWARI, Dr. C. SAKTHI KUMAR, M.E., Ph.D., AND Mrs. S.SARANYA SREE SAKTHI KUMAR, B.E., M.B.A,** for providing us with necessary facilities for completion of this project.

We also express our appreciation and gratefulness to our respected **Principal Dr. T. JAYANTHY, M.E., Ph.D.,** who helped us in the completion of the project. We wish to convey our thanks and gratitude to our Head of the Department, **Dr.KAVITHASUBRAMANI, M.E, Ph.D.,** for her full support by providing ample time to complete the project.

Special thanks to our Project Guide **Mrs. G.DHANALAKSHMI, M.E., (Ph.D.), Associate professor** for her expert advice, valuable information, and guidance throughout the completion of the project.

Last, we thank our parents and friends for providing their extensive moral support and encouragement during the course of the project.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO.
I	**ABSTRACT**	**1**
II	**LIST OF FIGURES**	**2**
III	**LIST OF ABBREVIATIONS**	**3**
1.	**INTRODUCTION**	**4**
2.	**LITERATURE SURVEY**	**6**
3.	**SYSTEM ANALYSIS**	**10**
	3.1 EXISTING SYSTEM	10
	3.1.1 DRAWBACKS	10
	3.2 PROPOSED SYSTEM	11
	3.3 FEASIBILITY STUDY	11
	3.3.1 TECHNICAL FEASIBILITY	12
	3.3.2 ECONOMICAL FEASIBILITY	12
	3.3.3 SOCIAL FEASIBILITY	12
4.	**REQUIREMENT SPECIFICATION**	**13**
	4.1 INTRODUCTION	13
	4.2 HARDWARE SPECIFICATION AND SOFTWARE SPECIFICATION	13
	4.2.1 HARDWARE REQUIREMENT	13

4.2.2 SOFTWARE REQUIREMENT 13

4.2.3 PROGRAMMING LANGUAGE 14

4.3 EXTERNAL INTERFACE REQUIREMENT 14

4.3.1 USER INTERFACE 14

4.3.2 HARDWARE INTERFACE 14

4.3.3 SOFTWARE INTERFACE 14

4.4 LANGUAGE SPECIFICATION 14

4.4.1 PYTHON 15

4.4.2 VISUAL STUDIO 17

4.4.2.1 FEATURE 18

5. SYSTEM ARCHITECTURE 20

5.1 ARCHITECTURE DIAGRAM 20

5.2 USE CASE DIAGRAM 23

5.3 CLASS DIAGRAM 24

5.4 ACTIVITY DIAGRAM 25

5.5 SEQUENCE DIAGRAM 26

5.6 COLLABORATION DIAGRAM 27

5.7 STATE-CHART DIAGRAM 28

6. SYSTEM DESIGN AND IMPLEMENTATION 29

6.1 MODULES 29

6.2 MODULES DESCRIPTION 29

6.2.1 INPUT SIGNAL 29

6.2.2 FEATURE EXTRACTION 29

6.2.3 SVM MODEL 30

6.2.4 TRAINING AND TESTING 30

6.2.5 EMOTION PREDICTION 30

7. TESTING 31

7.1 INTRODUCTION 31

7.2 TESTING OBJECTIVES 31

7.3 TYPES OF TESTING 32

7.3.1 UNCONVENTIONAL TESTING 32

7.3.2 CONVENTIONAL TESTING 33

7.4 TEST CASE DESIGN 33

7.4.1 UNIT TEST 33

7.4.2 INTEGRATION TEST 33

7.4.3 FUNCTIONAL TEST 34

7.4.4 SYSTEM TEST 35

7.4.5 ACCEPTANCE TEST 35

7.4.6 INTERFACE TEST 35

7.5 TESTING STRATEGIES 35

7.5.1 WHITE BOX TESTING 36

7.5.2 BLACK BOX TESTING 36

8.	**SCREENSHOTS**	**37**
9.	**APPENDIX - SOURCE CODE**	**38**
10.	**CONCLUSION AND FUTURE ENHANCEMENTS**	**48**
11.	**REFERENCES**	**49**

ABSTRACT

Recognizing emotion is a key feature needed to build socially aware systems. Therefore, it is an important part of human-computer interaction (HCI). Speech interaction between human interlocutors and between humans and machines is inescapably embedded within the laws and conditions of communication, which comprise the encoding and decoding of meaning as well. Here we deal with the human and machine interaction through synthesis and recognition applications. Speech emotion recognition (SER) has enormous potential given the ubiquity of speech-based devices. It involves capturing and digitizing the sound waves and contextually analyzing the emotions using tone, pitch, rhythm, etc., SER is usually beneficial for applications, which require human-computer interaction like speech synthesis, Speech recognition allows you to provide input to an application with your voice and the speech features, Mel-frequency Cepstral Coefficients (MFCC) is extracted from speech utterance. The applications and limitations enlighten the impact of speech processing in our modern technical field. This project drives increasing attention towards the recognition of emotions spontaneously from speech with thought-provoking tasks by Emotion Recognition from Speech using Machine Learning- Support Vector Machine Algorithm.

LIST OF FIGURES

FIGURE NUMBER	FIGURE NAME	PAGE NUMBER
5.1	ARCHITECTURE DIAGRAM	20
5.2	USE CASE DIAGRAM	23
5.3	CLASS DIAGRAM	24
5.4	ACTIVITY DIAGRAM	25
5.5	SEQUENCE DIAGRAM	26
5.6	COLLABORATION DIAGRAM	27
5.7	STATE-CHART DIAGRAM	28
8.1	EMOTION PREDICTION FOR FEMALE VERSION	37
8.2	EMOTION PREDICTION FOR MALE VERSION	37

LIST OF ABBREVIATIONS

SER	Speech Emotion Recognition
ASR	Automatic speech recognition
SVM	Support Vector Machine
MFCC	Mel-frequency Cepstral Coefficient
HCI	Human-Computer Interaction
IDE	Integrated Developed Environment

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVES

The main objective of this paper is to detect speech emotion correctly by using neural network techniques. The objective of Speech emotion recognition using Machine Learning is to acknowledge who is talking, speaker check, where the target is to verify a speaker's asserted character by identifying their speech signal, and word spotting, which incorporates observing a speech for the event of determined words. Speech emotion recognition is usually beneficial for applications, which require human-computer interaction like speech synthesis, customer service, education, forensics, and medical analysis. Recognizing emotional conditions in speech signals are so much challengeable area for many several reasons.

1.1 OVERVIEW OF THE PROJECT

Emotion recognition can play an important role in various fields such as healthcare (mood profiles) education (tutoring) and security and defense (surveillance). Speech emotion recognition (SER) has enormous potential given the ubiquity of speech-based devices. However, it is important that SER models generalize well across different conditions and settings showing robust performance. Conventionally, emotion recognition systems are trained with supervised learning solutions. The generalization of the models is often emphasized by training on a variety of samples with diverse labels. The state-of-the-art models for standard computer vision tasks utilize thousands of labeled samples. Similarly, automatic speech recognition (ASR) systems are trained on several hundred hours of data with transcriptions. Generally, labels for emotion recognition tasks are collected with perceptual evaluations from multiple evaluators. The raters annotate samples by listening or watching the stimulus. This evaluation procedure is cognitively intense and expensive. Therefore, standard benchmark datasets for SER have a limited number of sentences with emotional labels, often collected from a limited number of evaluators. This limitation severely affects the generalization of the systems. An alternative approach to increase the generalization of the models is by building robust models. An effective approach to achieve this goal is with Neural Network. These auxiliary tasks correspond to the reconstruction of feature representations at various layers in a neural network classifier.

CHAPTER 2

LITERATURE SURVEY

A literature survey or a literature review in a project report is that section that 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.

The purpose of the Literature Survey is to give a brief overview and also to establish complete information about the reference papers. The goal of the Literature Survey is to completely specify the technical details related to the main project concisely and unambiguously.

2.1 Dias Issa, M. Fatih Demirci, Adnan Yazici, Speech emotion recognition with deep convolutional neural networks, Biomedical Signal Processing and Control, Volume 59,2020,101894, ISSN 1746-8094.

They introduce a new architecture, which extracts Mel-frequency cepstral coefficients, chromatogram, Mel-scale spectrogram, Tonnetz representation, and spectral contrast features from sound files and uses them as inputs for the one-dimensional Convolutional Neural Network. In this project all of the proposed models work directly with raw sound data without the need for conversion to visual representations, unlike some previous approaches.

MERITS:

- Its Sound files are represented effectively by combining various features.
- Its framework sets the new SOTA on two datasets for speech emotion recognition.

- The advantage of the framework are its simplicity, applicability and generality.

DEMERITS:

- These frameworks contain low storage.

2.2 Jianhua Zhang, Zhong Yin, Peng Chen, Stefano Nichele, Emotion recognition using multi-modal data and machine learning techniques: A tutorial and review, Information Fusion, Volume 59,2020, Pages 103-126, ISSN 1566-2535

In this paper, the emotion recognition methods based on multi-channel EEG signals as well as multi-modal physiological signals are reviewed. They compare different ML and deep learning algorithms for emotion recognition and suggest several open problems and future research directions in this exciting and fast-growing area of AI.

MERITS:

- Different types of machine learning classifiers for emotion recognition are reviewed.
- Several major EEG feature extraction methods and EEG feature reduction methods are introduced.

DEMERITS:

- Several open problems and future research directions in the area of emotion recognition are not identified fully

2.3 T.Sai Samhith, G.Nishika, M.Prayuktha, M.Bharat Chandra, Dr.Sunil Bhutada, G.Prasadu, Speech Emotion Recognition using Machine Learning Algorithms, Volume 9, Issue 6 June 2021, © 2021 IJCRT, ISSN: 2320-2882

This paper is about recognizing the emotions of a person from a speech, to identify emotions using machine learning algorithms. They considered the classifiers to include random forest, extra trees, gradient boosting, decision tree, and light gradient boosting classifiers. They took some datasets, trained them using the classifiers mentioned, and got the results.

MERITS:

- Labelled speech samples were buffered into short-time blocks.
- The Torronto Emotional Speech Set (TESS) dataset that they considered is fine-tuned.

DEMERITS:

- They confined the quality of the feature selection affects the emotion recognition rate.

2.4 Leila Kerkeni, Youssef Serrestou, Mohamed Mbarki, Kosai Raoof, Mohamed Ali Mahjoub, and Catherine Cleder, Automatic Speech Emotion Recognition Using Machine Learning, March 2019

This application comprises the use of leftover stuff or item or money which can be used by poor ones. This application, provides features such as searching nearby NGOs, Chatbot through AI, Generate Report through ML, Push Notification, Support for multiple languages, Upload videos, pictures ,blogs, news for events, Payment Gateway, and Authentication through OTP.

MERITS:

- They presented an automatic speech emotion recognition (SER) system using three machine learning algorithms (MLR, SVM, and RNN) to classify seven emotions.
- SER reported the best recognition rate of 94% on the Spanish database.
- Berlin database, all of the classifiers achieve an accuracy of 83% .

DEMERITS:

- Feature selection techniques show that more information is not always good in machine learning applications.
- Recurrent neural networks (RNN) suffer from the problem of very long training times.
- the quality of the
- feature selection affects the emotion recognition rate.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

- The existing system uses robust speech recognition. The matter is that it's having high latency and low accuracy. It's also time-consuming. The speech depends upon the surrounding environment, length of speech used, and quiet emotions each individual has.
- The system is split into two stages,
  1. The primary stage may be a training stage where the system first trains for various voices or emotions.
  2. The second stage is to acknowledge the pattern by using the characteristics derived from the primary task.
- Principal Component Analysis
- Geometric methods.

3.1.1 DRAWBACKS

- Low discriminatory power and high computational load.
- In geometric-based methods, the geometric features like the distance between speech signals.

3.2PROPOSED SYSTEM

- Speech Emotion recognition for transform features system through textural analysis and NN classifier.
- The MFCC has been used as a feature for classifying the speech data into various emotion categories employing artificial neural networks.
- This technique manages a good balance between the computational volume and performance of accuracy of the real-time processes.
- The project is implemented to recognize the speech of the person by recognizing and hearing the voice of an individual. The system can identify the voice of the individual or a person whether that person is sad, happy, or angry. The relevant details about who is predicted to possess those needs, and what features the voice interface have got to meet those anticipated needs.

3.3 FEASIBILITY STUDY

The feasibility of the project is analyzed during this part and a business proposal is a place forth with a really general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be applied. This can be to confirm that the proposed system isn’t a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. 3 key considerations involved in the feasibility analysis are

- TECHNICAL FEASIBILITY
- ECONOMICAL FEASIBILITY
- SOCIAL FEASIBILITY

3.3.1 TECHNICAL FEASIBILITY:

This project needs a computer or a smartphone through which the website can be accessed. And a database that will store the details about the donations made.

3.3.2 ECONOMICAL FEASIBILITY:

The implementation of this application can be done with the help of open-source environments. In this application, the internet is required for the website. It will be very easy and less costly to implement this website on an android device or a computer. Since the cost of the system is only the implementation cost of the system. There's no need to spend any monthly thereafter. It is economically feasible.

3.3.3 SOCIAL FEASIBILITY:

The facet of the study is to check the level of acceptance of the system by the user. This includes the method of coaching the user to use the system efficiently. The user should not feel vulnerable by the system, instead must accept it as a necessity. The amount of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence should be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER 4

REQUIREMENT SPECIFICATIONS

4.1 INTRODUCTION

The requirements specification could be a technical specification of requirements for the software products. It’s the primary step in the requirements analysis process it lists the requirements of a particular software system including functional, performance, and security requirements. The requirements also provide usage scenarios from a user, an operational, and an administrative perspective. The purpose of software requirements specification is to provide a detailed overview of the software project, its parameters, and goals. This describes the project’s audience and its program, hardware, and software requirements. It defines how the shopper, team, and audience see the project and its functionality.

4.2 HARDWARE AND SOFTWARE SPECIFICATION

4.2.1 HARDWARE REQUIREMENT

Processor	:	Core i3
Motherboard	:	Genuine Intel
RAM	:	8 GB
Hard Disk	:	500 GB

4.2.2 SOFTWARE REQUIREMENT

Operating system	:	Windows XP / above 10
Technology Used	:	MATLAB 7.5 and above versions

4.2.3 PROGRAMMING LANGUAGE

Python

4.3 EXTERNAL INTERFACE REQUIREMENT:

4.3.1 USER INTERFACE

- We define speech emotion recognition (SER) systems as a collection of methodologies that process and classify speech signals to detect embedded emotions. SER is not a new field, it has been around for over two decades, and has regained attention thanks to the recent advancements..

4.3.2 HARDWARE INTERFACE

- For voice recognition to work, you must have a computer with a sound card and either a microphone or a headset. Other devices like smartphones have all of the necessary hardware built into the device.

4.3.3 SOFTWARE INTERFACE

- Speech recognition software is a computer program that's trained to take the input of human speech, interpret it, and transcribe it into text.

4.4 LANGUAGE SPECIFICATION

This project can be implemented using Python because it is easy and convenient to create the user website.

4.4.1 PYTHON:

Speech recognition in Python works with algorithms that perform linguistic and acoustic modeling. Acoustic modeling is used to recognize phenones/phonetics in our speech to get the more significant part of speech, as words and sentences.

Movies and TV shows love to depict robots who can understand and talk back to humans. Shows like Westworld, movies like Star Wars and I, Robot are filled with such marvels. But what if all of this exists in this day and age? Which it certainly does. You can write a program that understands what you say and respond to it.

All of this is possible with the help of speech recognition. Using speech recognition in Python, you can create programs that pick up audio and understand what is being said. In this tutorial titled 'Everything, You Need to Know About Speech Recognition in Python

ADVANTAGES:

Following are some of the advantages of using Python for website development.

1. Easy to Read, Learn and Write - Python is a high-level programming language that has English-like syntax. This makes it easier to read and understand the code.

Python is really easy to pick up and learn, that is why a lot of people recommend Python to beginners. You need fewer lines of code to perform the same task as compared to other major languages like C/C++ and Java.

2. Improved Productivity - Python is a very productive language. Due to the simplicity of Python, developers can focus on solving the problem. They don't need to spend too much time understanding the syntax or behavior of the programming language. You write less code and get more things done.

3. Interpreted Language - Python is an interpreted language which means that Python directly executes the code line by line. In case of any error, it stops further execution and reports back the error which has occurred.

Python shows only one error even if the program has multiple errors. This makes debugging easier.

4. Dynamically Typed - Python doesn't know the type of variable until we run the code. It automatically assigns the data type during execution. The programmer doesn't need to worry about declaring variables and their data types.

5. Free and Open-Source - Python comes under the OSI-approved open-source license. This makes it free to use and distribute. You can download the source code, modify it and even distribute your version of Python. This is useful for organizations that want to modify some specific behavior and use their version for development**.**

6. Vast Libraries Support - The standard library of Python is huge, you can find almost all the functions needed for your task. So, you don't have to depend on external libraries**.**

DISADVANTAGES:

1. Slow Speed - We discussed above that Python is an interpreted language and dynamically-typed language. The line-by-line execution of code often leads to slow execution.

The dynamic nature of Python is also responsible for the slow speed of Python because it has to do the extra work while executing code. So, Python is not used for purposes where speed is an important aspect of the project.

2. Not Memory Efficient - To provide simplicity to the developer, Python has to do a little tradeoff. The Python programming language uses a large amount of memory. This can be a disadvantage while building applications when we prefer memory optimization.

3. Weak in Mobile Computing - Python is generally used in server-side programming. We don’t get to see Python on the client-side or mobile applications because of the following reasons. Python is not memory efficient and it has slow processing power as compared to other languages.

4. Database Access - Programming in Python is easy and stress-free. But when we are interacting with the database, it lacks behind.

Python’s database access layer is primitive and underdeveloped in comparison to popular technologies like JDBC and ODBC.

Huge enterprises need smooth interaction of complex legacy data and Python is thus rarely used in enterprises.

5. Runtime Errors - As we know Python is a dynamically typed language so the data type of a variable can change at any time. A variable containing an integer number may hold a string in the future, which can lead to Runtime Errors.

Therefore, Python programmers need to perform thorough testing of the applications.

4.4.2 VISUAL STUDIO:

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE.

The first version of VS(Visual Studio) was released in 1997, named Visual Studio 97 having version number 5.0. The latest version of Visual Studio is 15.0 which was released on March 7, 2017. It is also termed Visual Studio 2017. The supported .Net Framework Versions in the latest Visual Studio are 3.5 to 4.7. Java was supported in old versions of Visual Studio but the latest version doesn't provide any support for Java language.

An integrated development environment (IDE) is a feature-rich program that supports many aspects of software development. The Visual Studio IDE is a creative launching pad that you can use to edit, debug, build code, and then publish an app. Over and above the standard editor and debugger that most IDEs provide, Visual Studio includes compilers, code completion tools, graphical designers, and many more features to enhance the software development process.

4.4.2.1 FEATURES:

Code editor- it includes a code editor that supports syntax highlighting an dcode completion using the IntelliSense for variables, functions, methods etc.The code editor is used for all supported languages.

Debugger- It works as both a source level debugger as well as a machine level language. The debugger can be configured to be launched when an application running outside the Visual Studio environment crashes.

Designer-Visual Studio includes a host of visual designers to aid in the development of applications. These tools include:

- Windows Form sDesigner
- WPF Designer
- Class designer
- Data designer

Extensibility-Visual Studio allows developers to write extensions for Visual Studio to extend its capabilities. Extensions are supported in the Standard (and higher) versions of Visual Studio 2005.

CHAPTER 5

SYSTEM ARCHITECTURE

5.1 ARCHITECTURE DIAGRAM:

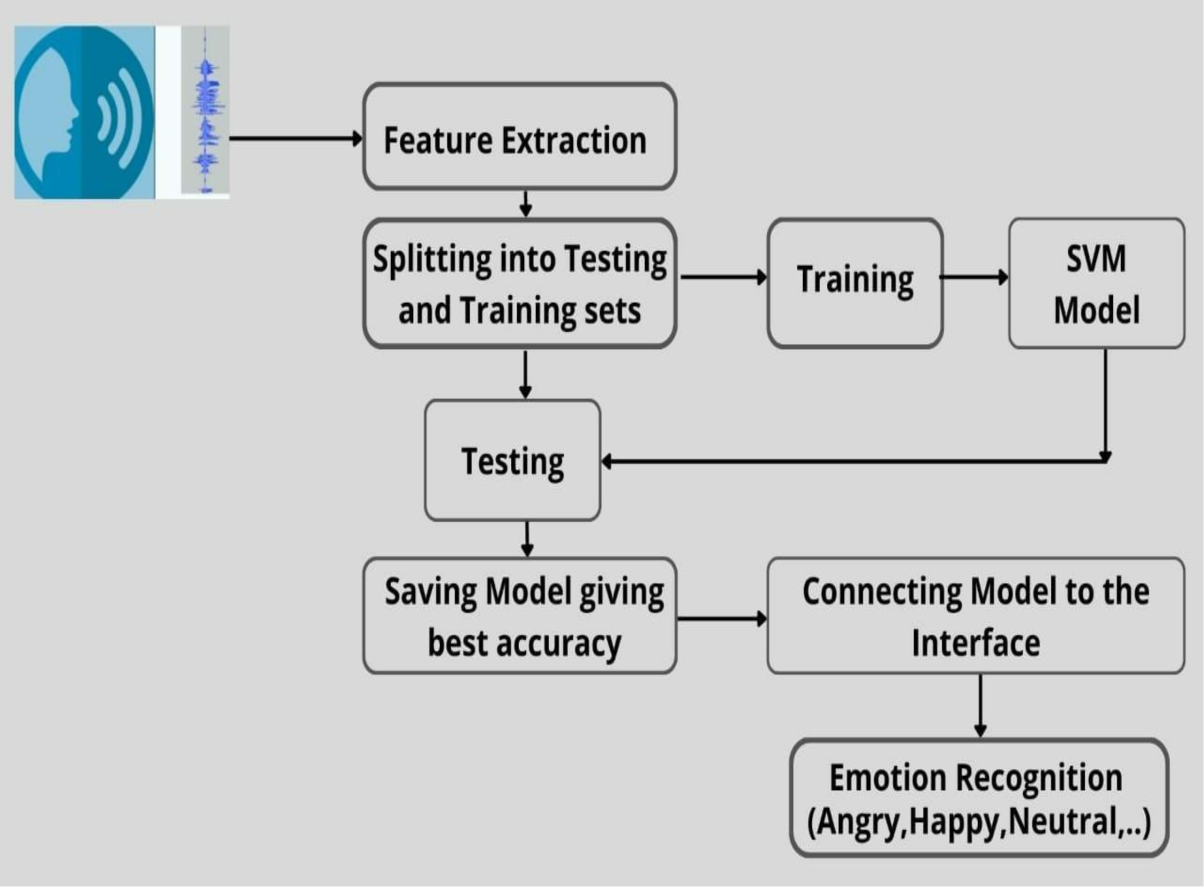

Feature Extraction
Splitting into Testing and Training sets
Training
SVM Model
Testing
Saving Model giving best accuracy
Connecting Model to the Interface
Emotion Recognition (Angry,Happy,Neutral,..)

Figure 5.1: Architecture Diagram

The system architecture of the proposed model consists of five modules:

- Input Signal
- Feature Extraction
- SVM Model
- Training and Testing
- Emotion Prediction

➢ **INPUT SIGNAL :**

The signal goes into an electronic system. Here is the input voice or audio file.

➢ **PREPROCESSING :**

- The technique of preparing (cleaning and organizing) the raw data to make it suitable for building and training Machine Learning models.

- Five major steps of data preprocessing :
- Preprocessing in Data Mining: Data preprocessing is a data mining technique that is used to transform the raw data into a useful and efficient format Steps
- Involved in Data Preprocessing
- Data Cleaning
- Data Transformation
- Data Reduction

➢ **TRAINING AND TESTING DATASETS:**

- Separating data into training and testing sets is an important part of evaluating data mining models. Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing.
- By default, the Test set is split into 30 % of actual data and the training set is split into 70% of the actual data.
- For training purposes,we use the predefined un trained dataset CSV file as main input for training the machine.

DATA SET SPLITTING SYNTAX:

train_test_split(*arrays, test_size=None, train_size=None, random_state=None, shuffle=True, stratify=None)

MEL-FREQUENCY CEPSTRUM COEFFICIENT (MFCC):

MFCC is the most used representation of the spectral property of voice signals. These are the best for speech recognition as it takes human perception sensitivity with respect to frequencies into consideration.

EMOTION RECOGNITION:

For detecting the emotion run the train.py program to train the data

5.2 USE CASE DIAGRAM:

A use case diagram is a graphical depictions of the possible interactions that can be made by the user in the system. The use case is a methodology used in system analysis to identify, clarify and organize system requirements. Use case diagram describes the system functionality as a set of tasks that the system must carry out.

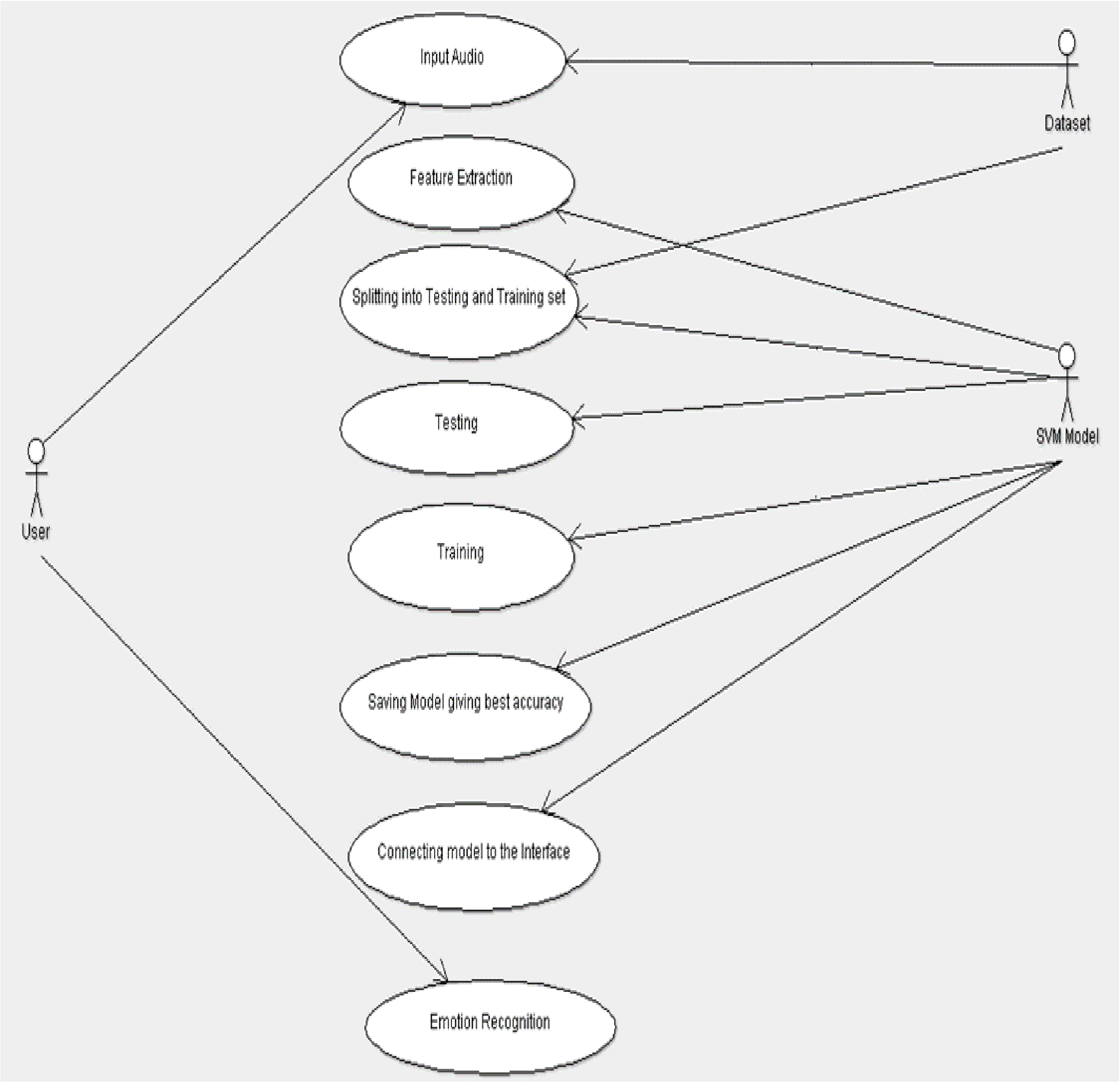

Input Audio
Dataset
Feature Extraction
Splitting into Testing and Training set
SVM Model
Testing
User
Training
Saving Model giving best accuracy
Connecting model to the Interface
Emotion Recognition

Figure 5.2 : Usecase Diagram

5.3 CLASS DIAGRAM:

The class diagram is a blueprint of the system to be implemented. The class diagram has three components such as Classname, Attributes, and Operations. Class diagrams are static and they display only the interactions that take place and not what happens when they interact.

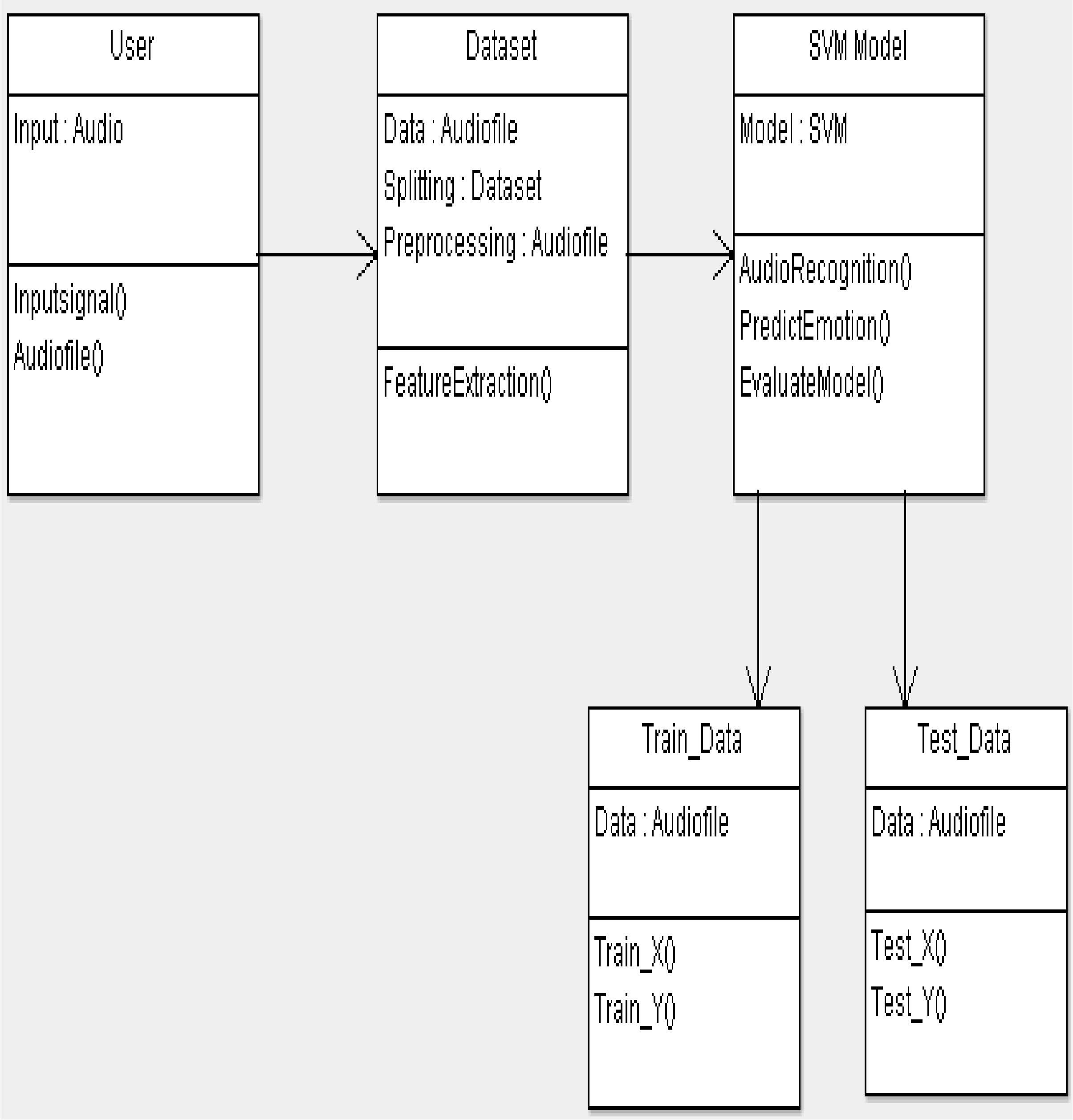


Figure 5.3 : Class Diagram

5.4 ACTIVITY DIAGRAM:

Activity diagram is nothing but the diagram that shows the workflow of the system. It is a graphical representations of worflowa of stepwise activities and actions with support for choice, iteration and concurrency. In the UML, activity diagrams are intended to model both computational and organizational processes.

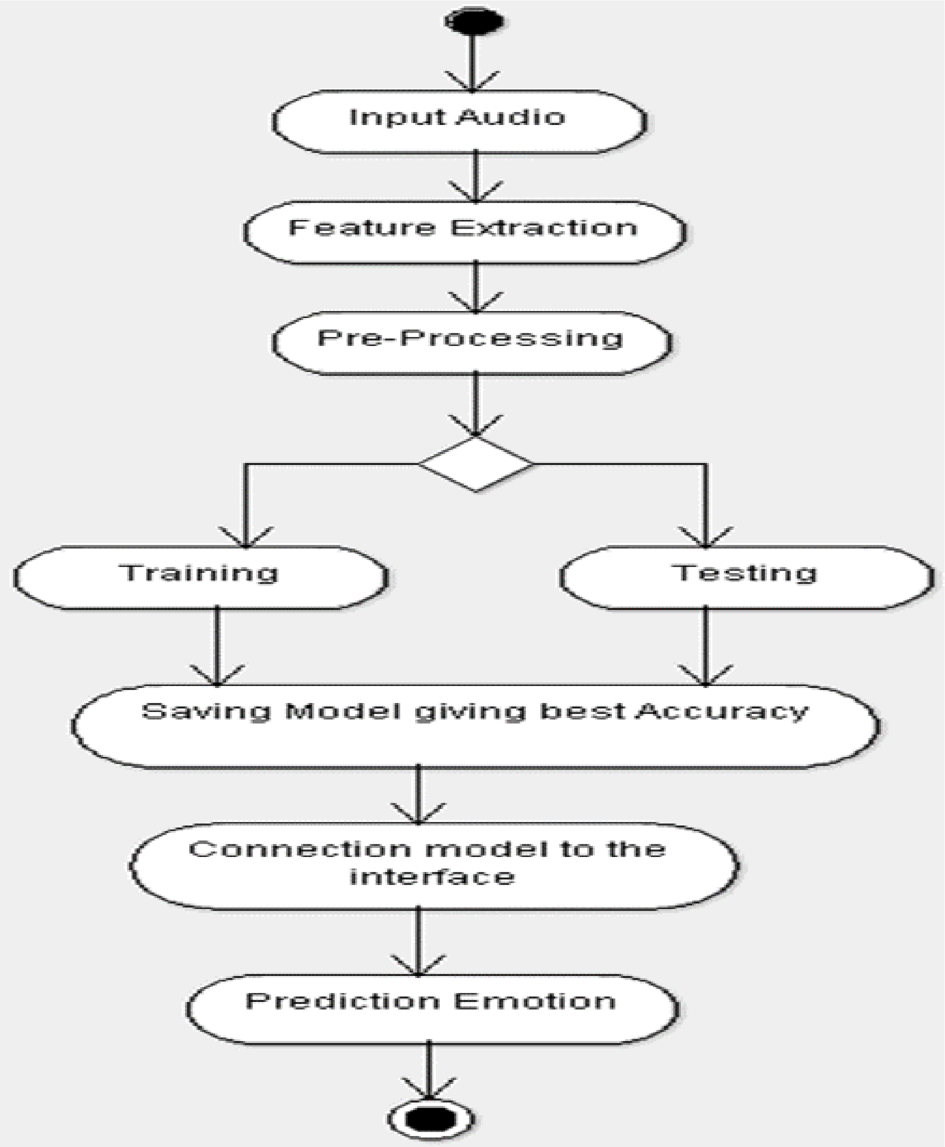


Figure 5.4 : Activity Diagram

5.5 SEQUENCE DIAGRAM:

A sequence diagram shows the processes invloved in the system and the sequence of the messages exchanged. These diagrams are used by the softeware developers and the professionals to better understand the requirements of the system.

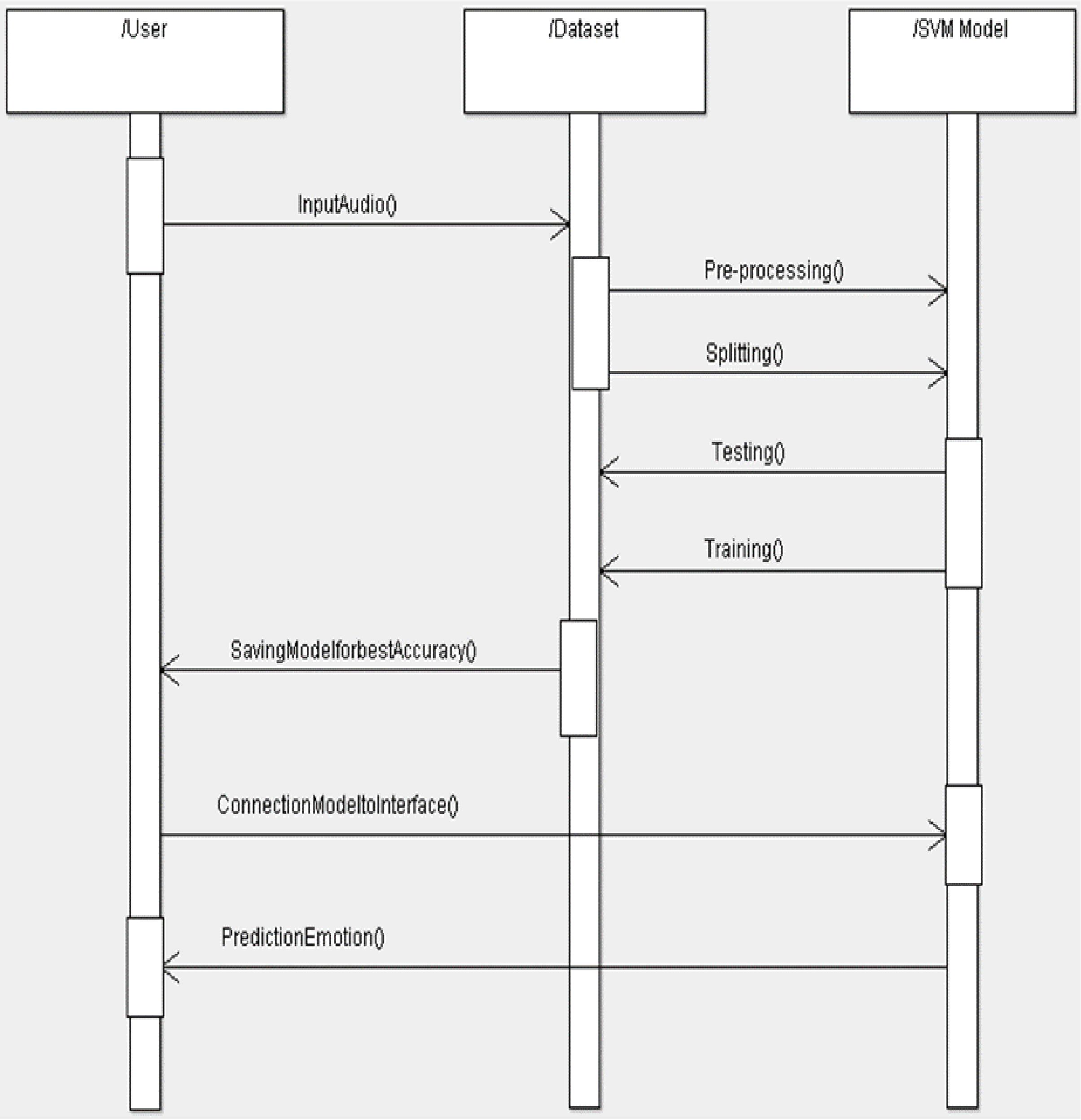


Figure 5.5 : Sequence Diagram

5.6 COLLABORATION DIAGRAM:

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.

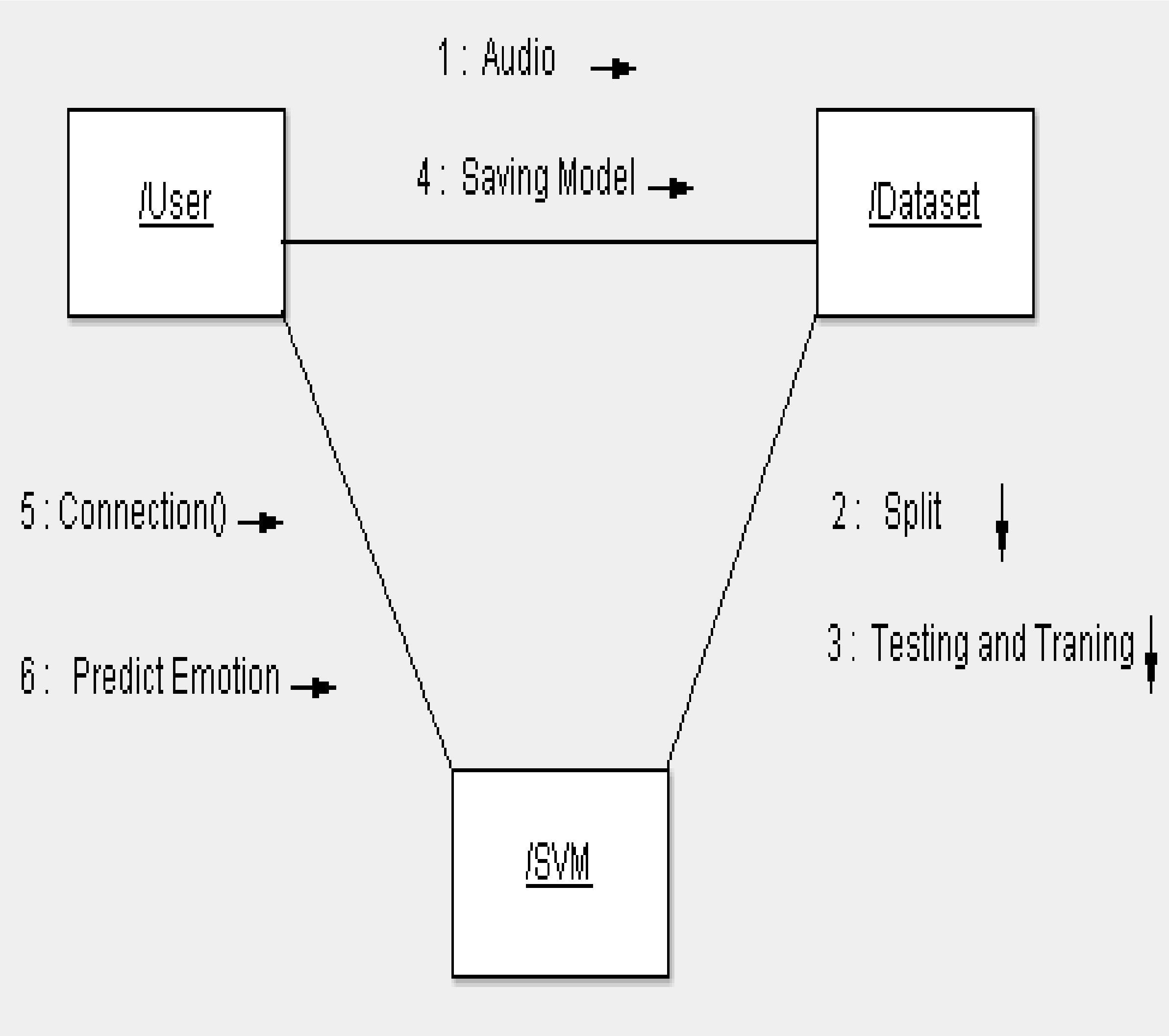


Figure 5.6 : Collaboration Diagram

5.7 STATE CHART DIAGRAM:

A state diagram is used to represent the condition of the system or part of the system at finite instances of time. It's a behavioral diagram and it represents the behavior using finite state transitions.

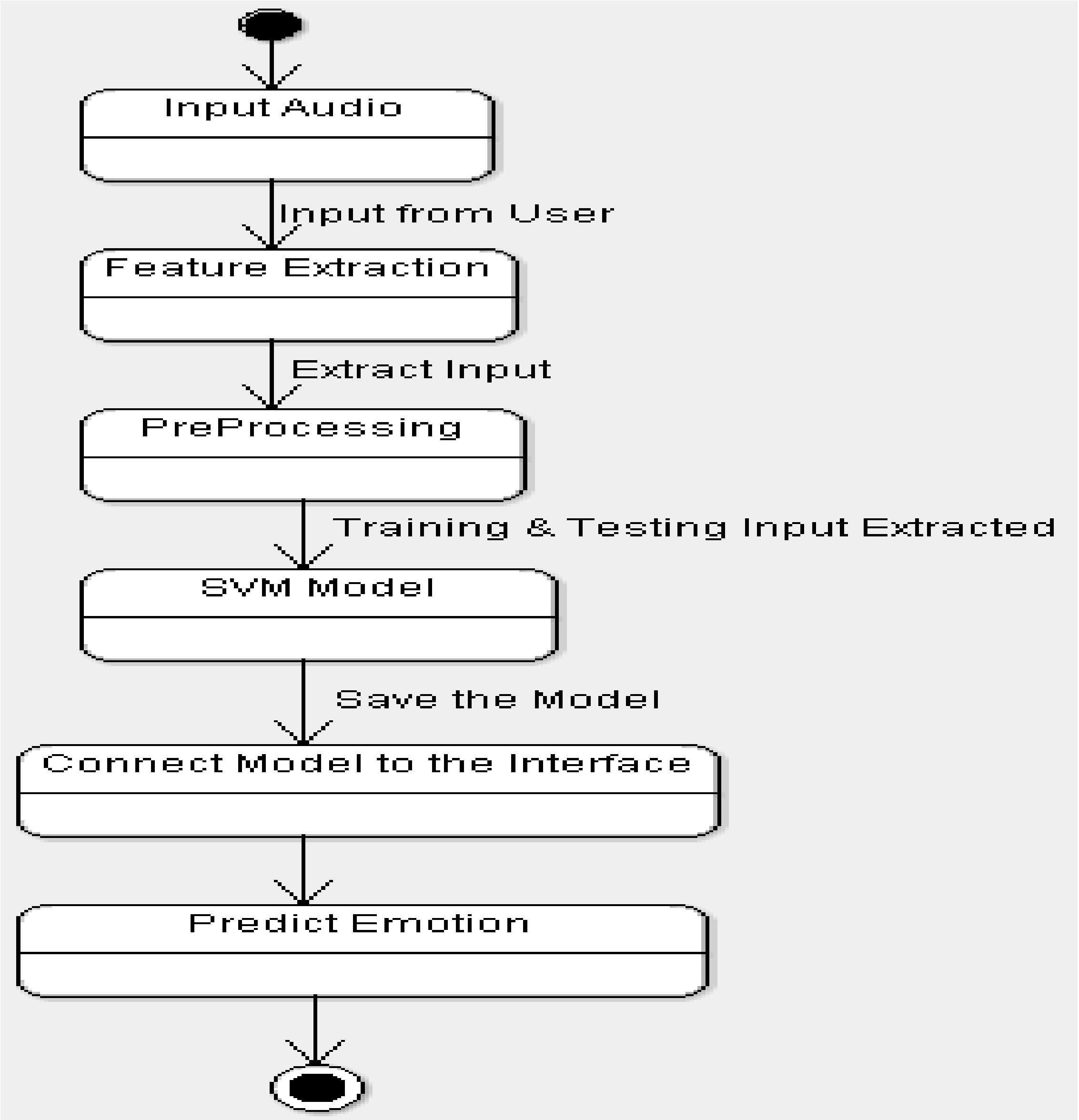


Figure 5.7 : State Chart Diagram

CHAPTER 6

SYSTEM DESIGN AND IMPLEMENTATION

6.1 MODULES

- Input Signal
- Feature Extraction
- SVM Model
- Training and Testing
- Emotion Prediction

6.2 MODULES DESCRIPTION

6.2.1 INPUT SIGNAL:

This module gives a brief detail about emotion. It also describes about the system. And helps to understand about the system better. We have used Python for this module.

6.2.2 FEATURE EXTRACTION:

The key to speech emotion recognition is feature extraction process. The quality of the features directly influences the accuracy of classification results. Typically, the feature extraction method designs handcraft features based on acoustic features of speech.

6.2.3 SVM MODEL:

The Support Vector Machine(SVM) is used as classifier to classify different emotional states such as anger, happiness, sadness, neutral, fear. The Berlin emotion database and Hindi emotion database are used for extracting the features from emotional speech .

6.2.4 TRAINING AND TESTING:

Testing is a medium of expression of one's perspective or one's mental state to others. Speech Emotion Recognition (SER) can be defined as extraction of the emotional state of the speaker from his or her speech signal.

Training task of recognizing the emotion from speech irrespective of the semantic contents. However, emotions are subjective and even for humans it is hard to notate them in natural speech communication regardless of the meaning.

6.2.5 EMOTION PREDICTION:

Emotion prediction is a method that recognizes the human emotion derived from the subject's psychological data. SVM were used for emotion intensity recognition. A comparative study and implementation of algorithms for measuring facial emotions and their intensities based on the different AUs (Action Units) are presented.

CHAPTER 7

TESTING

7.1 INTRODUCTION

The purpose of testing is to get errors. Testing is that the method of attempting to get each conceivable fault or weakness in a very work product. It provides how to examine the practicality of parts, sub-assemblies, assemblies and/or a finished product. it's the method of effort software package with the intent of guaranteeing that the computer code meets its needs And user expectations and doesn't fall in an unacceptable manner. There area unit varied varieties of testing. every take a look at kind addresses a particular testing demand.

7.2 TESTING OBJECTIVES

Testing may be a set of activities that may be planned earlier and conducted consistently. Testing is completed to seek out error. Testing is that the integral a part of entire development and maintenance method and used for quality assurance. The goal of testing section is to verify that the specification has been accurately and utterly incorporated into the look still on make sure the correctness of the look itself. For this reason, a model for software package testing, a collection of steps into that we are able to place specific test suit style techniques and testing ways ought to be outlined for software package method. Testing is one among the necessary steps in software package development section. Testing checks for the errors, the project testing involves:

- Static analysis is employed to analyze the structural properties of the ASCII text file
- Dynamic take a look at is employed to analyze the behavior of the ASCII text file by execution the program on the take a look at information

7.3 TYPES OF TESTING

Varieties of testing the aim of testing is to get errors. Testing is that the method of attempting to get each conceivable fault or weakness in a very work product. It provides how to examine the practicality of parts, sub-assemblies, assemblies and/or a finished product. it's the method of effort software package with the intent of guaranteeing that the computer code meets its needs and user expectations and doesn't fall in an unacceptable manner. There area unit varied varieties of testing. every take a look at kind addresses a particular testing demand.

There are two type of testing according to their behaviors:

1. Unconventional Testing
2. Conventional Testing

7.3.1 UNCONVENTIONAL TESTING

Unconventional testing may be a method of verification that is doing by SQA (Software Quality Assurance) team. it's a interference technique that is working from the beggary to ending of the project development. during this method SQA team verifies project development activities and insuring that developing project is fulfilling the necessity of the consumer or not in this testing the SQA team follows these methods:

- Peer review
- Code walks and throw

- Inspection
- Document Verification

7.3.2 CONVENTIONAL TESTING

Conventional Testing may be a method of finding the bugs and confirmatory the project. Testing team involves during this testing method and confirmatory that developed project is in step with consumer demand or not. This method may be a correction technique wherever testing team notice bugs and news to the event team for correction on developed

7.4 TEST CASE DESIGN

7.4.1 UNIT TEST

Unit testing involves the planning of test cases that validate that the inner program logic is functioning properly, which program inputs manufacture valid outputs. All call branches and internal code flow ought to be valid. it's the testing of individual software package units of the appliance. it's done when the completion of a personal unit before integration. this can be a structural testing, that depends on information of its construction and is invasive. Unit checks perform basic tests at element level and test a particular business method, application, and/or system configuration. Unit tests make sure that every distinctive path of a business method performs accurately to the documented specifications and contains clearly outlined inputs and expected results.

7.4.2 INTEGRATION TEST:

Software integration testing is that the progressive integration testing of 2 or additional integrated software package parts on one platform to provide failures caused by interface defects.

The task of the mixing check is to envision that parts or software package applications, e.g., parts during a software or – one accelerates – software package applications at the corporate level – move while not error.

Test results: All the check cases mentioned higher than have passed with success and no defects encountered. Integration testing is specifically aimed toward exposing the issues that arise from the mixture of parts.

7.4.3 FUNCTIONAL TEST:

Functional tests offer systematic demonstrations that functions tested ar obtainable as mere by the business and technical needs, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

organization and preparation of purposeful tests is targeted on needs, key functions, or special check cases. additionally, systematic coverage concerning determine Business method flows; information fields, predefined processes, and consecutive processes should be thought of for testing. Before purposeful testing is complete, further tests ar known and therefore the effective worth of current tests is decided

7.4.4 SYSTEM TEST:

System testing ensures that the complete integrated software meets the necessities. It tests a configuration to confirm far-famed and certain results. associate example of system checking is that the configuration destined system

integration test. System testing is predicated on method descriptions and flows, accenting pre-driven method links and integration points.

7.4.5 ACCEPTANCE TEST:

User acceptance testing may be a crucial part of any project and needs vital participation by the top user. It additionally ensures that the system meets the purposeful needs.

Test results: All the check cases mentioned higher than have passed with success and no defects encountered.

7.4.6 INTERFACE TEST:

The Interface Testing is performed to verify the interfaces between sub modules whereas playacting integration of sub modules aiding master module recursively.

7.5 TESTING STRATEGIES

A number of software testing strategies have been proposed in the literature All offer the software package developer with a guide for testing and everyone have the subsequent generic characteristics:

- Testing begins at the element level and works “outward” toward the mixing of the complete computer-based system.

- Different testing techniques ar applicable at totally different points in time.

- The developer of the s/w conducts testing and for big comes, freelance check cluster.

7.5.1 WHITE BOX TESTING:

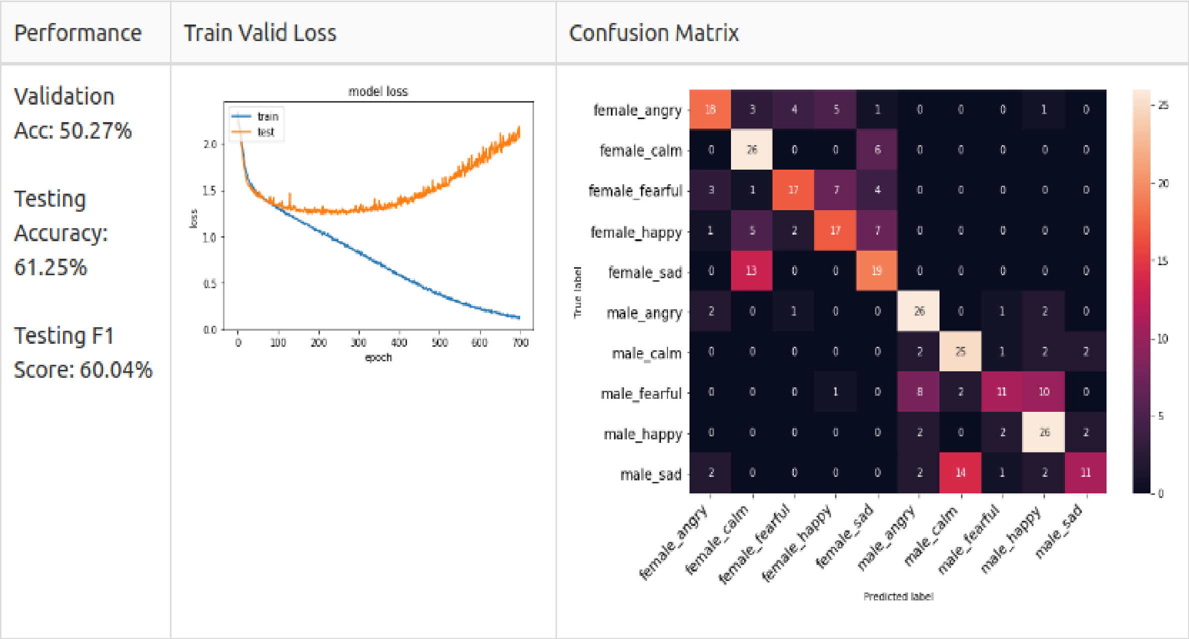
White box testing may be a testing during which the software package tester has information of the inner workings, structure and language of the software package, or a minimum of its purpose. it's accustomed check areas that can't be reached from a recording machine level.

7.5.2 BLACK BOX TESTING:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other forms of tests, should be written from a definitive supply document like specification or needed document. it's a checking during which the software package beneath test is treated, as a recording machine you can't see into it. The check provides inputs and responds to outputs while not considering however the software package works.

CHAPTER 8

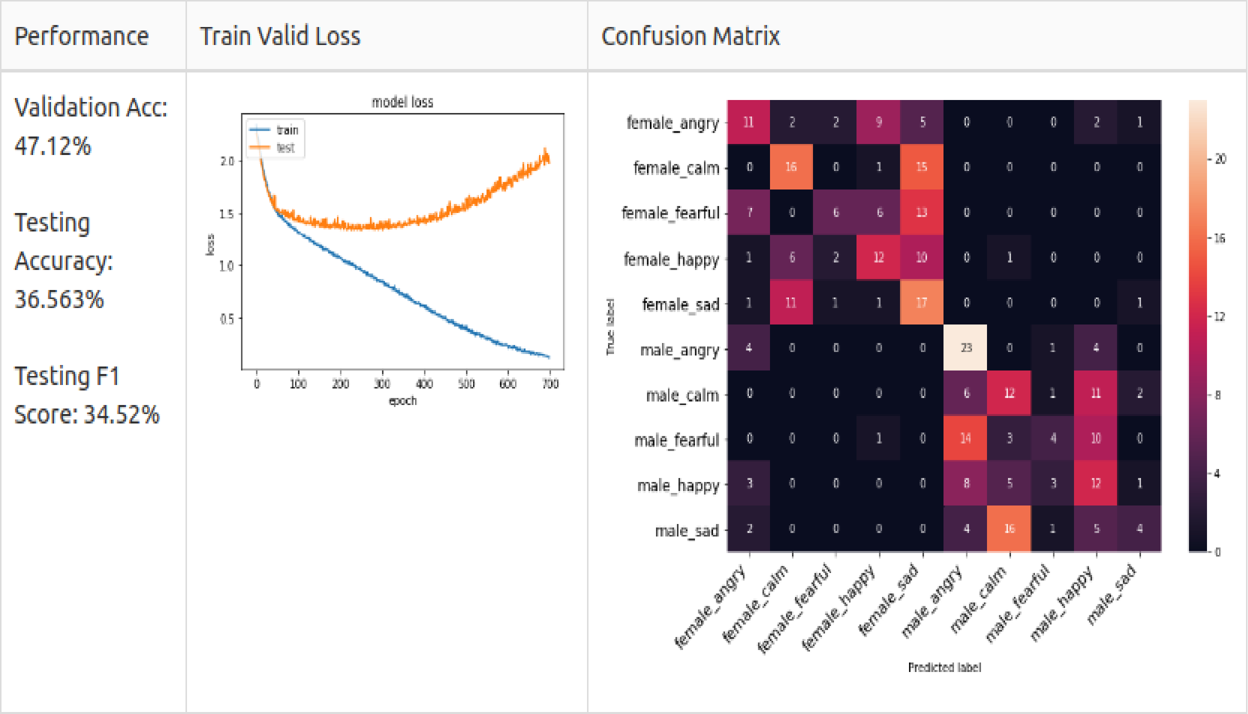
SCREENSHOTS



Performance	Train Valid Loss	Confusion Matrix
Validation Acc: 50.27% Testing Accuracy: 61.25% Testing F1 Score: 60.04%		

True label \ Predicted label	female_angry	female_calm	female_fearful	female_happy	female_sad	male_angry	male_calm	male_fearful	male_happy	male_sad
female_angry	18	3	4	5	1	0	0	0	1	0
female_calm	0	26	0	0	6	0	0	0	0	0
female_fearful	3	1	17	7	4	0	0	0	0	0
female_happy	1	5	2	17	7	0	0	0	0	0
female_sad	0	13	0	0	19	0	0	0	0	0
male_angry	2	0	1	0	0	26	0	1	2	0
male_calm	0	0	0	0	0	2	25	1	2	2
male_fearful	0	0	0	1	0	8	2	11	10	0
male_happy	0	0	0	0	0	2	0	2	26	2
male_sad	2	0	0	0	0	2	14	1	2	11

Figure 8.1 : Emotion Prediction for Female Version



Performance	Train Valid Loss	Confusion Matrix
Validation Acc: 47.12% Testing Accuracy: 36.563% Testing F1 Score: 34.52%		

True label \ Predicted label	female_angry	female_calm	female_fearful	female_happy	female_sad	male_angry	male_calm	male_fearful	male_happy	male_sad
female_angry	11	2	2	9	5	0	0	0	2	1
female_calm	0	16	0	1	15	0	0	0	0	0
female_fearful	7	0	6	6	13	0	0	0	0	0
female_happy	1	6	2	12	10	0	1	0	0	0
female_sad	1	11	1	1	17	0	0	0	0	1
male_angry	4	0	0	0	0	23	0	1	4	0
male_calm	0	0	0	0	0	6	12	1	11	2
male_fearful	0	0	0	1	0	14	3	4	10	0
male_happy	3	0	0	0	0	8	5	3	12	1
male_sad	2	0	0	0	0	4	16	1	5	4

Figure 8.2 : Emotion Prediction for Male Version

CHAPTER 9

APPENDIX - SOURCE CODE

```
from google.colab import drive
drive.mount('/content/drive')
!pip install soundfile

import soundfile
import numpy as np
import librosa
import glob
import os
from sklearn.model_selection import train_test_split

# all emotions on RAVDESS dataset
int2emotion = {
    "01": "neutral",
    "02": "calm",
    "03": "happy",
    "04": "sad",
    "05": "angry",
    "06": "fearful",
    "07": "disgust",
    "08": "surprised"
}
```

```
# we allow only these emotions
AVAILABLE_EMOTIONS = {
    "angry",
    "sad",
    "neutral",
    "happy"
}

def extract_feature(file_name, **kwargs):
    """
    Extract feature from audio file `file_name`
        Features supported:
            - MFCC (mfcc)
            - Chroma (chroma)
            - MEL Spectrogram Frequency (mel)
            - Contrast (contrast)
            - Tonnetz (tonnetz)
        e.g:
        `features = extract_feature(path, mel=True, mfcc=True)`
    """
    mfcc = kwargs.get("mfcc")
    chroma = kwargs.get("chroma")
    mel = kwargs.get("mel")
    contrast = kwargs.get("contrast")
    tonnetz = kwargs.get("tonnetz")
    with soundfile.SoundFile(file_name) as sound_file:
        X = sound_file.read(dtype="float32")
        sample_rate = sound_file.samplerate
        if chroma or contrast:
            stft = np.abs(librosa.stft(X))
        result = np.array([])
        if mfcc:
            mfccs = np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
            result = np.hstack((result, mfccs))
        if chroma:
            chroma = np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
            result = np.hstack((result, chroma))
        if mel:
            mel = np.mean(librosa.feature.melspectrogram(X, sr=sample_rate).T,axis=0)
            result = np.hstack((result, mel))
        if contrast:
            contrast = np.mean(librosa.feature.spectral_contrast(S=stft, sr=sample_rate).T,axis=0)
            result = np.hstack((result, contrast))
        if tonnetz:
            tonnetz = np.mean(librosa.feature.tonnetz(y=librosa.effects.harmonic(X), sr=sample_rate).T,axis=0)
            result = np.hstack((result, tonnetz))
    return result


def load_data(test_size=0.2):
    X, y = [], []
    try :
      for file in glob.glob("/content/Speech Audio-Dataset.zip"):
          # get the base name of the audio file
          basename = os.path.basename(file)
          print(basename)
          # get the emotion label
          emotion = int2emotion[basename.split("-")[2]]
          # we allow only AVAILABLE_EMOTIONS we set
          if emotion not in AVAILABLE_EMOTIONS:
              continue
          # extract speech features
          features = extract_feature(file, mfcc=True, chroma=True, mel=True)
          # add to data
          X.append(features)
          y.append(emotion)
    except :
        pass
    # split the data to training and testing and return it
    return train_test_split(np.array(X), y, test_size=test_size, random_state=7)
#@title Default title text
from sklearn.neural_network import MLPClassifier

from sklearn.metrics import accuracy_score


import os
import pickle


from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
```

```
print(classification_report(y_test,y_pred))
print(confusion_matrix(y_test,y_pred))
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
print(classification_report(y_test,y_pred))
print(confusion_matrix(y_test,y_pred))
```

```
              precision    recall  f1-score   support

       angry       0.94      0.86      0.90        90
       happy       0.81      0.80      0.80        94
     neutral       0.70      0.73      0.71        44
         sad       0.81      0.86      0.83       101

    accuracy                           0.82       329
   macro avg       0.81      0.81      0.81       329
weighted avg       0.83      0.82      0.82       329

[[77  9  2  2]
 [ 5 75  5  9]
 [ 0  2 32 10]
 [ 0  7  7 87]]
```

```
@title Default title text
X_train, X_test, y_train, y_test = load_data(test_size=0.25)
# print some details
# number of samples in training data
print("[+] Number of training samples:", X_train.shape[0])
```

```
# number of samples in testing data
print("[+] Number of testing samples:", X_test.shape[0])
# number of features used
# this is a vector of features extracted
# using utils.extract_features() method
print("[+] Number of features:", X_train.shape[1])
# best model, determined by a grid search
model_params = {
    'alpha': 0.01,
    'batch_size': 256,
    'epsilon': 1e-08,
    'hidden_layer_sizes': (300,),
    'learning_rate': 'adaptive',
    'max_iter': 500,
}
# initialize Multi Layer Perceptron classifier
# with best parameters ( so far )
model = MLPClassifier(**model_params)

# train the model
print("[*] Training the model...")
model.fit(X_train, y_train)

# predict 25% of data to measure how good we are
y_pred = model.predict(X_test)

# calculate the accuracy
accuracy = accuracy_score(y_true=y_test, y_pred=y_pred)
```

```
print("Accuracy: {:.2f}%".format(accuracy*100))

# now we save the model
# make result directory if doesn't exist yet
if not os.path.isdir("result"):
    os.mkdir("result")

pickle.dump(model, open("result/mlp_classifier.model", "wb"))

m_params = {
    'alpha': 0.01,
    'batch_size': 200,
    'epsilon': 1e-08,
    'hidden_layer_sizes': (300,),
    'learning_rate': 'adaptive',
    'max_iter': 500,
}
# initialize Multi Layer Perceptron classifier
# with best parameters ( so far )
m1 = MLPClassifier(**m_params)

# train the model
print("[*] Training the model...")
m1.fit(X_train, y_train)

# predict 25% of data to measure how good we are
y_p = m1.predict(X_test)
```

```
# calculate the accuracy
accuracy = accuracy_score(y_true=y_test, y_pred=y_p)

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

# now we save the model
# make result directory if doesn't exist yet

from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
print(classification_report(y_test,y_p))
print(confusion_matrix(y_test,y_p))

mn_params = {
    'alpha': 0.01,
    'batch_size': 256,
    'epsilon': 1e-08,
    'hidden_layer_sizes': (300,2),
    'learning_rate': 'adaptive',
    'max_iter': 500,
}
# initialize Multi Layer Perceptron classifier
# with best parameters ( so far )
m2 = MLPClassifier(**mn_params)



# train the model
```

```
print("[*] Training the model...")
m2.fit(X_train, y_train)

# predict 25% of data to measure how good we are
y_p2 = m2.predict(X_test)

# calculate the accuracy
accuracy = accuracy_score(y_true=y_test, y_pred=y_p2)

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

from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
print(classification_report(y_test,y_p2))
print(confusion_matrix(y_test,y_p2))

import os
import wave
import pickle
from sys import byteorder
from array import array
from struct import pack
from sklearn.neural_network import MLPClassifier #print("Please talk")
filename = "D:\project_emotion/test2.wav"# record the file (start talking)
    #record_to_file(filename)
    # extract features and reshape it
```

```
features = extract_feature(filename, mfcc=True, chroma=True, mel=True).reshape(1, -1)
   # predict
result = model.predict(features)[0]
   # show the result !
print("result:", result)
```

CHAPTER 10
CONCLUSION AND FUTURE ENHANCEMENT

10.1 CONCLUSION

This paper is predicated on speech emotion recognition using a support vector machine model. During this section, we summarize the datasets, methods, and approaches in SER. Here speech is the major source of human understanding so it is often utilized in major events for basic communication. This technology is employed in various fields of human-machine interaction. Here it is often utilized in online communication where machine understands human emotion and may ask human what emotion were human is. This proposed work of Emotion Recognition for input speech is based on feature extraction and classification techniques. Here we are extracting the MFCC feature from a human's speech signal for different emotional states. This system aims to solve the manual method problems of the existing system. This system is user-friendly and creates a bridge between machines and humans.

FUTURE ENHANCEMET

In the future, we increased the performance of this process and able to get more accuracy. The main implementation of our project is to recognize the emotions even on live ongoing calls.

CHAPTER 11
REFERENCES

[1] Dias Issa, M. Fatih Demirci, Adnan Yazici,Speech emotion recognition with deep convolutional neural networks,Biomedical Signal Processing and Control,Volume 59,2020,101894,ISSN 1746-8094.

[2] Jianhua Zhang, Zhong Yin, Peng Chen, Stefano Nichele, Emotion recognition using multi-modal data and machine learning techniques: A tutorial and review, Information Fusion, Volume 59,2020,Pages 103-126, ISSN 1566-2535.

[3] T.Sai Samhith, G.Nishika, M.Prayuktha, M.Bharat Chandra, Dr.Sunil Bhutada, G.Prasadu, Speech Emotion Recognition using Machine Learning Algorithms, Volume 9, Issue 6 June 2021, © 2021 IJCRT, ISSN: 2320-2882

[4] Leila Kerkeni, Youssef Serrestou, Mohamed Mbarki, Kosai Raoof, Mohamed Ali Mahjoub and Catherine Cleder, Automatic Speech Emotion Recognition Using Machine Learning, March 2019.