SAVITRIBAI PHULE PUNE UNIVERSITY

A PROJECT REPORT ON

Robust Speaker Recognition System for online authentication and real-time verification

SUBMITTED TOWARDS THE
PARTIAL FULFILLMENT OF THE REQUIREMENTS OF

BACHELOR OF ENGINEERING (Computer Engineering)

BY

Pranay Kundu	B120054397
Mridul Khanna	B120054350
Akshay Hiwale	B120054279
Aaditya Pandilwar	B120054376

Under The Guidance of

Prof. S. N. Girme

DEPARTMENT OF COMPUTER ENGINEERING
Pune Institute of Computer Technology
Dhankawadi, Pune-411043

Pune Institute of Computer Technology
DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the Project Entitled

Robust Speaker Recognition System for online authentication and real-time verification

Submitted by

Pranay Kundu	B120054397
Mridul Khanna	B120054350
Akshay Hiwale	B120054279
Aaditya Pandilwar	B120054376

is a bonafide work carried out by Students under the supervision of Prof. S. N. Girme and it is submitted towards the partial fulfillment of the requirement of Bachelor of Engineering (Computer Engineering).

Prof. S. N. Girme
Internal Guide
Dept. of Computer Engg.

Dr. R. B. Ingle
H.O.D
Dept. of Computer Engg.

Dr. P. T. Kulkarni
Principal
Pune Institute of Computer Technology

Signature of Internal Examiner

Signature of External Examiner

PROJECT APPROVAL SHEET

A Project Title

Robust Speaker Recognition System for online authentication and real-time verification

Is successfully completed by

Pranay Kundu	B120054397
Mridul Khanna	B120054350
Akshay Hiwale	B120054279
Aaditya Pandilwar	B120054376

at

DEPARTMENT OF COMPUTER ENGINEERING

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

SAVITRIBAI PHULE PUNE UNIVERSITY,PUNE

ACADEMIC YEAR 2017-2018

Prof. S. N. Girme	Dr. R. B. Ingle
Internal Guide	H.O.D
Dept. of Computer Engg.	Dept. of Computer Engg.

SPONSORSHIP SHEET

A Project Title

Robust Speaker Recognition System for online authentication and real-time verification

Is successfully completed by

Pranay Kundu	B120054397
Mridul Khanna	B120054350
Akshay Hiwale	B120054279
Aaditya Pandilwar	B120054376

which has been carried out under the open-source license. The project was guided by L3Cube mentors. The project is sponsored by **L3Cube**.

Mr. Omkar Patil Mentor, L3Cube	Mr. Shreyas Pansare Mentor, L3Cube	Mr. Niranjan Khetkar Mentor, L3Cube

Abstract

The robust speaker recognition systems are the extension of the field of research in speech recognition systems. The voice like fingerprint is an unique biometric. The problem statement focuses on the voice as authentication method. Voice can have upto 100 features which defines it uniquely.
The project targets to extract as many features as possible and address current challenges in existing models The system will build on Conventional Speaker Recognition System(CSR) and enhance the features using STRF extraction, LPCC extraction and implement robust system to maintain the SNR ratio for any real life environment application. The system will try to solve the current three challenges; Make existing models robust against noice in the audio input samples. Differ between the mimiced voice and original voice. Differentiate between the computer generated voice and human voice. We employ novel feature extraction techniques proposed in various literature and come up with new modified MFCC algorithm for voice feature extraction and robust training and verification system based on clustering in comparison to neural networks.
The main motivation towards this project are :
HSBC Widely popular Telephone Banking
Login mode for physically challenged
Satisfactory Customer Service cutting time and money in many BPOs

Acknowledgments

It gives us great pleasure in presenting the preliminary project report on ***'Robust Speaker Recognition System for online authentication and real-time verification'****.*

I would like to take this opportunity to thank my internal guide ***Prof. S. N. Girme*** *for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.*

I am also grateful to ***Dr. Rajesh Ingle****, Head of Computer Engineering Department, PICT, Pune for his indispensable support, suggestions.*

In the end our special thanks to ***Lab Assistants*** *for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for Our Project.*

Akshay Hiwale
Aaditya Pandilwar
Mridul Khanna
Pranay Kundu
(B.E. Computer Engg.)

Contents

List of Figures

List of Tables

CHAPTER 1
SYNOPSIS

1.1 Project Title

Robust Speaker Recognition System for online authentication and real-time verification.

1.2 Project Option

Industry Sponsored - Open Source.

1.3 Internal Guide

Prof. S. N. Girme

1.4 Sponsorship and External Guide

L3Cube Sponsorship - Mr. Omkar Patil, Mr. Shreyas Pansare, Mr. Niranjan Khetkar

1.5 Technical Keywords (As per ACM Keywords)

1. D.4.6 Security and Protection : Authentication
2. I.2.7 Natural Language Processing : Speech recognition and synthesis
3. H.5.5 Sound and Music Computing : Signal analysis, synthesis, and processing
4. I.5 Pattern Recognition

1.6 Problem Statement

Take the input voice sample in .wav file format and process it to generate MFCC(Mel Frequency Cepstral Coefficients) coefficients and use these coefficients to train and match the future input samples for authentication.

1.7 Abstract

The system will build on Conventional Speaker Recognition System(CSR) and enhance the features using STRF(Spectral Temporal Receptive Fields) extraction, LPCC(Linear Prediction Cestral Coefficients) extraction and implement robust system to maintain the SNR(Signal to Noise Ratio) ratio for any real life environment application. The system will try to solve the current three challenges; 1) Make existing models robust against noice in the audio input samples. 2) Differ between the mimiced voice and original voice. 3) Differentiate between the computer generated voice and human voice. We employ novel feature extraction techniques proposed in various literature and come up with new modified MFCC algorithm for voice feature extraction and robust training and verification system based on clustering in comparison to neural networks.

1.8 Goals and Objectives

- Achieve higher accuracy with noisy input samples
- Differentiate between mimic voice and original voice
- Differentiate between computer generated and true human voice

1.9 Relevant mathematics associated with the Project

Let S be the solution perspective of given problem statement.

$$S = \{s, e, X, D, Y, F, DD, NDD, Su, Fa\}$$

where,

s=start state
such that, $Y = \{\}$

e=end state;
such that, $Y = \{y_1\}$

y_1 = Authentication approved or denied.
X = set of input or input bitstream

such that $X = \{x_1, x_2, x_3, ..., x_n\}$

x_i = 16 bits or 1 sample
D = Data Store ;

such that $D = \{Z_1, Z_2\}$
where,
Z_1 = Label;
$Z_2 = \{< z_1, z_2, z_3, ... > \| mfcccoefficients\}$
Y = set of output ;

such that $Y = \{y_1, y_2\}$
y_1 = Authentication approved or denied.
y_2 = output coefficients of signal processor.
F = set of function ;

such that $F = \{f_1, f_2, f_3, f_4, f_5\}$
f_1 = function to accept WAV file bit stream
f_2 = function to process the signal and generate mfcc coefficients
f_3 = function to train the speaker model.
f_4 = function for pattern matching or recognition
f_5 = function to authenticate the pattern match results and give access or not

$f_1(X_i) = \{X = X_i\}$
f_2(X) = $\{y_2\}$
$f_3(y_2) = D$
$f_4(y_2)$ = temp
f_5(temp,D) = $\{y_1 | 0 or 1\}$

DD = Deterministic data.
$DD = |X|$

NDD = Non deterministic data
$NDD = \{y_1, y_2\}$

Su = Success
If the access is granted for a given voice input.

Fa = Failure
If the access is not granted and classified as false input or illegal input.

1.10 Names of Conferences / Journals where papers can be published

- ICASSP, the International Conference on Acoustics, Speech, and Signal Processing, 22nd April, 2018, Canada
- IEEE International Conference On Machine Learning And Applications (ICMLA), 17th December, 2018, Orlando, USA
- International Conference on Signal Processing Systems, 27th November, 2017, Auckland, New Zealand

1.11 Review of Conference/Journal Papers supporting Project idea

Dumopana et al [6] presents in this work, the inclusion of breath sounds in the training phase of the speaker recognition is analyzed using the popular Gaussian mixture model-universal background model (GMM-UBM) and deep neural network (DNN) based systems. It is shown that the DNN-based systems have a better learning capability to perform well even on unseen data compared to GMM-UBM-based systems. Verma et al [7] analyze resource utilization in terms of power consumption, memory and space requirements of three standard speaker recognition techniques, viz. GMM-UBM framework, Joint Factor Analysis and i-vectors. Experiments are performed on the MIT MDSVC corpus using the Energy Measurement Library (EML).

Chakraborty et al [10] Conventional MFCC algorithm for extraction of voice features was explained for implementation. Wang et al [5] STRF + MFCC for more features to extract than conventional MFCC and SVM for training and verification. Soleymanpour et al [4] Feature Extraction by MFCC, Clustering algorithm for sorting features of relevance, Neural Network as classifier and comparison with HMM and GMM models.

Wang et al [8] presents a speaker recognition method based on MFCC and Back-Propagation Neural Networks. Experimental studies have proven that the recognition rate is successful when the number of questionable speakers is not very larger. When the number of speakers increases, the rate of recognition decreases. Anand et al [9] presents a text-independent speaker recognition method particularly suitable for identification in AmI environments. The proposed method first computes the Mel Frequency Cepstral Coefficients (MFCC) and then creates Information Set Features (ISF) by

applying a fuzzy logic approach. Finally, it estimates the user's identity by using a hierarchical classification technique based on computational intelligence.

Nematollahi et al [2] says watermarking in speech signal using multiplicative technique along with DWPT(discrete wavelet packet transform) and threat analysis for online attacks on the basis of speaker. Omer [3] says the extracted speech features (MFCCs) of a speaker is quantized to a number of centroids using the LBG algorithm that clusters the feature vectors to constitute a codebook for each speaker in the system.

Luo et al, Three typical audio feature extraction algorithms, Mel Frequency Cepstrum Coefficients(MFCC), Spectrogram image features(SIF), Octave-Based Spectral Contrast, are chosen to parallelize. Choudhary [1] says Train the system with the GMM classifier using neural responses to save the classified data in the database (training); testing the data by using two types of speech signals(the original person and imposters)(testing). Kalaivani et al [11] he development of speaker identification system using perceptual linear prediction (PLP) for feature extraction and hidden Markov model (HMM) for speaker modelling.

1.12 Plan of Project Execution

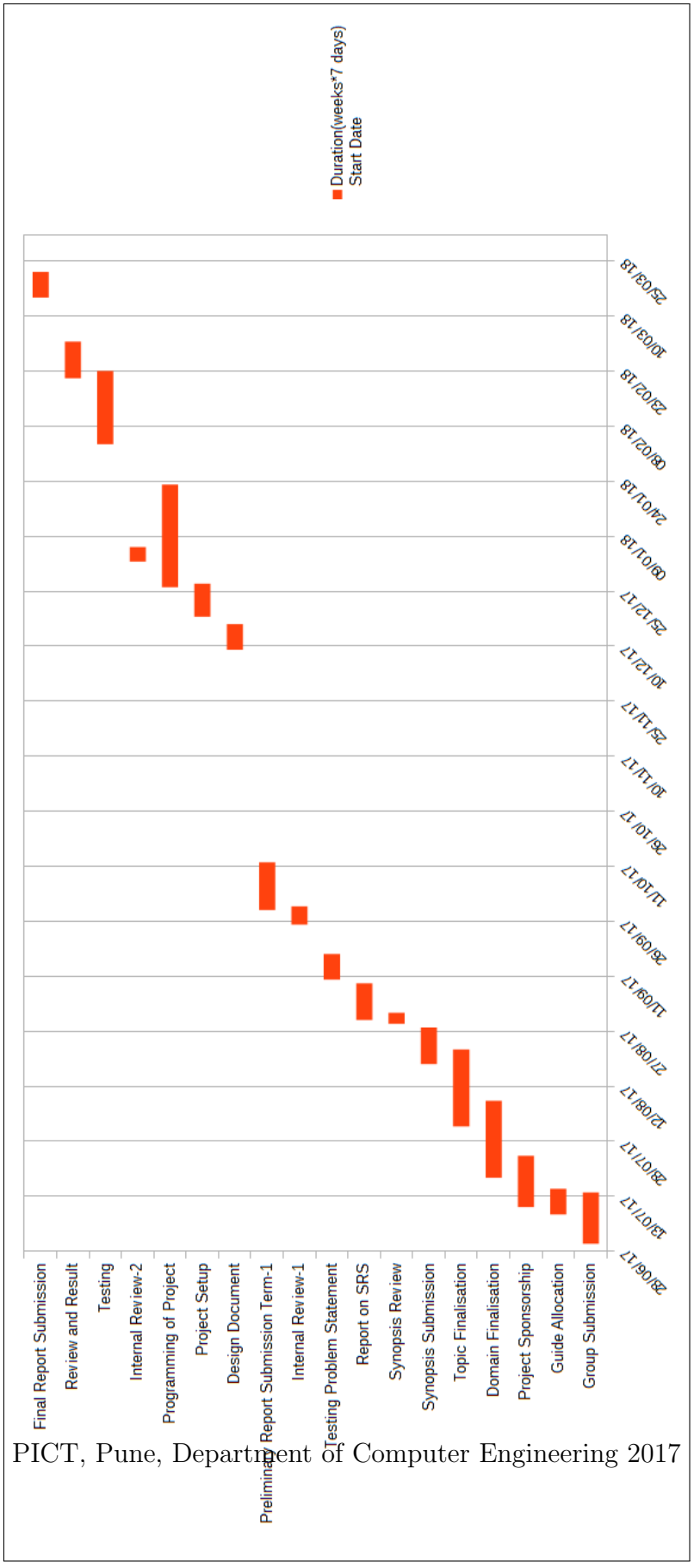


Figure 1.1: Project Plan

CHAPTER 2
TECHNICAL KEYWORDS

2.1 Area of Project

1. Research and Application in Speech Processing.
2. Signal Processing Systems.
3. Voice biometric and authentication systems.

2.2 Technical Keywords

1. Speech Processing Systems
   (a) Voice Recognition
      i. Voice
      ii. Biometric Technology
      iii. Feature extraction
      iv. FFT
      v. Authentication
      vi. MFCC
      vii. Discrete cosine transform
      viii. STRF
      ix. Vector Quantizatiion
   (b) Training Model
      i. Hidden Markov Model
      ii. Perceptual linear prediction
      iii. Spectrum
      iv. Speaker Identification rate
      v. Speaker Recognition
      vi. Pattern Recognition
      vii. Neural networks

2. Authentication Model

   (a) Multi Factor Authentication

      i. Speech Watermarking
      ii. Online Speaker Recognition
      iii. Discrete wavelet packet transform
      iv. Threat model
      v. Attack analysis

CHAPTER 3
INTRODUCTION

3.1 Project Idea

Authentication and security are the important factor for any system. Considering uniqueness of voice and using it as a authentication factor will greatly enhance the security of the system.

3.2 Motivation of the Project

- HSBC Widely popular Telephone Banking.
- Login mode for physically challenged
- Satisfactory Customer Service cutting time and money in many BPOs.

3.3 Literature Survey

Chakraborty et al [10] Conventional MFCC algorithm for extraction of voice features was explained for implementation. Wang et al [5] STRF + MFCC for more features to extract than conventional MFCC and SVM for training and verification. Soleymanpour et al [4] Feature Extraction by MFCC, Clustering algorithm for sorting features of relevance, Neural Network as classifier and comparison with HMM and GMM models.

Nematollahi et al [2] says watermarking in speech signal using multiplicative technique along with DWPT(discrete wavelet packet transform) and threat analysis for online attacks on the basis of speaker. Omer [3] says the extracted speech features (MFCCs) of a speaker is quantized to a number of centroids using the LBG algorithm that clusters the feature vectors to constitute a codebook for each speaker in the system.

Luo et al, Three typical audio feature extraction algorithms, Mel Frequency Cepstrum Coefficients(MFCC), Spectrogram image features(SIF), Octave-Based Spectral Contrast, are chosen to parallelize. Choudhary [1] says Train the system with the GMM classifier using neural responses to save the classified data in the database (training); testing the data by using two types of speech signals(the original person and imposters)(testing). Kalaivani et al [11] he development of speaker identification system using perceptual linear prediction (PLP) for feature extraction and hidden Markov model (HMM) for speaker modelling.

CHAPTER 4

PROBLEM DEFINITION AND SCOPE

4.1 Problem Statement

Take the input voice sample in .wav file format and process it to generate MFCC coefficients and use these coefficients to train and match the future input samples for authentication.

4.1.1 Goals and objectives

Goal and Objectives:

- Achieve higher accuracy with noisy input samples
- Differentiate between mimic voice and original voice
- Differentiate between computer generated and true human voice

4.1.2 Statement of scope

- Input: Voice sample in .wav file format
- Output: Generate MFCC coefficients and use of these coefficients to train and match the future input sample for authentication.
- The product would be able to differentiate between mimic voice and original voice and also between computer generated and human voice.
- It will contain the coefficients of each user's voice and this would further help in authentication.

4.2 Software context

- The software for speaker recognition system or the final product can be used in banking systems to verify the identity of telephone customers.
- It can also serve as a login mode for physically challenged.

4.3 Major Constraints

- The speaker recognition system is sensitive to noise or loud voices in the background; they may interfere with the user's voice and affect the recognition results.

- Settings for clear sound must be ensured; some audio software, hardware or drivers may have sound modification enabled by default. For example, the Microsoft Windows OS usually has, by default, sound boost enabled.
- A minimum 11025 Hz sampling rate, with at least 16-bit depth, should be used during voice recording.

4.4 Methodologies of Problem solving and efficiency issues

- Observation vector will be computed by k means clustering algorithm using the obtained features.
- Using the assumed initial state probability and the computed observation vector, HMM parameters like transition matrix (A) and emission matrix (B) are to be determined in the training phase
- The probabilities of match between test speaker and the speaker models are to be computed using the test voice signals observation vector and the HMM parameters (A and B) of the registered speakers, in the identification phase.

4.5 Scenario in which multi-core, Embedded and Distributed Computing used

The multi-core and distributing computing could be used to in deep neural networks and its deployment for faster processing. The distributed system as a whole will boost the capacity and efficiency and lessen the time of computation per hidden layer.

4.6 Outcome

- Achieve higher accuracy with noisy input samples.
- Differentiate between mimic voice and original voice.
- Differentiate between computer generated and true human voice.

4.7 Applications

- Banking systems to verify the identity of telephone customers.
- Login mode for physically challenged.
- Criminal investigations.

4.8 Hardware Resources Required

Sr. No.	Parameter	Minimum Requirement	Justification
1	CPU Speed	2.7 GHz	Multicore requirements
2	RAM	4 GB	Multicore requirements

Table 4.1: Hardware Requirements

4.9 Software Resources Required

Platform :

1. Operating System: Ubuntu 16.04
2. IDE: Keras Tensorflow
3. Programming Language: Python

CHAPTER 5
PROJECT PLAN

5.1 Project Estimates

Waterfall model will be used in the Speaker Recognition project. The information about the reconciled estimates is described below. The project resources are identified in the section after estimates.

5.1.1 Reconciled Estimates

5.1.1.1 Cost Estimate

Description	Quantity	Cost(Rs.)
NVidia GeForce GT 720	1	2748
Voice Corpus Dataset	1	2000
		total=4748

Table 5.1: Reconciled Cost Estimates

5.1.1.2 Time Estimates

Phase/Task	Estimated Time(Days)
Initiation	15
Definition	10
Realization/Implementation	60
Testing/Bug Fixing	20
Further Research	30
	total = 135

Table 5.2: Reconciled Time Estimates

5.1.2 Project Resources

The available resources are divided into following:

1. People
   (a) Omkar Patil - Project Manager
   (b) Pranay Kundu - Developer
   (c) Mridul Khanna - Developer
   (d) Akshay Hiwale - Developer
   (e) Aaditya Pandilwar - Developer
2. Hardware
   (a) NVidia GeForce GT 270
   (b) Intel Core i5 Processor
   (c) Microsoft Azure Virtualization
3. Software
   (a) Python, Jython, Octave
   (b) TensorFlow module
   (c) Keras module
   (d) Flask
   (e) python_speech_features module

5.2 Risk Management w.r.t. NP Hard analysis

This section discusses risks involved in Speaker Recognition project and the approaches to managing them.

5.2.1 Risk Identification

For risks identification, review of scope document, requirements specifications and schedule is done. Answers to questionnaire revealed some risks. Below, we mention the risks identified. Please refer table 5.3 for all the risks' description.

1. The project being more inclined towards research, a product might not emerge as a result of lack of research findings in the stipulated time.
2. The unaccounted noise might affect the quality of the product.
3. Wrong time estimation.
4. Continuous changing requirements.

5.2.2 Risk Analysis

The risks for the Project can be analyzed within the constraints of time and quality

ID	Risk Description	Probability	Impact		
			Schedule	Quality	Overall
1	The project being more inclined towards research, a product might not emerge as a result of lack of research findings in the stipulated time.	High	High	Low	High
2	The unaccounted noise might affect the quality of the product.	High	Low	Medium	Medium
3	Wrong time estimation.	Medium	Low	Low	Low
4	Continuous changing requirements.	Low	Low	High	High

Table 5.3: Risk Table

5.2.3 Overview of Risk Mitigation, Monitoring, Management

Following are the details for each risk.

Probability	Value	Description
High	Probability of occurrence is	$> 75\%$
Medium	Probability of occurrence is	$26 - 75\%$
Low	Probability of occurrence is	$< 25\%$

Table 5.4: Risk Probability definitions [**?**]

Impact	Value	Description
Very high	$> 10\%$	Schedule impact or Unacceptable quality
High	$5 - 10\%$	Schedule impact or Some parts of the project have low quality
Medium	$< 5\%$	Schedule impact or Barely noticeable degradation in quality Low Impact on schedule or Quality can be incorporated

Table 5.5: Risk Impact definitions [**?**]

Risk ID	1
Risk Description	The project being more inclined towards research, a product might not emerge as a result of lack of research findings in the stipulated time.
Category	Schedule Related Risk
Source	Software Requirement Specification document.
Probability	High
Impact	High
Response	Mitigate
Strategy	1. Thorough literature survey to know the nature of limitations faced by other researchers. 2. Rapid implementations of previous works, more focus on new research.
Risk Status	Identified

Risk ID	2
Risk Description	The unaccounted noise might affect the quality of the product.
Category	Technical Risk
Source	Software Design Specification documentation review.
Probability	High
Impact	Medium
Response	Mitigate
Strategy	1. Incorporate noise as variable as possible in the training datasets. 2. Better testing.
Risk Status	Identified

Risk ID	3
Risk Description	Wrong time estimation.
Category	Schedule Related Risk.
Source	Early development stages
Probability	Medium
Impact	Low
Response	Avoid
Strategy	Regular tracking of work progress and adapting to changes accordingly.
Risk Status	Identified

Risk ID	4
Risk Description	Continuous changing requirements.
Category	Requirements
Source	Software Requirements Specification document.
Probability	Low
Impact	High
Response	Mitigate
Strategy	Continuous surveys to keep informed about changing requirements.
Risk Status	Identified

5.3 Project Schedule

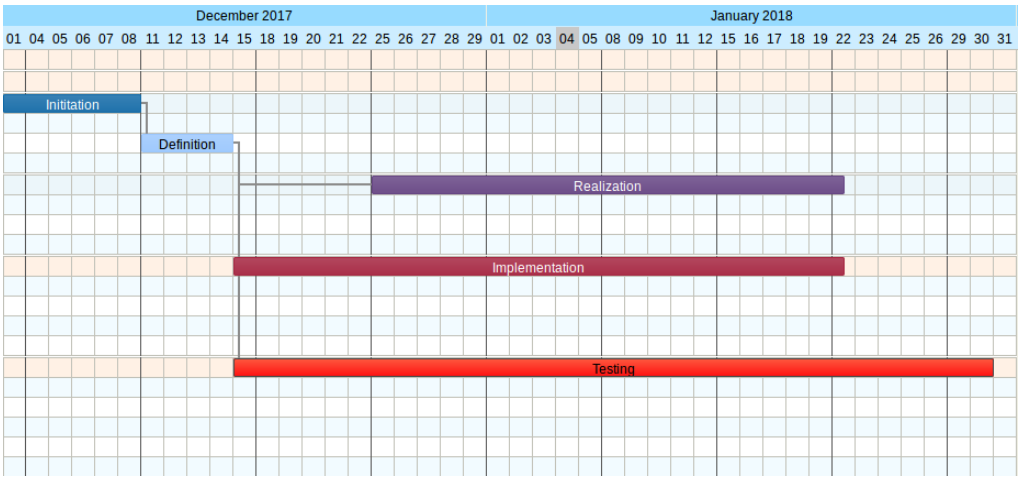
5.3.1 Project task set

Major Tasks in the Project stages are:

- Task 1: Initiation
- Task 2: Definition
- Task 3: Realization
- Task 4: Implementation
- Task 5: Testing

5.3.2 Task network

Project tasks and their dependencies are noted in this diagrammatic form.



5.3.3 Timeline Chart

A project timeline chart is presented. This may include a time line for the entire project. Above points should be covered in Project Planner as Annex C and you can mention here Please refer Annex C for the planner C

5.4 Team Organization

The manner in which staff is organized and the mechanisms for reporting are noted.

5.4.1 Team structure

We choose Mixed Control Team structure as we have limited number of project members. We use the project format instead of functional format because of this scarcity.

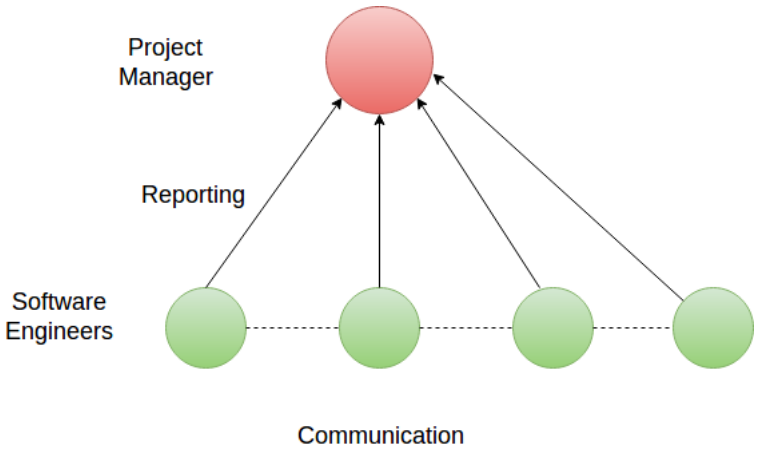
Roles of the team members are as follows:

1. Project Manager: The person who is responsible for controlling the progress of the project and coordinate other team members.

2. Software Engineer/Developer: The person who is responsible for carrying out the SDLC activities and developing the software alongside.

5.4.2 Management reporting and communication

As we have chosen the Mixed Control Team structure, the Project Manager is going to supervise the software developers personally.
The software engineers/developers would report to the project manager directly.
The software developers can also communicate between themselves for better coordination.
The structure is shown in the diagram below.



CHAPTER 6

SOFTWARE REQUIREMENT SPECIFICATION (SRS IS TO BE PREPARED USING RELEVANT MATHEMATICS DERIVED AND SOFTWARE ENGG. INDICATORS IN ANNEX A AND B)

6.1 Introduction

6.1.1 Purpose and Scope of Document

The purpose of this Software Requirements Specification is to outline the requirements of the Speaker Recognition API. This API will be built using Python, Java, Keras libraray, TensofFlow library. It will be importable from Python modules or can be used directly through our application.
The API would be used by Customer Experience companies or Banking sector companies or developers interested in using this functionality in their own applications for seamless authentication and recognition of their customers.

6.1.2 Overview of responsibilities of Developer

The activities carried out by the developers for the Speaker Recogniton project are:

1. Feasibility survey
2. Requirements Gathering
3. Software Requirements Validation
4. Requirements Elicitation

6.2 Usage Scenario

This section provides various usage scenarios for the Speaker Recognition API. It outlines the various types of users who will use the product and describe the ways in which they will interact with the product.

6.2.1 User profiles

The profiles of all user categories are described below.(Actors and their Description).

6.2.2 Use-cases

All use cases for the sortware are presented. Descrption of all main use cases are also provided in the table below.

Sr No.	Actor	Description
1	Python Developer	This actor will interact with the API directly through his program and make use of the API functionality.
2	Customer	This actor is the customer who wants to authenticate himself using the API, through our application software.

Table 6.1: User Profiles

6.2.3 Use Case View

Below, we show the use case diagram for Speaker Recognition API.

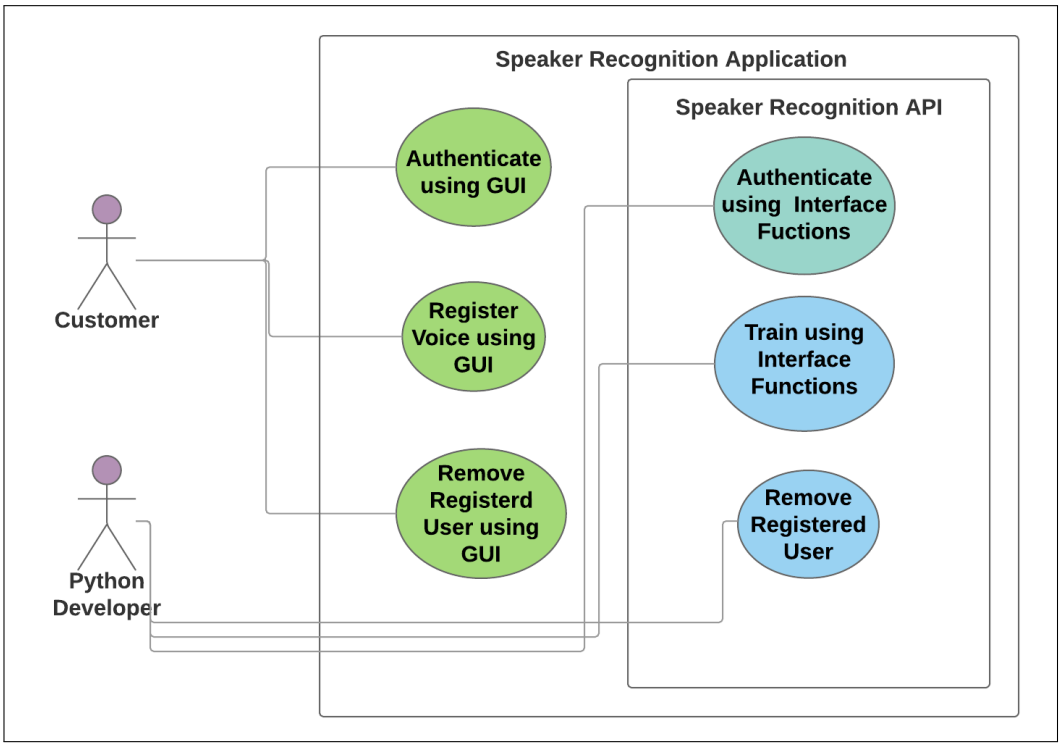


Figure 6.1: Use Case Diagram

Sr No.	Use Case	Description	Actors
1	Use of the API through interface functions for training	In this use case, the actor will use the API through the interface functions provided by the API for training	Python Developer
2	Use of the API through interface functions for authentication	In this use case, the actor will use the API through the interface functions provided by the API for authentication	Python Developer
3	Use of the API application for registering	In this use case, the actor will directly use the functioinality using a GUI provided by the application for registering his voice.	Customer
4	Use of the API application for authentication	In this use case, the actor will directly use the functioinality using a GUI provided by the application for authentication.	Customer
5	Use of the API application for User Removal	In this use case, the actor will directly use the functioinality using a GUI provided by the application for removing a registered user.	Customer
6	Use of the API directly for User Removal	In this use case, the actor will directly use the functioinality provided by the API for removing a registered user.	Python Developer

Table 6.2: Use Cases

6.3 Data Model and Description

6.3.1 Data Description

Speaker Recognition system being proposed will be using the following data structures and database.

1. **Database:**
   Only one central database will be used to store the authentic users in the system. It will have two tables. The structure of the database will be as shown below.

Field	**Type**	**Null**	**Default**
User_ID	int(11)	No	-
$feature_1$	float(3,7)	No	0
$feature_2$	float(3,7)	No	0
$feature_3$	float(3,7)	No	0
$feature_n$	float(3,7)	No	0

Table 6.3: users_features

Field	**Type**	**Null**	**Default**
User_ID	int(11)	No	-
First_name	varchar(50)	No	-
Last_name	varchar(50)	Yes	-
Privilege	int(11)	Yes	1
Gender	tinyint(1)	Yes	1

Table 6.4: user_description

2. **Data structures:**

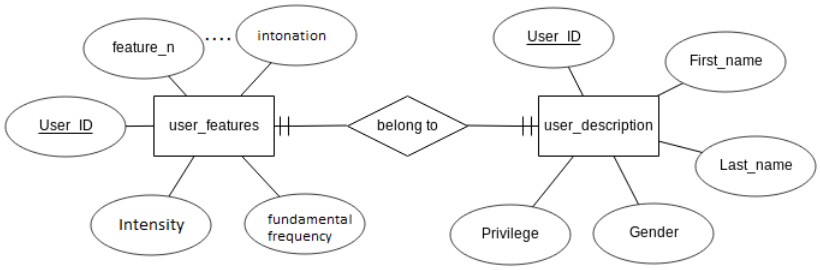
   (a) **Global Data Structures:** We will be using a 1-Dimensional feature coefficient vector as the global data structure which can be

implemented using a 1-Dimensional array of floating point variables. Similar array will be used for extracted feature vectors.

(b) **Temporary Data Structures:** These data structures would be the ones generated for a short period of time after which they will be destroyed. These include JSON objects and the WAV files generated.

6.3.2 Data objects and Relationships

Here we show the database relationships using an ERD i.e. Entity Relations Diagram. It is shown as below.



6.4 Functional Model and Description

A description of each major software function of the Speaker Recognition system is presented along with data flow diagrams as a way of structured analysis. DFDs for major functions are shown below.

6.4.1 Input Processing:

6.4.2 Identification/Verification:

6.4.3 Data Flow Diagram

6.4.3.1 Level 0 Data Flow Diagram

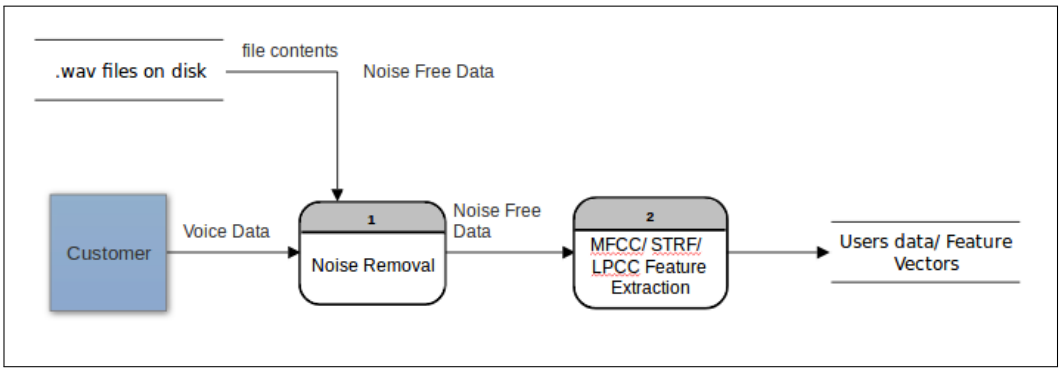


Figure 6.2: Functional Model-1

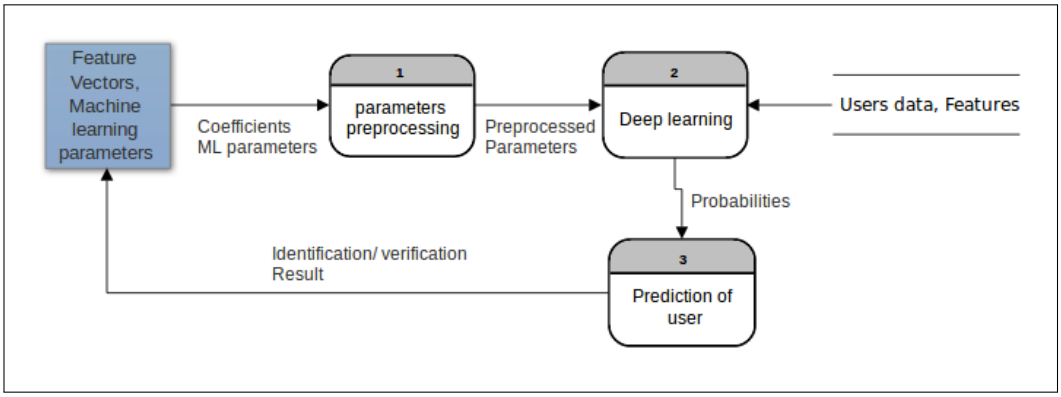


Figure 6.3: Functional Model-2

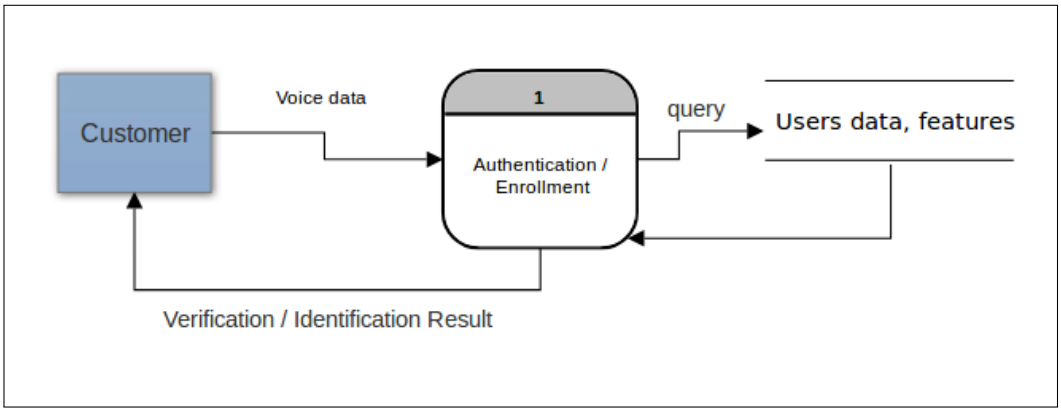


Figure 6.4: Level-0 Data Flow

6.4.3.2 Level 1 Data Flow Diagram

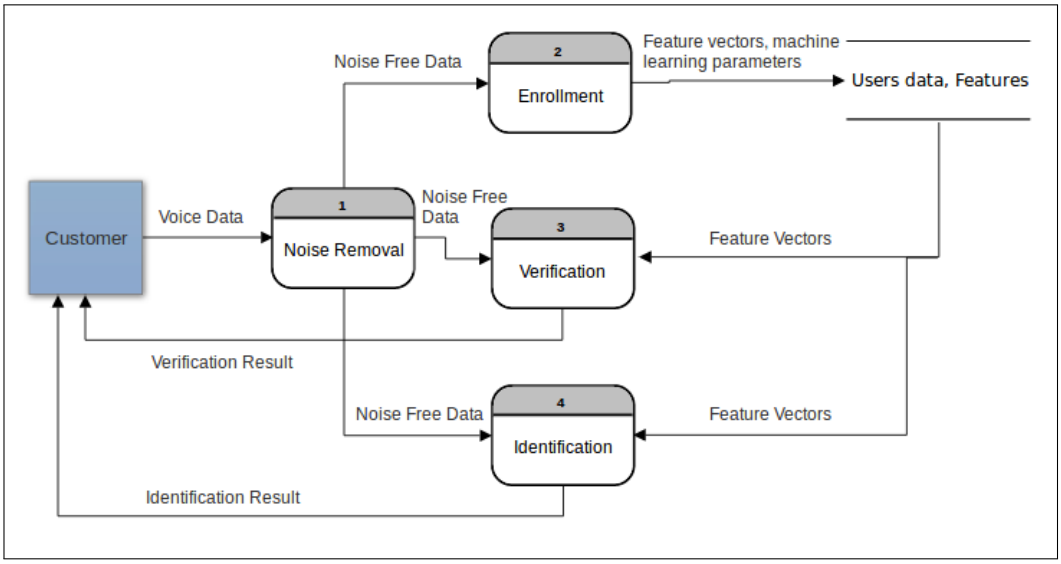


Figure 6.5: Level-1 Data Flow

6.4.4 Description of functions

A description of each software function in the Speaker Recognition is presented.

1. **Input Processing:**
   The input processing module is responsible for the input fetching and noise removal in the voice sample. The input may come directly from the user or from a disk file. This function is also responsible for silence removal in the voice data.

2. **Identification/ Verification:**
   This function is responsible for giving the core functionality of the API. It receives its input from the previous input processing module and the database stored on the system for stored user data and feature vectors. It makes use of deep learning techniques to carry out the process and predicts the appropriate result.

6.4.5 Activity Diagram:

- The Activity diagram represents the steps taken.

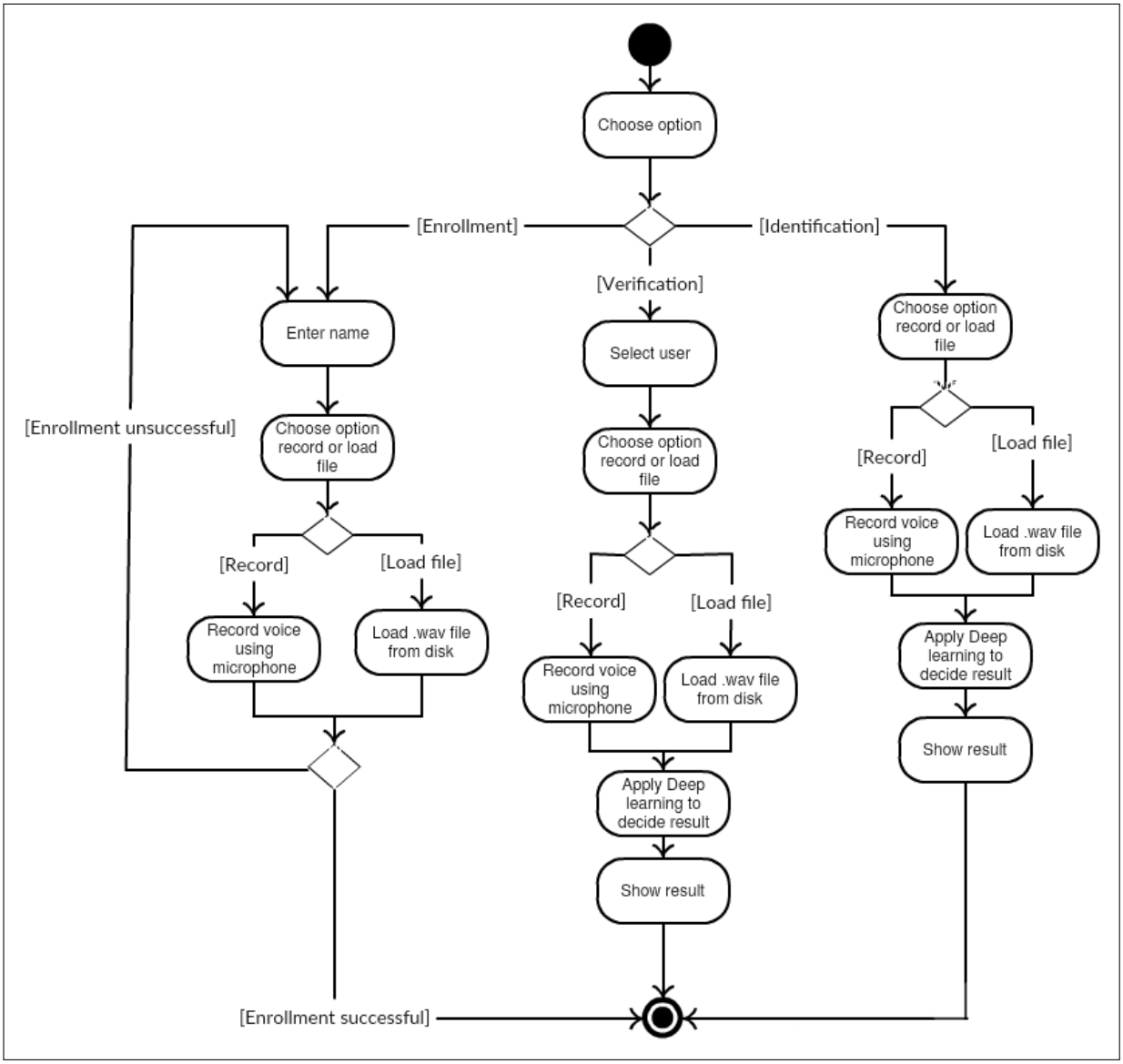


Figure 6.6: Activity diagram

6.4.6 Non Functional Requirements:

- **Interface Requirements:**
  The Speaker Recognition API requires some libraries to be installed on the system. The interface to the library the API gets is through its functions. The libraries which are needed are Keras,TensorFlow,

SciKitLearn and numpy.

- **Performance Requirements:**

  1. The enrollment of the user should take a minimum of 10 samples of 5 second voice sample.
  2. The real time verification and identification can be done in a single click.

- **Availability:**
  System availability depends on the Internet connection if the API is being used over REST API calls, and the database connectivity.

- **Modifiability:**
  Most of the code written will be in python3 for making the API future proof and easily modifiable. All the coding conventions will be followed strictly.

- **Testability:**
  Testing scripts will be written for easy evaluation of the code and testing with various input conditions.

6.4.7 State Diagram:

State Transition Diagram
Fig.6.7 example shows the state transition diagram of Cloud SDK. The states are represented in ovals and state of system gets changed when certain events occur. The transitions from one state to the other are represented by arrows. The Figure shows important states and events that occur while creating new project.

6.4.8 Design Constraints

The design constraints that will impact the Speaker Recognition subsystem are mentioned below.

- **Hard Drive space:**
  For the API module to be installed, no more than 30 MB space on the disk is required.
  Database space requirement will depend on the number of enrolled users.

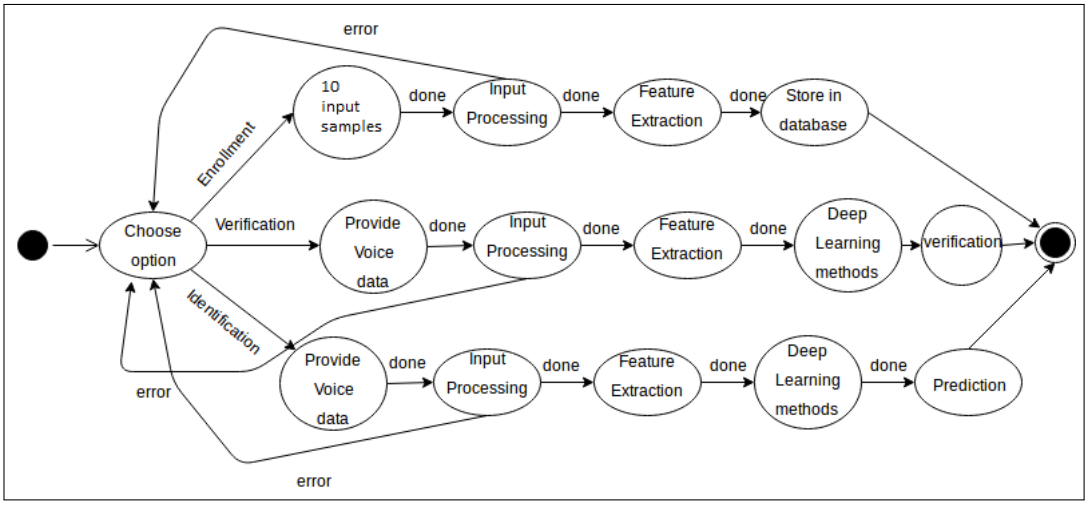


Figure 6.7: State transition diagram

- **Memory Usage:**
  Main memory usage of the API will be no more than 20 MB when doing real time analysis.
  It will be no more than 10 MB for other operations.

6.4.9 Software Interface Description

The software interfaces to the outside world are described in this section. The requirements for interfaces for the Speaker Recognition system are stated below.

- The API will communicate with external applications through the use of interface functions once it is imported in code.
- The API will use systemcalls and system functions for system services such as disk access and microphone access.
- The API along with the application will communicate with the users through the use of GUI that will be provided by the application itself.

CHAPTER 7

DETAILED DESIGN DOCUMENT USING APPENDIX A AND B

7.1 Introduction

The Speaker Recognition system in consideration will be running majorly on three modules : **Input processor** - to generate the required features from the voice sample, **Training Module** - to train the system for a particular speaker, **Recognizer** - to identify the speaker of incoming input sample.

7.2 Architectural Design

The architecture mainly consists of the **api** and **application layer**.

1. **API** : We propose the API as the final product which will take the input from application layer. The API consists of various modules such as Input Processor, Authentication Module, Deep Learning Module and Database Module. The brief description of each module is as follows:

    - **FAIL OVER** : The module takes care if any module crashes or fails to process the input and generate required output.
    - **INPUT PROCESSOR** : The input received from application layer, is processed to remove noise and generate features for training and pattern recognition.
    - **DATABASE MODULE** : The module is responsible for the data fetch and store in the form of audio or labelled coefficients.
    - **DEEP LEARNING MODULE** : The deep learning module consists of **trainer** and **recognizer** which train and pattern matching on input samples.
    - **CURRENT DATA BACKUP** : The current data for the processing will be held in the backup.

2. **Application Layer** : The **Application Layer** is involved in accepting input and processing and converting to required format for the API.

    - **User Interface** : The buttons and functions used to accept input and other features of application to increase its usability.
    - **Input Fetcher** : The input received is processed and converted to required format.

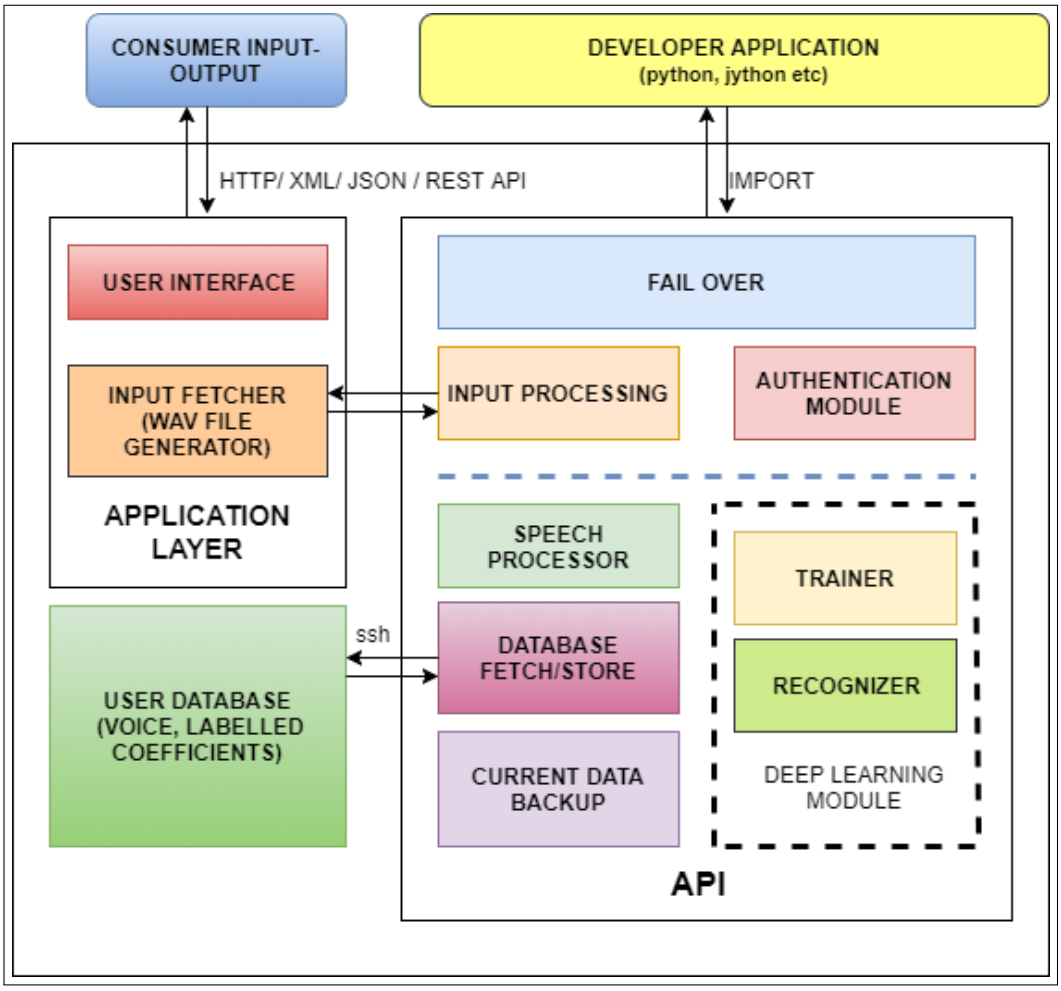


Figure 7.1: Architecture diagram

7.3 Data design (using Appendices A and B)

The structure of data can be viewed at three levels, namely, program component level, application level, and business level. At the program component level, the design of data structures and the algorithms required to manipulate them is necessary, if high-quality software is desired. At the application level, it is crucial to convert the data model into a database so that the specific business objectives of a system could be achieved. At the business level, the collection of information stored in different databases should be reorganized into data warehouse, which enables data mining that has an influential impact on the business.

7.3.1 Internal software data structure

Neural network being used for the development of the application is implemented by vectors only, where each layer with multiple neurons is multidimensional vectors.

7.3.2 Global data structure

Vectors containing the input coefficients will be globally available for all the modules to be used. The vector list will be used for generating output at various stages of the deep learning module.

7.3.3 Temporary data structure

Hashmap or Hashtable will be used for the data processing before storing and populate the hashtables in further stages of processing. The hashtables will be used to store the MFCC coefficients generated and will be stored as key-value pair where the key is the labelled speaker for given set of coefficients in training phase.

7.3.4 Database description

Database or Datasets will be comprised of audio files(.wav files) and labelled coefficients. (.wav) files can be processed as a byte stream of signal features. WAV audio format is uncompressed audio in the linear pulse code modulation (LPCM) format. Since LPCM is uncompressed and retains all of the samples of an audio track, professional users or audio experts may use the WAV format with LPCM audio for maximum audio quality.

7.4 Component Design

The complete architecture described above can be broken down into various components which represent the main architecture. We describe our component design in terms of class diagram. **SpeakerRec** is the parent class of all the modules where **coeff** is the global data structure described above.

7.4.1 Class Diagram

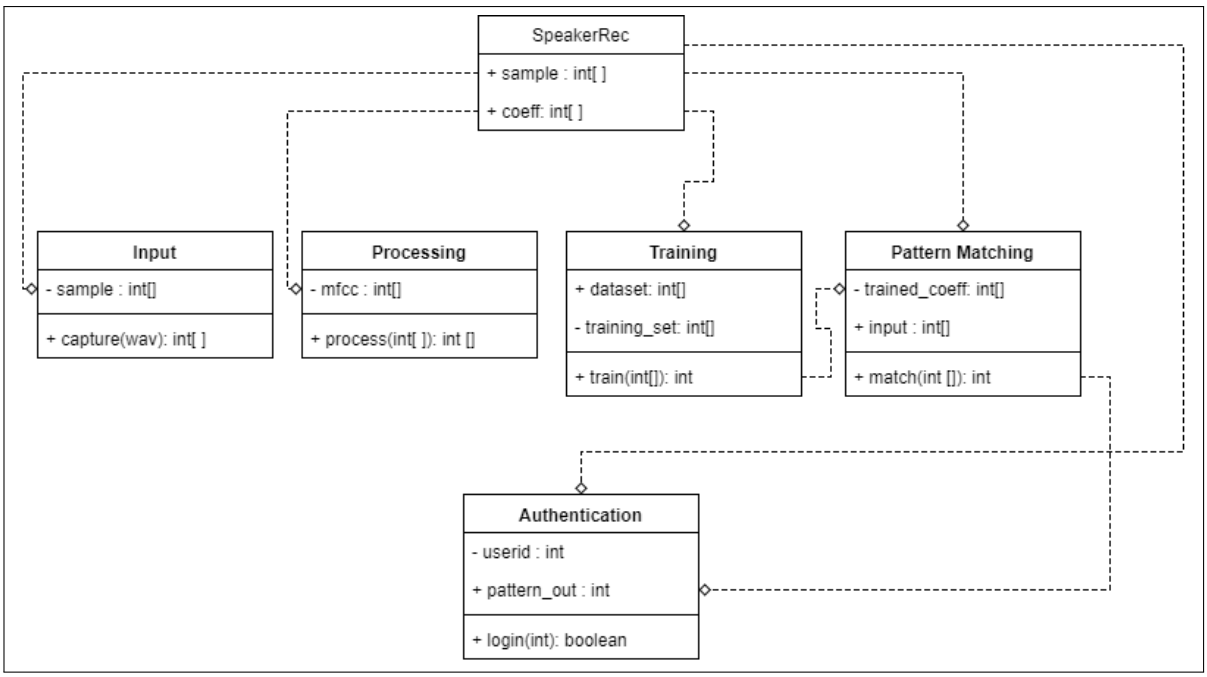


Figure 7.2: Class Diagram

CHAPTER 8
PROJECT IMPLEMENTATION

8.1 Introduction

Whole Speaker Recognition project is built using Python 2.7 as the only programming language along with minor bash scripting. The system is divided in mainly two parts:

- Graphical User Interface
- Core Speaker Recognition Module

The Speaker Recognition system houses all of its current functionality locally without any need of any web or cloud infrastructure. However, it can later extend as a deployment on cloud services making it ideal for web based access. The software is presented as a Python application with GUI which can be used for the whole end-to-end usage process involving the following steps:

1. Enrollment/Registration of new users with voice
2. Verification by Voice
3. Identification by Voice

8.1.1 Graphical User Interface

The GUI is divided into three tabs namely,

- Registration
- Verification/Identification
- Conversation Mode

The Registration tab provides interface for new users to give their profile details and provide their voice by recording or through a stored file.

The Verification/Identification tab gives the functionality of the core speaker recognition task, A user can claim his/her identity and verify oneself using voice. Identification part requires user to only provide voice, the system determines on its own whether the user is among stored speakers or not and tell who the user is among stored users, if yes.

The conversation mode tab can be used when a conference call or conversation between the registered speaker is to be monitored for identifying who is speaking currently. Real time processing of the audio is done in this mode. The conversation can be directly recorded through the application.

8.1.2 Core Speaker Recognition Module

The Speaker Recognition functionality uses deep learning techniques. The registration process of a new speaker and identification/verification and the conversation mode, all of the functionality is implemented using these methods.

A 256 dimensional speaker representation is created by a forward pass through the neural network, at the time of speaker enrollment/registration. This representation along with other user profile data is stored in a local database containing all the users' information.

At the time of verification/identification this stored representation is used to compare with the representation generated by the provided voice. The two representations are compared using a cosine similarity measure. A threshold value is used to draw a line between positive and negative predictions.

8.2 Tools and Technologies Used

The tools and external libraries that this project uses are listed below in the following sections:

- GUI Module
- Core Speaker Recognition Module

8.2.1 GUI Module Tools

1. **PyQt5** : Used for the widgets and tools for building GUI
2. **pyAudio, sounddevice** : Used for recording and processing audio from device
3. **librosa, wave** : Used for handling wave files

8.2.2 Core Speaker Recognition Module Tools

1. **python_speech_features** : Used for generating DSP features from audio data like MFCC, filterbank frequencies
2. **keras, tensorflow** : Used for Neural Network implementation in the deep learning module

3. **sci-kitlearn** : Used for basic machine learning techniques implementations like J48 Decision trees
4. **numpy, pandas** : Used for handling matrices and numeric data and some helpful datastructures
5. **sox, pysndfx** : Used for initial preprocessing of raw audio files and noise cancellation and silence removal
6. **librosa** : Used for generating spectrogram and other visualizations from audio data
7. **CuDNN** : Used for parallelizing the deep learning training algorithms over NVidia GPU
8. **matplotlib** : Used for visualizing image(2D) data

8.3 Methodologies/Algorithm Details

The algorithms used in the Speaker Recognition project while training and verification/identification process are described below in terms of pseudo code:

8.3.1 Identification/Verification/Algorithm

1. Choose Identification/Verification.
2. if choice == Identification goto step 3 else goto 10
3. Take voice input or disk file of 5 sec voice
4. Perform noise cancellation
5. Perform silence removal, amplification
6. Generate representation from voice
7. Compare representation with all stored users using cosine similarity.
8. if maxsimilarity > threshold, output Username else output no match
9. goto step 17
10. Choose one user from enrolled users/speakers.

11. Take voice input or disk file of 5 sec voice.
12. Perform noise cancellation
13. Perform silence removal, amplification
14. Generate representation from voice
15. Compare representation with chosen speaker using cosine similarity.
16. if similarity > threshold, output Username else output mismatch
17. Exit.

8.3.2 Softmax Pre-training/Pseudo Code

```
void Pretrain(nbepoch,datadir):

    // dataset reading
    data := ReadDirectory(datadir)
    // data is a datastructure containing all the
    // audio filenames in datadir
    speakers := data["speaker-id"].unique()
    nbspeakers := len(speakers)

    // model generation
    spkrep := models.SpeakerRepresentation()
    // generate model from models class
    spkrep.add(softmax(nbspeakers))
    spkrep := Model(spkrep)
    spkrep.compile('categorical_crossentropy','optimizer','metrics')

    [xtrain,xtest] := train_test_split(data)
    [ytest,ytest] := get_outputs(xtrain,xtest)

    for epoch in nbepoch:
        nbbatches := len(xtrain)/batchsize

        for batch in nbbatches:
            batch := next_batch(xtrain)
            batchinput := readdata(batch)
            batchoutput := onehotencode(batch)
            spkrep.train_on_batch(batchinput,batchoutput)
```

```
    nbbatches := len(xtest)/batchsize

    batch := next_batch(xtest)
    batchinput := readdata(batch)
    batchoutput := onehotencode(batch)
    acc = spkrep.test_on_batch(batchinput,batchoutput)
    logging.info(acc+epoch)

    spkrep.save_weights("weight_"+epoch+".h5")
```

8.3.3 Training Siamese Network/Pseudo Code

```
void SiameseTrain(nbepoch,datadir):

    // dataset reading
    data := ReadDirectory(datadir)
    // data is a datastructure containing all the
    // audio filenames in datadir

    // model generation
    spkrep := models.SpeakerRepresentation()
    // generate model from models class
    input1 := inputtensor()
    input2 := inputtensor()
    spk1 := spkrep(input1)
    spk2 := spkrep(input2)
    lastlayer := cosine_similarity(spk1,spk2)
    spkrep := Model(spk1,spk2)
    spkrep.compile('binary_crossentropy','optimizer','metrics')

    [xtrain,xtest] := train_test_split(data)

    xtrain = get_pairs(xtrain)
    xtest = get_pairs(xtest)

    for epoch in nbepoch:
        nbbatches := len(xtrain)/batchsize
```

```
for batch in nbbatches:
    batch := next_batch(xtrain)
    batchinput := readdata(batch)
    batchoutput := get_output(batch)
    spkrep.train_on_batch(batchinput,batchoutput)

nbbatches := len(xtest)/batchsize

batch := next_batch(xtest)
batchinput := readdata(batch)
batchoutput := get_output(batch)
acc = spkrep.test_on_batch(batchinput,batchoutput)
logging.info(acc+epoch)

spkrep.save_weights("weight_"+epoch+".h5")
```

8.4 Verification and Validation for Acceptance

Following are the measures we took to ensure the quality of the software through verification and validation.

8.4.1 Verification

Verification methods ensure that the development of the software is going into the right direction so as to fulfil the requirements specified in SRS of the software. The verification measures deal with the theoretical aspects of the development and they do not involve the actual software built after development. Verification includes all the activities associated with producing high quality software: testing, inspection, design analysis, specification analysis, and so on. We took the following steps as Verification of the development process:

1. Ensured that the user registration time will not take more than 500 msec, we ensured it by having a representation based approach in the design.
2. Ensured that the effects of ambient noise and silence will least affect the system's accuracy by using noise and silence removal methods.
3. Ensured that the conversation mode will be real time by using a different frame level classifier instead of utterance level processing.

8.4.2 Validation

Validation is an extremely subjective process. It involves making subjective assessments of how well the (proposed) system addresses a real-world need. It involves testing the actual software or the prototype to see how well it is modelling its specified requirements. We performed different types of testing methods to ensure the system's performance. Below are the actions taken for validation of the software:

1. Performed positive testing on the software with relatively easy test cases and fixed the system for effective modelling.

2. Performed negative testing on the software with a lot of real world difficult test cases to check its robustness.

3. Measured the system's response times for user registration and identification/verification and made necessary architectural changes to further decrease the response time as per the requirement.

4. Tested conversation mode to check if the outputs were relevant with high speed processing.

CHAPTER 9
SOFTWARE TESTING

9.1 Type of Testing Used

Unit Testing Unit testing is the testing of an individual unit or group of related units.It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

Integration Testing Integration testing is a testing in which a group of components are combined to produce output.The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

System Testing System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing. It is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the systems compliance with the specified requirements.

Acceptance Testing Acceptance testing is a level of software testing where a system is tested for acceptability. The purpose of this test is to evaluate the systems compliance with the business requirements and assess whether it is acceptable for delivery.

Functional testing Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

Stress testing Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

Performance testing Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

9.2 Test Cases and Test Results

Unit testing -

Test case- The registration,identification,verification and conversation modes are some of the small modules which are used in unit testing.For example- taking input of the bio and voice recording of a user in the registration and identification parts respectively.

Results - The registration module successfully takes input of the bio of the user such name,age,sex,sample voice recording,contact icon,saving the recording and training it to create a model for that speaker/user.

- The identification module is able to take input of sample recording of a user and successfully identifies the speaker from the registered speakers in the database. - The verification module involves claiming his/her identity and then verifying whether the sample recording actually belongs to the claimed identity or not.

- The conversation mode is the dynamic mode and includes real-time identification of the speaker i.e. identifying the speaker who is currently speaking.

Integration testing -

Test case - It includes combining all the modules i.e. registration,identification,verification and conversation mode to check the interaction between these modules.

Results - All the modules were properly integrated and each module was producing its result correctly.The interaction among the modules did not create any problem in the final output.

System testing -

Test case- This included testing the product or the program in various operating systems.

Results - The program worked properly in both the operating systems i.e. Windows 10 and Ubuntu 16.04

Stress testing-

Test case- This included testing the system in unfavourable conditions like noise i.e. it involved lot of disturbance along with the desired voice.

Results- The overall accuracy was reduced while identifying the speaker as noise removal also involved removing of the desired voice sample.

Performance testing-

Test case- It involves how fast or quickly the speaker can be identified from his voice sample in the identification module and even in the real time identification of speaker in the conversation mode.

Results - The identification of a speaker needed a sample recording of 7-8 seconds to achieve a higher accuracy.

CHAPTER 10
RESULTS

10.1 Screen shots and Outputs

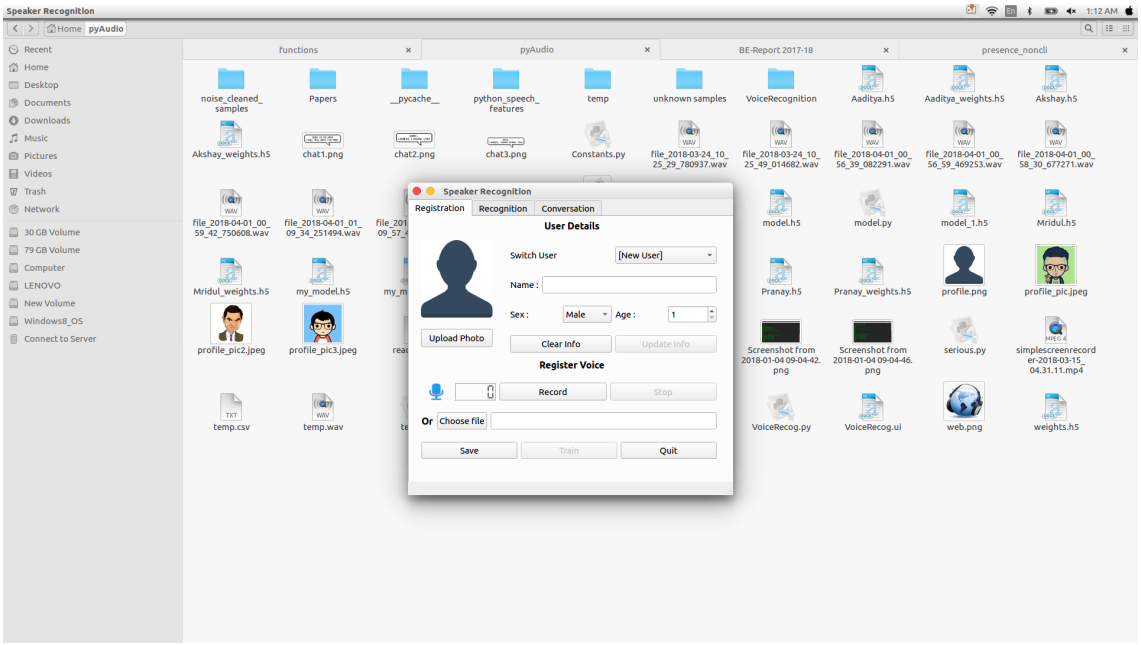


Figure 10.1: User and voice registration tab.

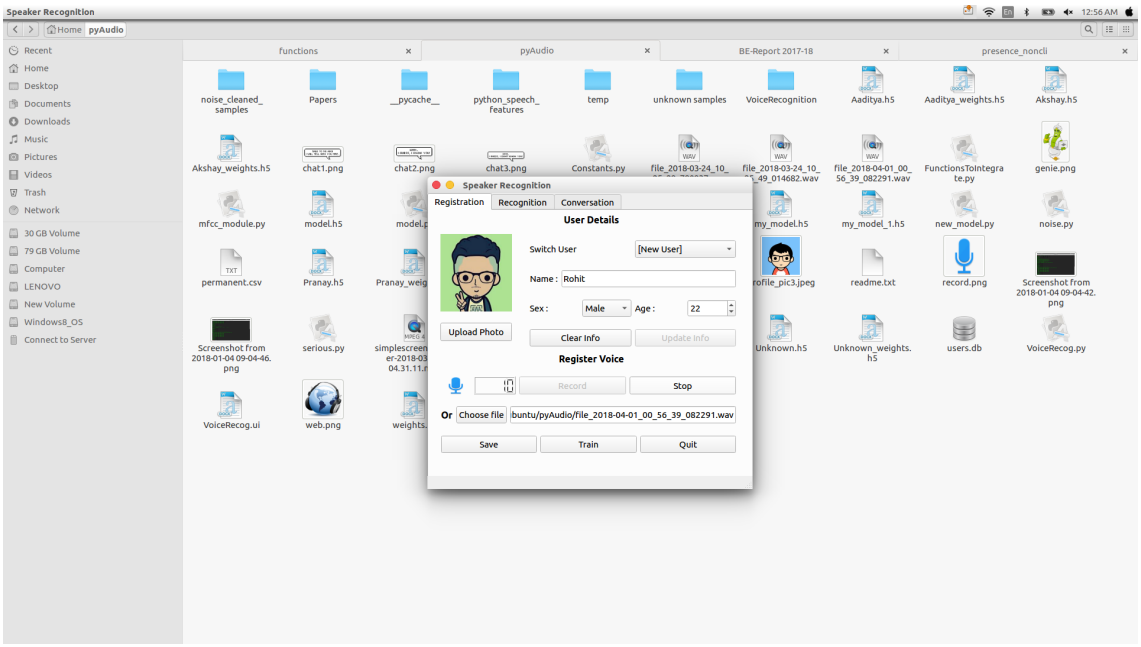


Figure 10.2: Registering new user.

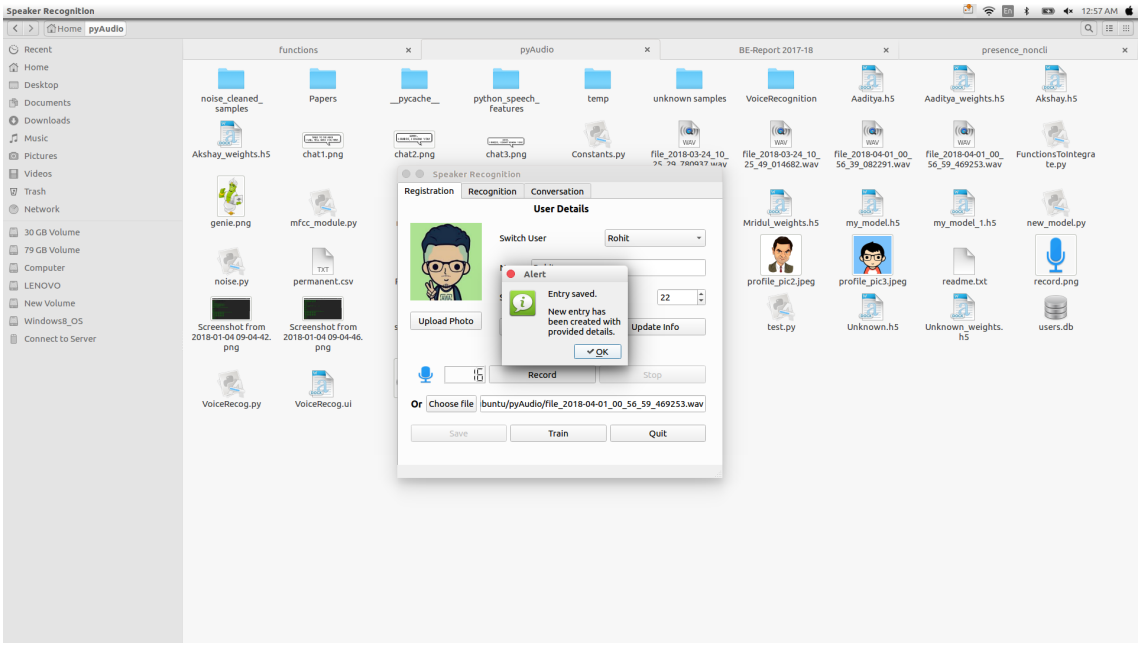


Figure 10.3: Saving new user in database.

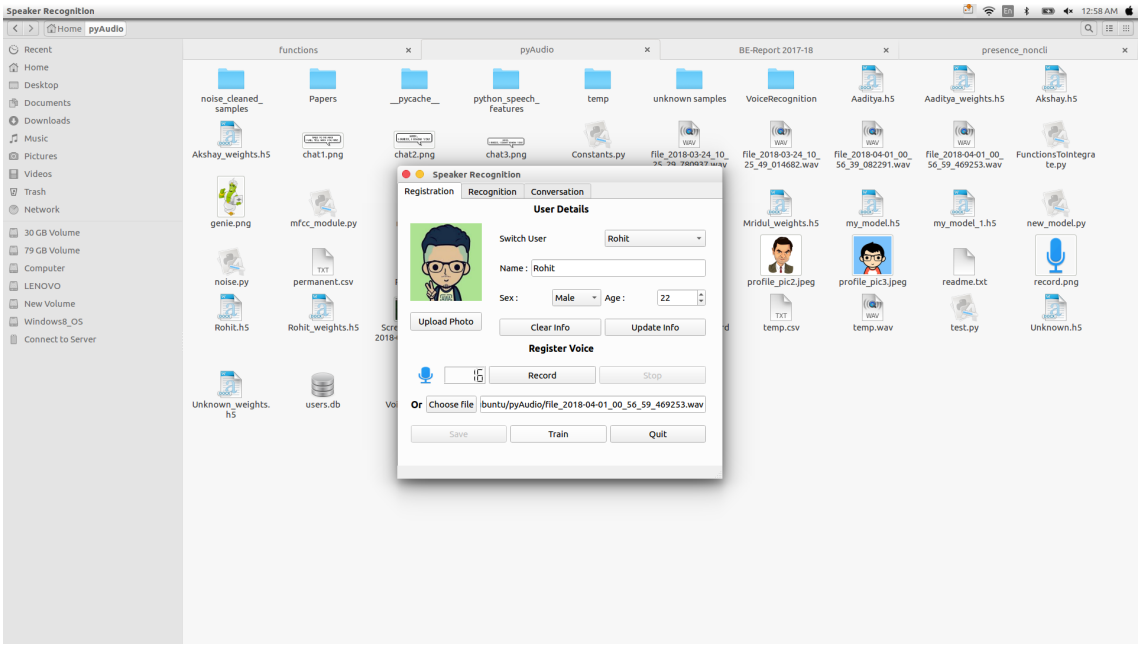


Figure 10.4: Training new user's voice sample.

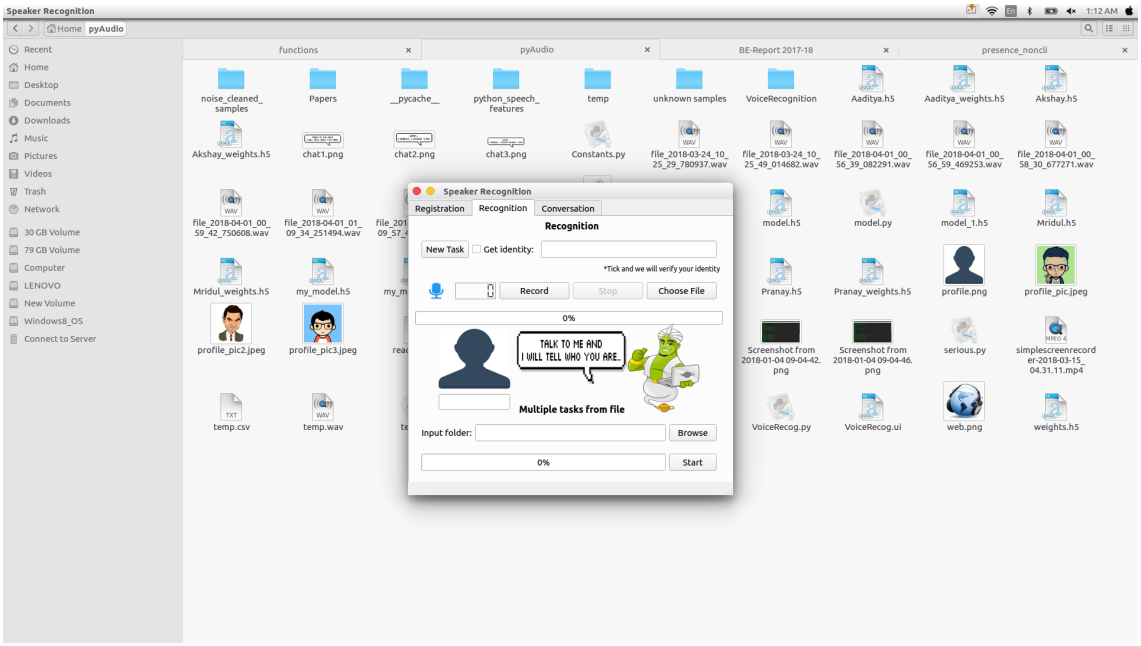


Figure 10.5: Speaker recognition tab.

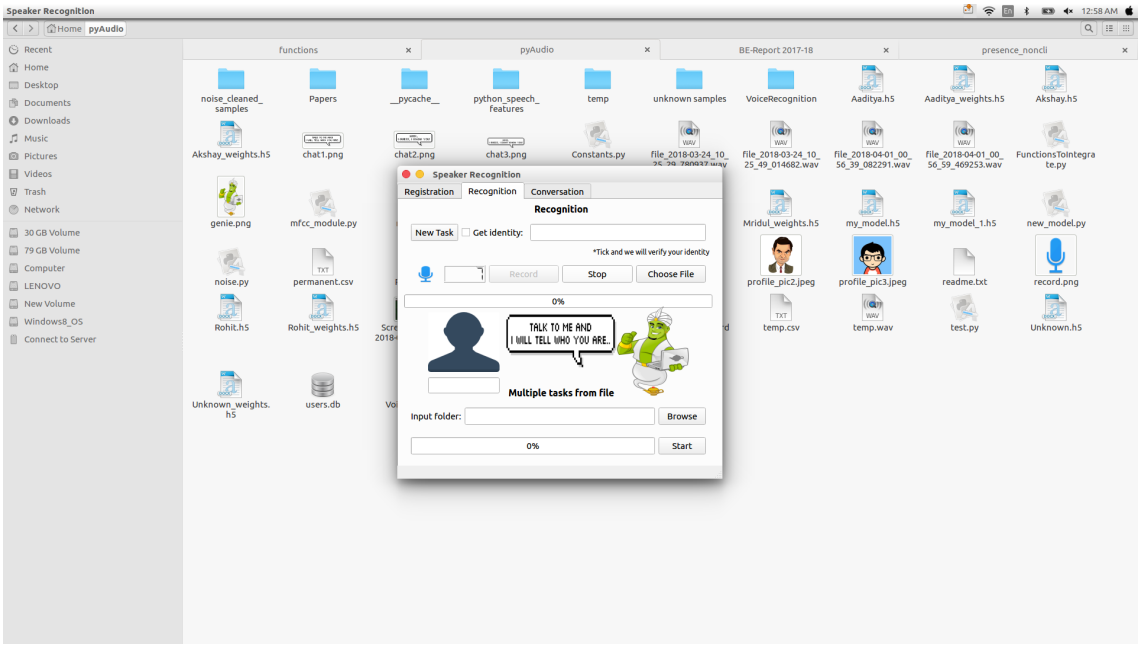


Figure 10.6: Recognising the voice sample of a user.

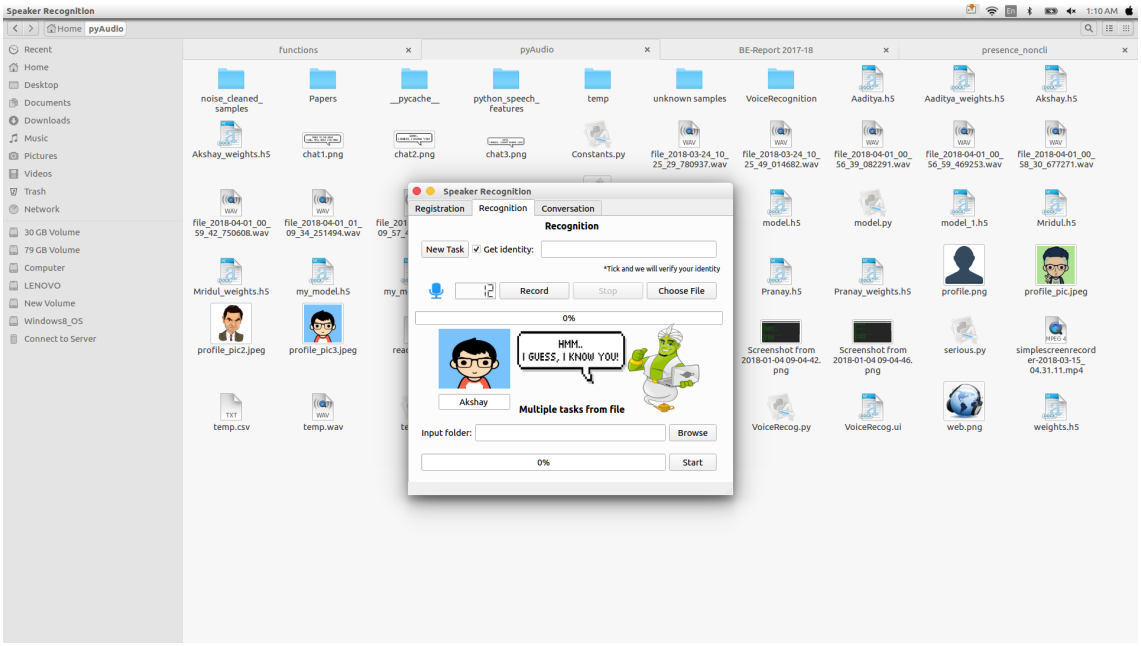


Figure 10.7: User voice recognised and displaying user name.

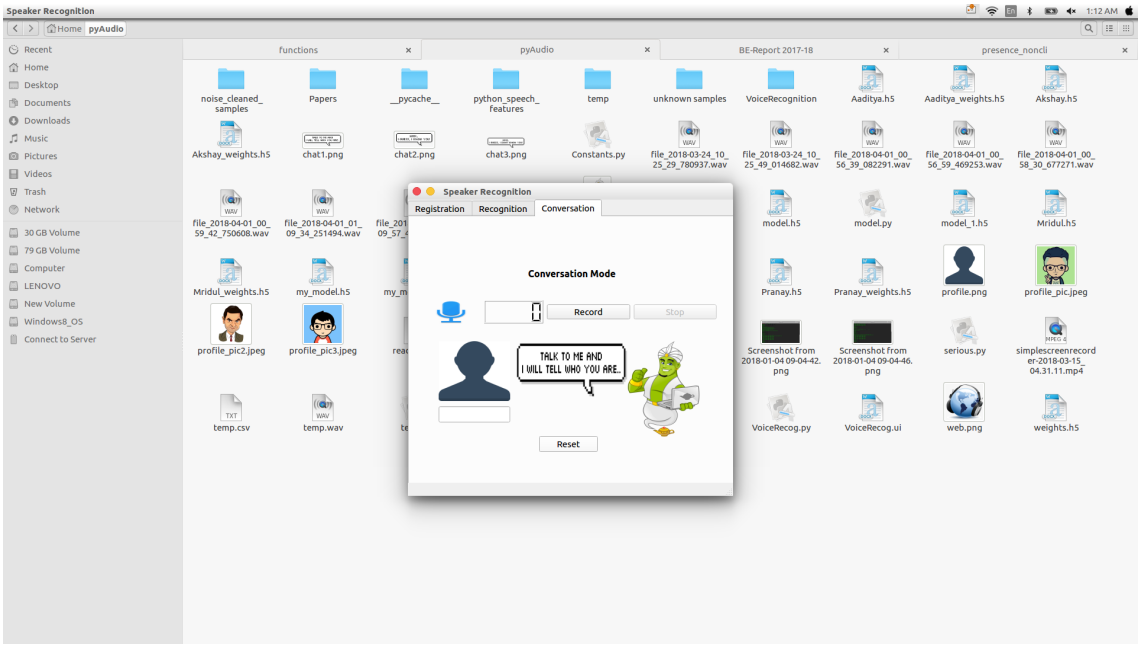


Figure 10.8: Conversation mode tab.

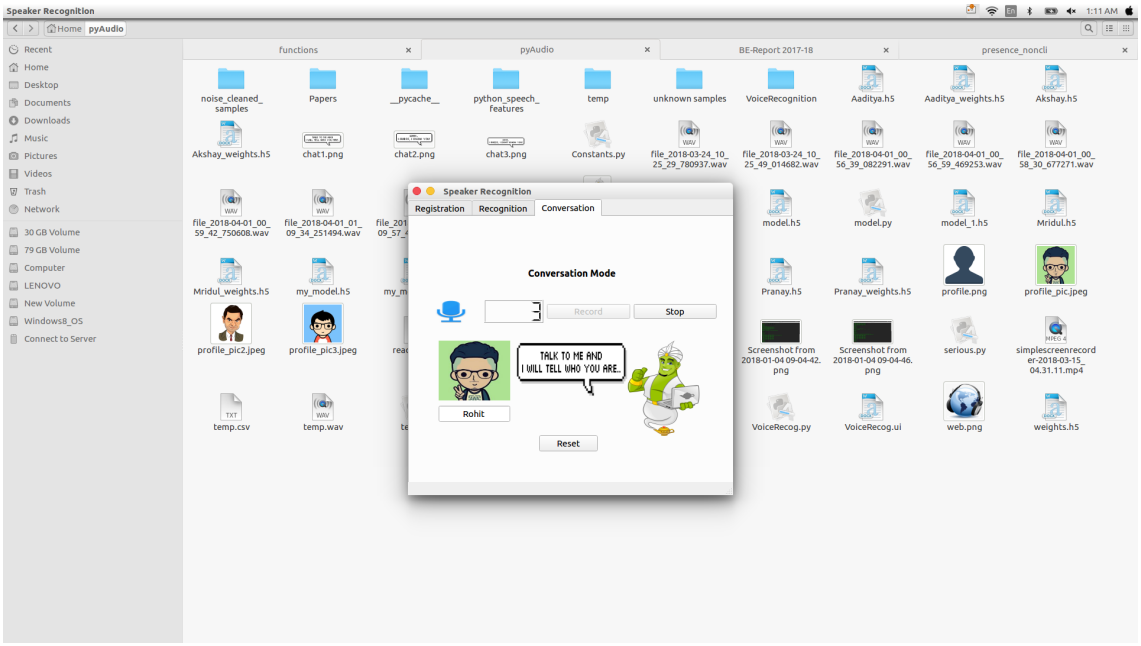


Figure 10.9: Recognising user in conversation mode.

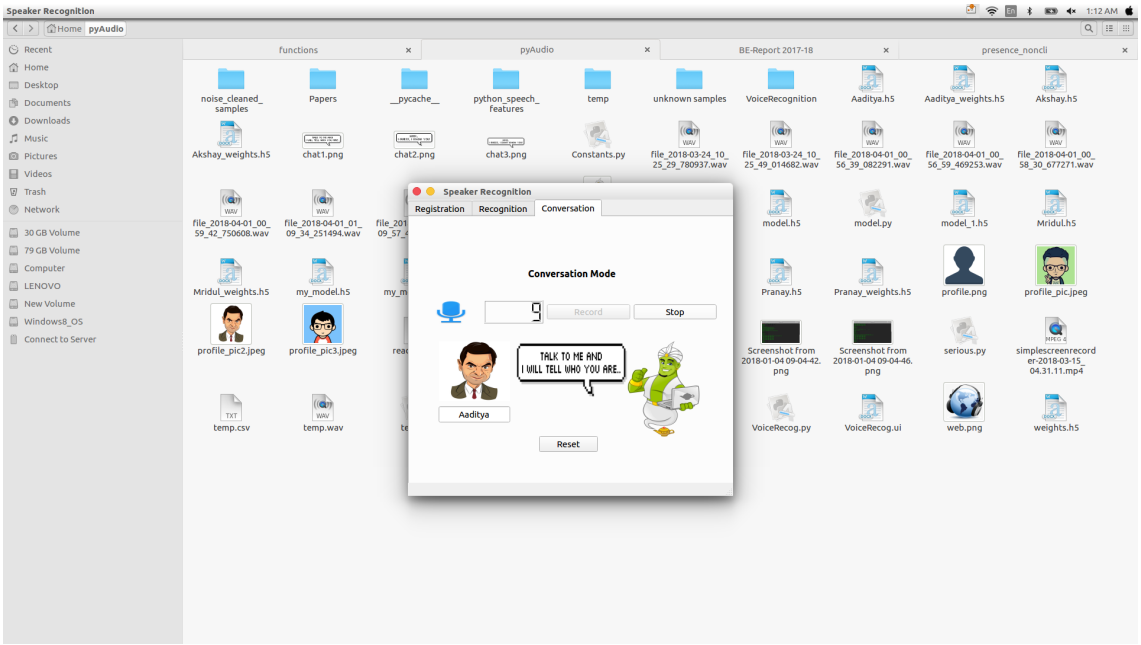


Figure 10.10: Recognising user in conversation mode.

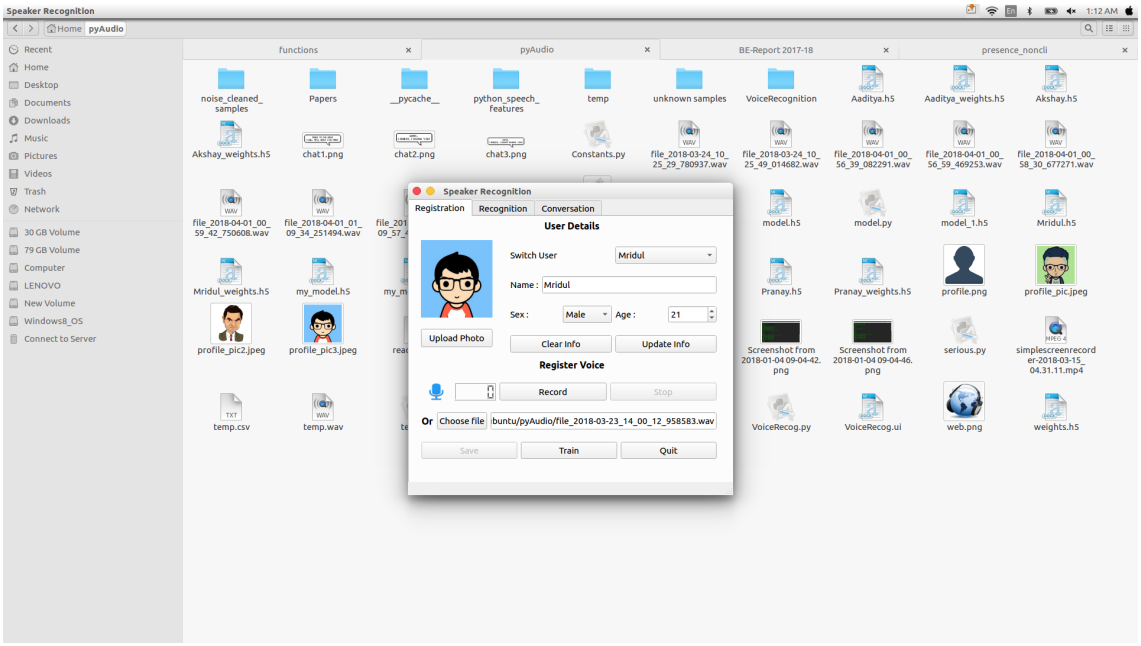


Figure 10.11: Selecting user for updation.

Analysis :

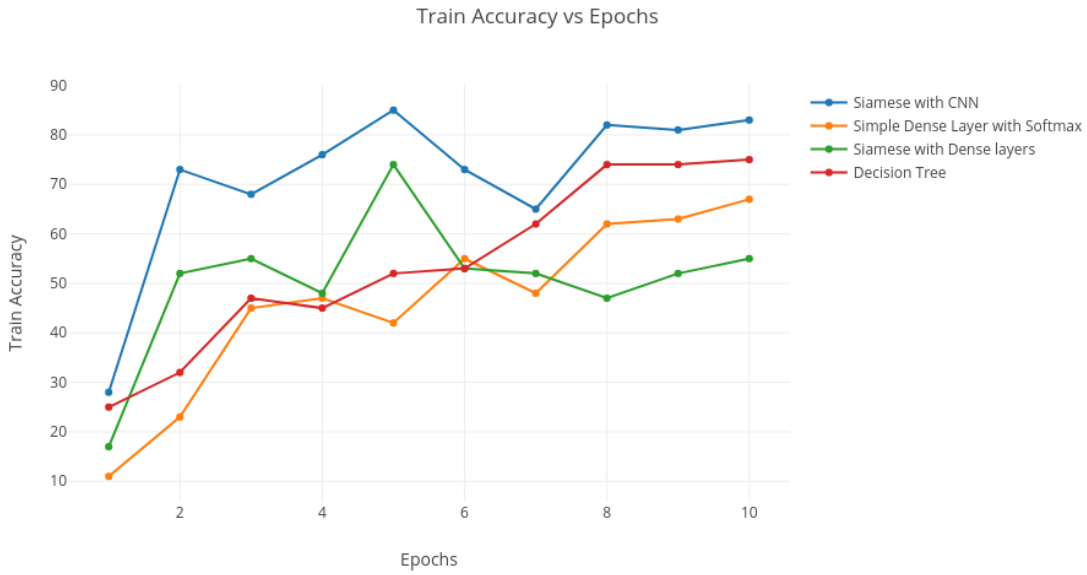


Figure 10.12: Figure depicts the train accuracy versus epochs time with varying results with multiple algorithms. Algorithms depicted are (i) Siamese Network with CNN layers (ii) Siamese Network with Dense Layers (iii) Simple Neural Network with Dense Layers (iv) Decision Trees. The CNN model shows a good train accuracy curve and decision trees have more steady growth per epoch.

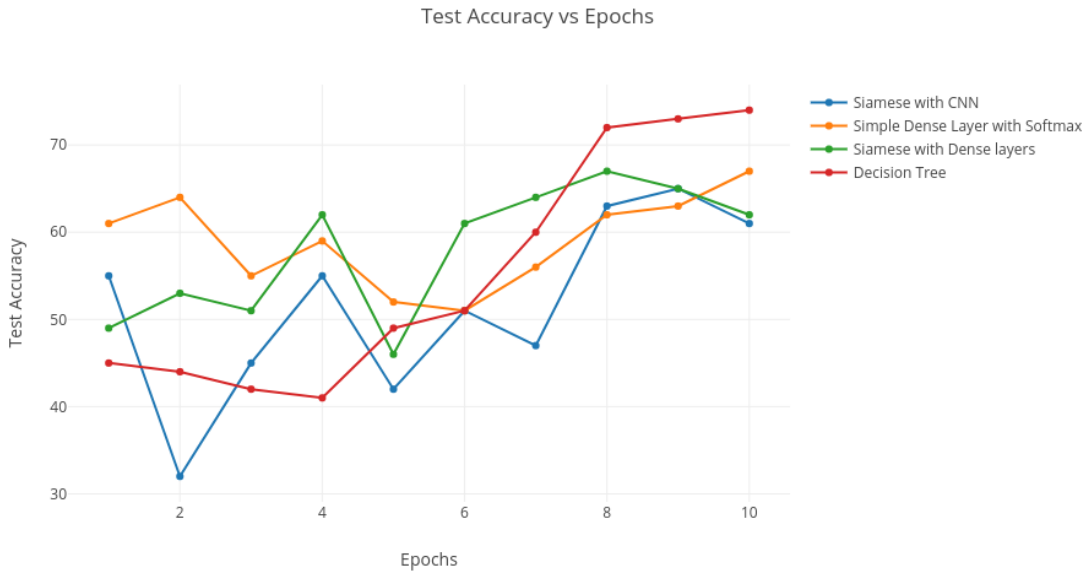


Figure 10.13: Figure depicts the test accuracy versus epochs time with varying results with multiple algorithms. Decision tree seems to be promising for our small dataset. However, CNN model accuracy for test falls as compared to its train accuracy. Simple NN shows a much more complying results with its training counterpart.

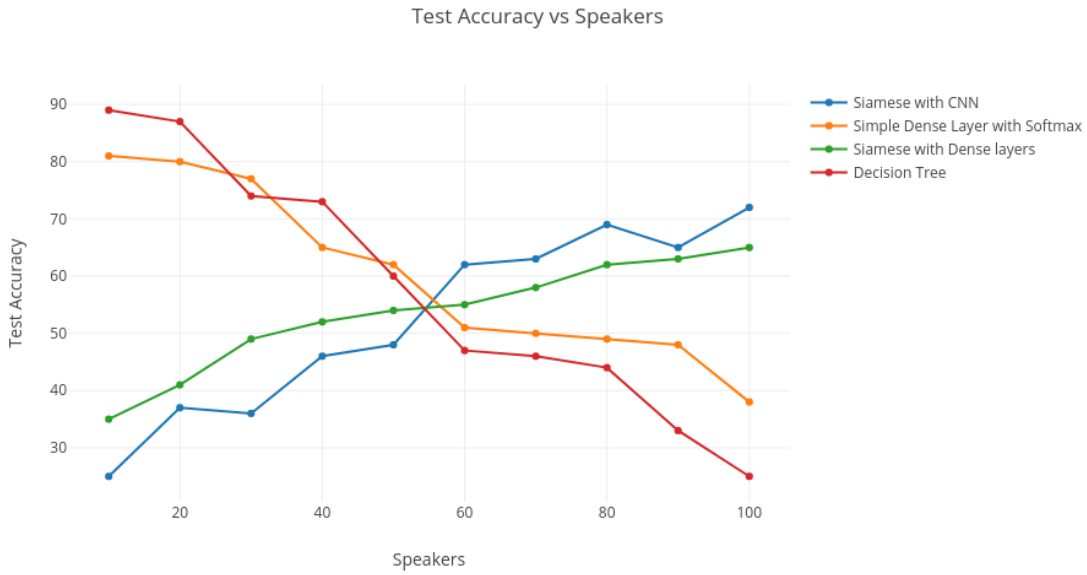


Figure 10.14: Figure depicts the test accuracy versus number of speakers with varying results with multiple algorithms. Decision tree seems to be promising for small number of speakers and its accuracy falls with increasing speakers. Simple NN shows a similar behavior and its accuracy also falls. However, Siamese network with either model fairs well with very large dataset, and accuracy increases with good number of counter examples and complex features to understand.

CHAPTER 11

DEPLOYMENT AND MAINTENANCE

11.1 Installation and un-installation

Installation

- Program is completely python based and hence Python3 is an essential requirement.
- Along with Python3 following libraries are required by the program
    1. PyQt5
    2. python speech features
    3. librosa
    4. pysndfx
    5. pyaudio
    6. wave
    7. keras
    8. tensorflow
    9. scipy
- Above mentioned libraries can be installed using following command:
  pip3 install package-name
- SpeakerRecognition program can be installed by unwrapping the tarball, command for the same is:
  tar -zxvf SpeakerRecognition.tar.gz

Un-installation

- For uninstalling the above mentioned libraries use following command
  pip3 uninstall package-name

11.2 User help

Graphical user interface is a tabed window containing three tabs.

- User details and voice registeration tab
- Voice recognition tab.
- Recognition in conversation mode tab.

1. User details and voice registeration tab (see fig 10.1).
    - User has to enter his/her name.
    - Select gender.
    - Enter his/her age.
    - Select profile picture.
    - Record a voice sample by clicking on record button as shown in fig 10.2.
    - Record voice sample for atleast 2 or more min and press stop button.
    - User can also select pre recorded voice sample from the device by clicking on choose file button (must be wav file). Name for selected file will appear in the text field below.
    - To save the user profile in database, user has to click on save button. Upon success, a pop window will open notifying the user that his/her profile is created in the database as shown in fig 10.3.
    - If user is getting an error pop up message, that means user must have forget to fill in the details required.
    - Once the user profile is saved click on Train button to start training see fig 10.4. Training is done in background and may take 15 to 20 seconds depending upon the CPU and memory of the target device.
    - In case if user wish to update his/her information, first select the user to be updated from the spinner provided and change the field you wish to update. Click on update button to update the information to the database refer fig10.11.
    - To exit the program click on quit button.
2. Voice recognition tab (see fig 10.5).

- In this tab user can recognize or claim his/her identity.
- User has to record his/her voice sample using record and stop button for minimum of 7 to 8 seconds.
- Or user can choose pre recorded wav file from the device by clicking on choose file button.
- Once the voice sample is ready select the checkbox indicating "Get Identity" as shown in fig 10.6.
- Program will detect the voice sample, user can check progress on the progress bar provided.
- Once the voice is recognised, user detail like name and profile pic will be displayed on the screen as shown in fig 10.7.
- In order to start new recognition, simply click on new task button which will reset every field in the tab.

3. Recognition in conversation mode tab (see fig 10.8).

    - This is a dynamic mode recognition tab. In this tab user's voice will be recognised as he/she speaks. Users has to speak in conference.
    - For this user has to click on record button and start speaking see fig 10.9.
    - Program will display the name and profile pic of the user speaking currently as shown in fig 10.10.
    - To stop the conversation mode click on stop buttton.
    - Clicking on reset button will reset all the field in the tab.

CHAPTER 12
SUMMARY AND CONCLUSION

The output of the project turned out that there is still a scope of achieving greater accuracy. The main idea of the project was to extract the voice features and use it to identify the individual in a resource-constricted environment is a challenging task. Real-time identification sends a corpus of 1-2 seconds and identifying the speaker in small data leads to loss of accuracy by **30%**. However, for smaller set of users as the use case, we diversified the algorithms to other machine learning algorithms like one-short learning, decision trees and convolutional nets. The application built, has two features as a whole, **validation** and **verification**. The Feed-forward nets proved to be good for verification, but siamese networks have shown better results in terms of validation. To achieve higher accuracy, the **WAV** files recorded are preprocessed for noise reduction and silence trimming. It helped to increase the accuracy by **5-10%**. The **API** developed allows you to record, preprocess, train and recognise as the functions. The overall accuracy of the project for **5 speakers** we claim to have 75-80%, however, the accuracy drops for less data and more number speakers. The application provides decent results for **4-5 mins** of audio sample for training and **6-7 seconds** of audio sample to recognise. But, keeping in mind, the use case, accuracy could have been improved by larger datasets for training.

Conclusion:

Hence we completed the following tasks related to project:

- End to End API for the proposed application.
- All basic features are packed into API including recording, preprocessing, training and recognizing.
- API is suitable for scalable applications however, larger the corpus, higher the accuracy.
- The domain poses a good area of research to make it robust and have least error rates.
- Documentation for the API and use help has been recorded for use.

CHAPTER 13
REFERENCES

[1] Shoumya Chowdhury, Nursadul Mamun, Ainul Anam Shahjamal Khan, Fahim Ahmed, *Text Dependent and Independent Speaker Recognition Using Neural Responses from the Model of the Auditory System.* International Conference on Electrical, Computer and Communication Engineering (ECCE), February 16-18, 2017, Coxs Bazar, Bangladesh.

[2] Mohammad Ali Nematollahi,Hamurabi Gamboa-Rosales, Francisco J. Martinez-Ruiz, Jose I. De la Rosa-Vargas, S. A. R. Al-Haddad, Mansour Esmaeilpour, *Multi-factor authentication model based on multipurpose speech watermarking and online speaker recognition.* Multimed Tools Appl (2017) 76: 7251

[3] Ala Eldin Omer, *Joint MFCC-and-Vector Quantization based Text- Independent Speaker Recognition System.* 2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE), Khartoum, Sudan

[4] Mohammad Soleymanpour, Hossein Marvi, *Text-independent speaker identification based on selection of the most similar feature vectors,.* Int J Speech Technol (2017) 20: 99

[5] Jia-Ching Wang, Chien-Yao Wang, Yu-Hao Chin, Yu-Ting Liu, En-Ting Chen, Pao-Chi Chang *Spectral-temporal receptive fields and MFCC balanced feature extraction for robust speaker recognition .* Multimed Tools Appl (2017) 76: 4055

[6] Sri Harsha Dumpala, Sunil Kumar Kopparapu *Improved Speaker Recognition System for Stressed Speech using Deep Neural Networks .* 2017 International Joint Conference on Neural Networks (IJCNN)

[7] Pulkit Verma, Pradip K. Das *A Comparative Study of Resource Usage for Speaker Recognition Techniques .* 2016 International Conference on Signal Processing and Communication (ICSC)

[8] Yi Wang, Dr. Bob Lawlor *Speaker Recognition Based on MFCC and BP Neural Networks.* 2017 28th Irish Signals and Systems Conference (ISSC)

[9] Abhinav Anand, Ruggero Donida Labati, Madasu Hanmandlu, Vincenzo Piuri, Fabio Scotti *Text-independent speaker recognition for Ambient Intelligence applications by using Information Set Features.* 2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)

[10] Koustav Chakraborty, Asmita Talele, Prof. Savitha Upadhya *Voice Recognition Using MFCC Algorithm.* 2014 International Journal of Innovative Research in Advanced Engineering (IJIRAE)

[11] Dr. S. Kalaivani, Rahul Singh Thakur, Dharmendra *Modified Hidden Markov Model for Speaker Identification System.* 2017 International Journal of Advances in Computer and Electronics Engineering (IJACEE)

[12] White Paper, *Why Combining Phonetics and Transcription Works Best* 2011, Nice Systems

ANNEXURE A

LABORATORY ASSIGNMENTS ON PROJECT ANALYSIS OF ALGORITHMIC DESIGN

A.1 IDEA Matrix

I	Improve	Accuracy of current speaker recognition systems.
I	Innovate	DSP algorithms and machine learning technique.
I	Ignore	Unknown Results.
D	Determine	Sources whose voice needs to be recorded.
D	Decide	Useful voice signal from noise.
D	Deliver	Robust real time authentication.
D	Decrease	Sources whose voice needs to be recorded.
E	Efficiency	Of the reusable-component based app.
A	Avoid	False positives.
A	Associate	The native functionalities to all the platforms.

Table A.1: IDEA Matrix

A.1.1 Intend

- To create an robust speaker recognition systems.
- To find features and use them to identify the speakers
- To use it for the authentication-based systems

A.1.2 Decrease

- The computation required to implement similar system.
- To decrease the cost required for implementation.
- To implement the system using minimum possible hardware.

A.1.3 Evaluate

- The voice features which uniquely defines it.
- Dependency on external environment noise.

A.1.4 Analyze

- Training accuracy and Recognition success rate.
- Quick fix approach in case of System failure.

A.2 To determine the feasibility of the project problem statement using NP-complete, NP-hard, or satisfiability issues using mathematical model.

A.2.1 Assessment of the problem statement

The MFCC algorithm is dimension-constrained and can be solved in polynomial time. However, the feature extraction may be NP-complete for some input. So, we will state the algorithm as NP-complete. But there is no such algorithm that guarantees the same for larger, multilayer neural networks.Unless P=NP,for any polynomial-time training algorithm there will be some sets of training data on which the algorithm fails to correctly train the network, even though there exist edge weights so the network could correctly classify the data. Therefore training a deep neural network is an NP-complete problem. While NP-completeness does not render a problem totally unapproachable in practice, and does not address the specific instances one might wish to solve, it often implies that only small instances of the problem can be solved exactly, and that large instances at best can be solved approximately even with large amounts of computer time.

A.2.2 Mathematical Model

Let S be the solution perspective of given problem statement.

$$S = \{s, e, X, D, Y, F, DD, NDD, Su, Fa\}$$

where,

s=start state
such that, $Y = \{\}$

e=end state;
such that, $Y = \{y_1\}$

y_1 = Authentication approved or denied.

X = set of input or input bitstream

such that $X = \{x_1, x_2, x_3, ..., x_n\}$
x_i = 16 bits or 1 sample
D = Data Store ;

such that $D = \{Z_1, Z_2\}$
where,
Z_1 = Label;
$Z_2 = \{< z_1, z_2, z_3, ... > \| mfcccoefficients\}$
Y = set of output ;

such that $Y = \{y_1, y_2\}$
y_1 = Authentication approved or denied.
y_2 = output coefficients of signal processor.
F = set of function ;

such that $F = \{f_1, f_2, f_3, f_4, f_5\}$
f_1 = function to accept WAV file bit stream
f_2 = function to process the signal and generate mfcc coefficients
f_3 = function to train the speaker model.
f_4 = function for pattern matching or recognition
f_5 = function to authenticate the pattern match results and give access or not

$f_1(X_i) = \{X = X_i\}$
f_2(X) = $\{y_2\}$
$f_3(y_2) = D$
$f_4(y_2)$ = temp
f_5(temp,D) = $\{y_1 | 0 or 1\}$

DD = Deterministic data.
$DD = |X|$

NDD = Non deterministic data
$NDD = \{y_1, y_2\}$

Su = Success
If the access is granted for a given voice input.

Fa = Failure
If the access is not granted and classified as false input or illegal input.

ANNEXURE B

LABORATORY ASSIGNMENTS ON PROJECT QUALITY AND RELIABILITY TESTING OF PROJECT DESIGN

B.1 Use of divide and conquer strategies to exploit distributed / parallel / concurrent processing of the above to identify objects, morphisms, overloading in functions (if any), and functional relations and any other dependencies (as per requirements).

B.1.1 Morphism

In many fields of mathematics, morphism refers to a structure-preserving map from one mathematical structure to another. The notion of morphism recurs in much of contemporary mathematics. In set theory, morphisms are functions; in linear algebra, linear transformations; in group theory, group homo-morphisms; in topology, continuous functions, and so on. The study of morphisms and of the structures (called objects) over which they are defined is central to category theory. Much of the terminology of morphisms, as well as the intuition underlying them, comes from concrete categories, where the objects are simply sets with some additional structure, and morphisms are structure-preserving functions. In category theory, morphisms are sometimes also called arrows. Morphism is refers to a structure-preserving map from one mathematical structure to another.

B.1.2 Class Diagram

please refer for class diagram 7.2

B.2 Function Dependency Diagram

please refer for Function Dependency diagram 6.2

B.3 To draw functional dependency graphs and relevant UML diagrams or other necessities using appropriate tools.

B.3.1 UML Diagrams:

B.3.1.1 Architecture Diagram:

please refer for Architecture diagram 7.1

B.3.1.2 Data Flow:

please refer for Data-flow diagram 6.4

B.3.1.3 Use Case Diagram:

please refer for Use Case diagram 6.1

B.4 Testing of project problem statement using generated test case data(using mathematical model, GUI, function testing principle) selection and appropriate use of testing tools, testing of UML diagram reliability.

B.4.1 Testing

Testing is an activity that can be planned and conducted systematically. Testing begins at the module level and works towards the integration of entire computer based system. Nothing is complete without testing, as it is vital for success of the system.

B.4.1.1 Testing Objectives

- Testing is a process of executing a program with the intent of finding errors.
- A good test case is the one having highest probability of finding an undiscovered error.

- It demonstrates that software functions appear to be working according to the specifications.

There are three reasons to test the correctness of the program

- For correctness
- For implementation efficiency
- For Computational Complexity

B.4.1.2 Unit Testing

Unit testing focuses on the smallest unit of software design i.e. the smallest component or module. Important control paths are tested to uncover errors within the boundary of the module. It focuses on the internal processing logic and data structures within the boundaries of a component. This type of testing can be conducted in parallel for multiple components.

B.4.1.3 Integration Testing

Integration testing is a systematic technique for constructing the software architecture while at the same time conducting tests to uncover errors associated with interfacing. The different modules in our project were interfaced and tested in small increments, thus making the errors easy to isolate and correct. This is known as incremental integration.

B.4.1.4 Validation Testing

Validation testing begins at the culmination of integration testing, when individual components have been exercised, software is completely assembled as a package, and interfacing errors have been uncovered and corrected. Here, testing focuses on user-visible actions and user recognizable output from the system. Validation succeeds when the software functions in a manner that can be reasonably expected by the customer.In our project, all functions and performance characteristics are tested and they conform to the required specifications and are accepted. If you are optimizing your site for Search engines then HTML/CSS validation is the most important one. Mainly validate the site for HTML syntax errors. Check if the site is crawlable to different search engines.

B.4.1.5 System Testing

This is the final step in testing. In this phase, we tested the entire system as a whole with all forms, code, modules and class modules. This testing is

known as Black Box testing or System testing. Black Box testing enables us to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing helps to discover incorrect or missing functions, interface errors, errors in data structure, performance errors,initialization and termination errors.

B.4.1.6 Functionality Testing

Test for all the links in web pages, database connection, forms used for submitting or getting information from user in the web pages.

B.4.1.7 Test cases

Control Test Scenario			
Test Case	Test Case Objective	Expected Result	Status
1	Check for number of speakers in frame	Label the frame as unknown	Pass
2	Check for for only noise frame	Show no speaker found	Pass
3	Give an invalid input	System shows error	Pass

Table B.1: Control Test Scenario

Performance Test Scenario			
Test Case	Test Case Objective	Expected Result	Status
1	System on pure noise	System Fails	Pass
2	Multiple voice in single frame	System labels it unknown	Pass
3	Accuracy of Training	System reaches the result with a decent deviation	Pass

Table B.2: Performance Test Scenario

ANNEXURE C
PROJECT PLANNER

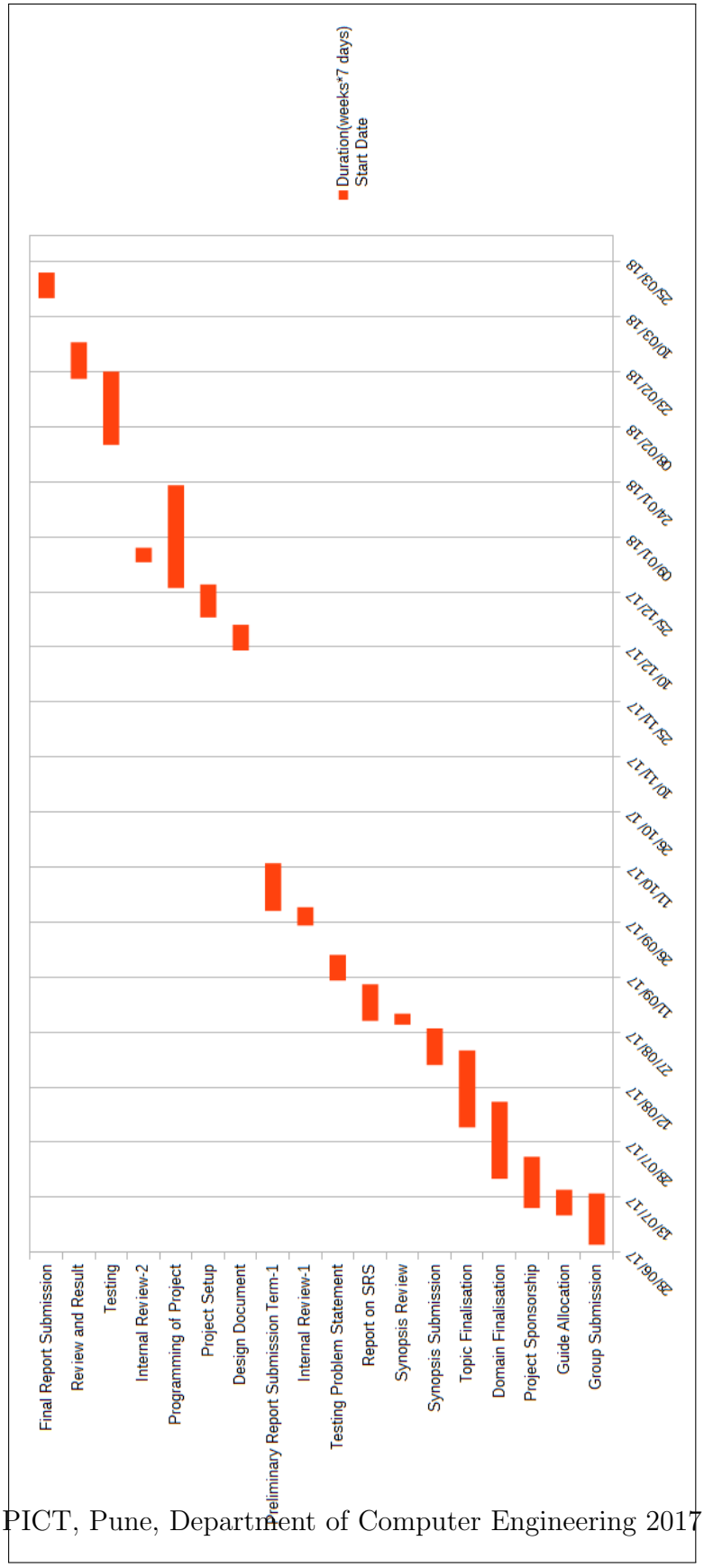


Figure C.1: Project Plan

ANNEXURE D

TERM-II PROJECT LABORATORY ASSIGNMENTS

1. **Review of design and necessary corrective actions :** The analysis of the difficulty of implementation, the project diversified to research orientation where we explored various methods to achieve accuracy and improve the then obtained accuracy.

   (a) **Siamese Network :** Siamese networks are a special type of neural network architecture. Instead of a model learning to classify its inputs, the neural networks learns to differentiate between two inputs. It learns the similarity between them. There are two sister networks, which are identical neural networks, with the exact same weights. The objective of the siamese architecture is not to classify input images, but to differentiate between them. So, a classification loss function (such as cross entropy) would not be the best fit. Instead, this architecture is better suited to use a contrastive function. Intuitively, this function just evaluates how well the network is distinguishing a given pair of images.

   (b) **Convolutional Neural Networks :** Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. (Note that the word depth here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.) For example, the input images in CIFAR-10 are an input volume of activations, and the volume has dimensions 32x32x3 (width, height, depth respectively).

   (c) **Feed-Forward Neural Networks :** Feedfoward Neural Networks are artificial neural networks where the connections between units do not form a cycle. Feed-foward neural networks were the first type of artificial neural network invented and are simpler than their counterpart, recurrent neural networks. They are called feed-forward because information only travels forward in the network (no loops), first through the input nodes, then through the hidden nodes (if present), and finally through the output nodes.

   (d) **Decision Trees :** Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) that is mostly used in classification problems. It works for both categorical and continuous input and output variables. In this technique, we split the population or sample into two or more homogeneous sets (or

sub-populations) based on most significant splitter / differentiator in input variables.

2. **Project workstation selection and installations :** Since the project relied heavily on GPU, the workstation required to meet the minimum of following system requirements:

   - 8 GB of RAM : To run tensorflow CPU-dependant
   - 4 GB or more of NVIDIA graphics card : To run tensorflow GPU-dependant
   - Python Support environment
   - Constant internet communication to deploy services on AWS cloud.

   Installation :

   - **Python Libraries :** pip install package-name
   - **Tensorflow :** https://www.tensorflow.org/install/

3. For Algorithms, Please refer to Chapter 8

4. For Testing, Please refer to Chapter 9

ANNEXURE E

INFORMATION OF PROJECT GROUP MEMBERS

1. Name : Akshay Hiwale
2. Date of Birth : 22nd January, 1996
3. Gender : Male
4. Permanent Address : Tilak Road, Pune
5. E-Mail : hiwaleakshay@gmail.com
6. Mobile/Contact No. : 8087507421
7. Placement Details : Software Developer at EQ Technologic

1. Name : Aaditya Pandilwar
2. Date of Birth : 26th January, 1996
3. Gender : Male
4. Permanent Address : Udgir, Latur
5. E-Mail : aadityap.pandilwar@gmail.com
6. Mobile/Contact No. : 9561626981
7. Placement Details : Analyst at Deutsche Bank

1. Name : Mridul Khanna
2. Date of Birth : 26th June, 1996
3. Gender : Male
4. Permanent Address : Koparkhairne, Thane
5. E-Mail : mridulk.khanna@gmail.com
6. Mobile/Contact No. : 9158260566
7. Placement Details : Software Developer at Sagitec Solutions

1. Name : Pranay Kundu
2. Date of Birth : 16th September, 1994
3. Gender : Male
4. Permanent Address : ACC Colony, Chandrapur
5. E-Mail : pranaykundu16@gmail.com
6. Mobile/Contact No. : 7385490193
7. Placement Details : Technology Analyst at Morgan Stanley