Chapter 1 Introduction

Speaker recognition is the process of automatically recognizing who is speaking by using the speaker-specific information included in speech waves to verify identities being claimed by people accessing systems; that is, it enables access control of various services by voice. Applicable services include voice dialing, banking over a telephone network, telephone shopping, database access services, information and reservation services, voice mail, security control for confidential information, and remote access to computers.

Speech is a complicated signal produced as a result of several transformations occurring at several different levels: semantic, linguistic, articulatory, and acoustic. Differences in these transformations are reflected in the differences in the acoustic properties of the speech signal. In speaker recognition, all these differences are taken into account and used to discriminate between speakers. The system designed has potential in several security applications. Examples may include, users having to speak a PIN (Personal Identification Number) in order to gain access to the laboratory they work in, or having to speak their credit card number over the telephone line to verify their identity. By checking the voice characteristics of the input utterance, using an automatic speaker recognition system similar to the one that we will describe, the system is able to add an extra level of security.

1.1 Problem definition

The goal of the proposed SID system is to determine which person(s) spoke which audio stream that contains both overlapping and non-overlapping speech. Although an ideal system should be designed as closely as to the practical use, it is difficult to consider all the application scenarios in an initial development stage. Hence, we began this research by identifying the important factors that influence the effectiveness of the system. We then defined the scope of this preliminary study.

In summary, our aim, as shown in Figure, is to label the identity of speaker(s) in a long speech utterance.

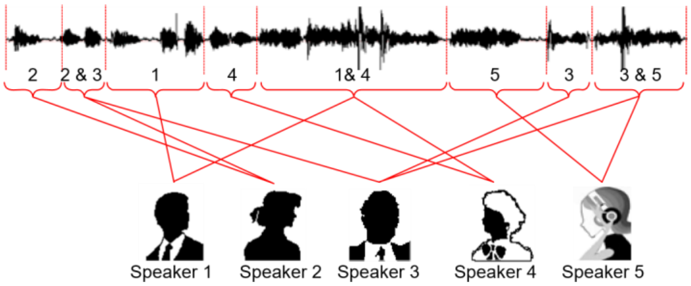


Figure 1.1 Illustration of our speaker-identification task.

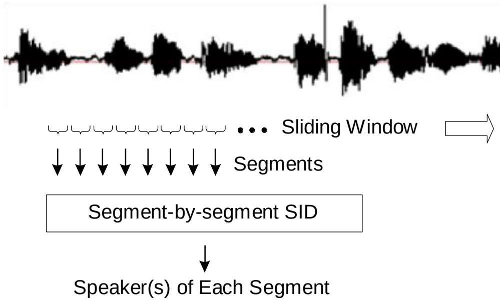


Figure 1.2 Our basic strategy for identifying the speakers in an audio stream.

1.2 Relevance of the Project

The proposed software product is the Speaker Identification System. The system will enable user to extract and isolate the individual voice streams at the receiver end. It can be used in various situations such as office meetings, our parliament sessions to identify the speaker and his content to draw conclusions. The system will help the user to highlight a particular speaker's voice amongst other speakers.

1.3 Scope of the Project

Sound Isolation in a Multispeaker Environment is a system which is designed to identify individual speakers in a mixed audio signal using various sound features and mechanisms. Features like MFCC, pitch, peak proceeds with the identification and segregating the signal into small size frames. After the formation of the frames of every speaker in the signal, the frames will be checked with the database and displayed.

Chapter 2 Review of Literature

2.1 Speaker Identification in Overlapping Speech

The paper we have chosen as the base of our project is a technical paper written by Wei-Ho Tsai and Shih-Jie Liao from the National Taipei University of Technology. The paper highlights the issue of identifying separate speakers in a multi-speaker environment.

The paper introduces Single Speaker Identification which has seen a lot of development and success and goes on to explain the problem of multiple speakers and their identification in a conversation.

The important applications of Multi-speaker identification are also listed, which include the likes of suspect identification in police work, and automated minuting of meetings,

The paper explains two approaches to solving this problem.

1. A two stage process where the signal is first tested to identify whether it contains speech from a single speaker or from multiple speakers.

2. The second approach is a single stage process that carries out the single speaker and multi-speaker identification in parallel.

The system advocates the first approach as the more efficient one. Given an input signal, the system first gives the output of phase one as shown in figures:

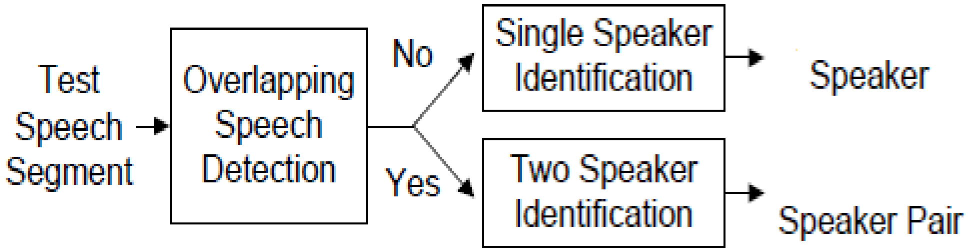
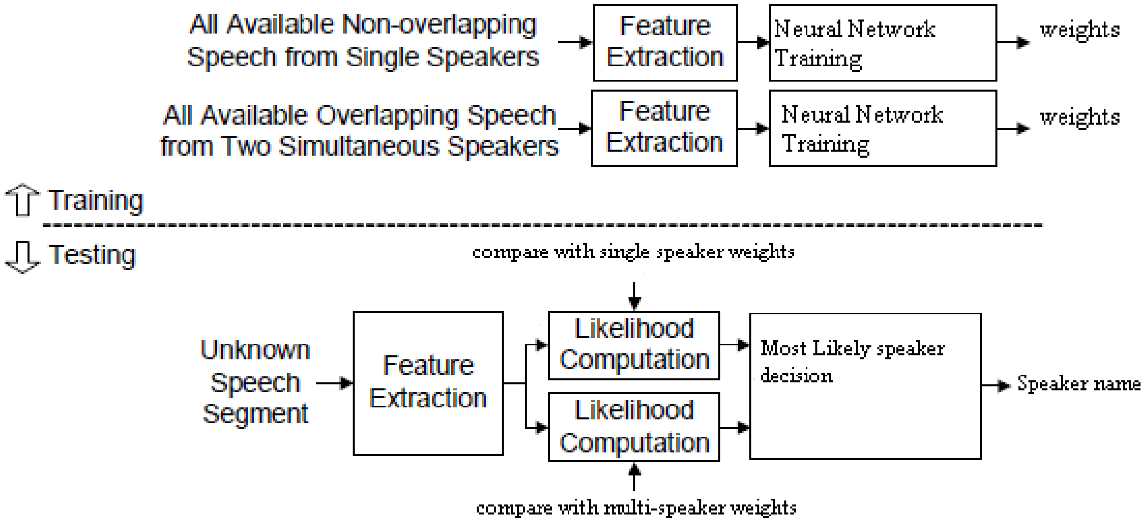

Test Speech Segment
Overlapping Speech Detection
No
Yes
Single Speaker Identification
Two Speaker Identification
Speaker
Speaker Pair

Figure 2.1 Two stage Speaker Identification System

After identifying single or multi-speaker the result is computed as follows:



All Available Non-overlapping Speech from Single Speakers
Feature Extraction
Neural Network Training
weights
All Available Overlapping Speech from Two Simultaneous Speakers
Feature Extraction
Neural Network Training
weights
Training
Testing
compare with single speaker weights
Unknown Speech Segment
Feature Extraction
Likelihood Computation
Likelihood Computation
Most Likely speaker decision
Speaker name
compare with multi-speaker weights

Figure 2.2 Speaker identification task

2.2 Assumptions:

- **Multiplicity:** In general, the complexity of the SSID problem grows as the number of simultaneous speakers in an audio recording increases. This study focuses on SSID capable of determining the identity of three speakers.
- **Overlapping energy ratio:** Depending on the signal energy received by a microphone, some speakers in a recording may sound like foreground speaker(s) while others may sound like background speaker(s). In such a case, identifying the background speakers would be more difficult. In this study, no test data has background speakers, i.e., people speak simultaneously with roughly equal signal energies.
- **Content variations:** Simultaneous speakers may speak the same or different contents (sentences). The recordings are of each speaker speaking the numbers 0-9 separately and simultaneously. We consider both cases in this study.
- **Open-set/close-set:** The SID problem at hand is a close-set classification problem, which identifies the speaker(s) among a set of candidate persons in test recordings. This study does not

discuss the problem of open-set classification, which needs to determine whether the speaker(s) is/are among the candidate persons.

- **Audio quality:** Ideally, a successful SID system should be robust against various signal distortions. This study, regardless, does not address this issue specifically because it is an inevitable problem for most speech-recognition research topics. We only consider the speech data recorded by high quality microphones and in quiet environments. When it is required to deal with noisy or distorted speech, there are numerous related techniques that can be applied in this work without specific tailing.
- **Non-speech factors:** Distinguishing speech from non-speech segments brings a number of other issues, such as speech/music discrimination, speech/animal sound discrimination, which are beyond the scope we can address in this study. Thus, to concentrate on SID problem, the speech data used in this work discards silent non-speech segments in the silence removal phase.

Chapter 3 Description

3.1 Proposed System

Like any other pattern recognition systems, the proposed speaker recognition (identification) system involves two phases namely, training and testing. Figure2 shows the layout of system. Each phase has its own stages. As shown in the figure, the stages pre-processing and feature extraction are in common. The stages can be summarized as follows:

3.1.1 Preprocessing: This stage includes the following steps:

- Pre-emphasis
- Framing
- Silence Removal
- Windowing.

To pre-emphasis speech signal, a high pass filter is implemented in this process. To achieve stationary signal, audio is divided into segments (frames) with fixed duration, each with of 480 samples. Silence is removed by testing the energy value of each frame respect to certain threshold value to determine which frame is not silent. Hamming windowing process is applied on overlapped frames.

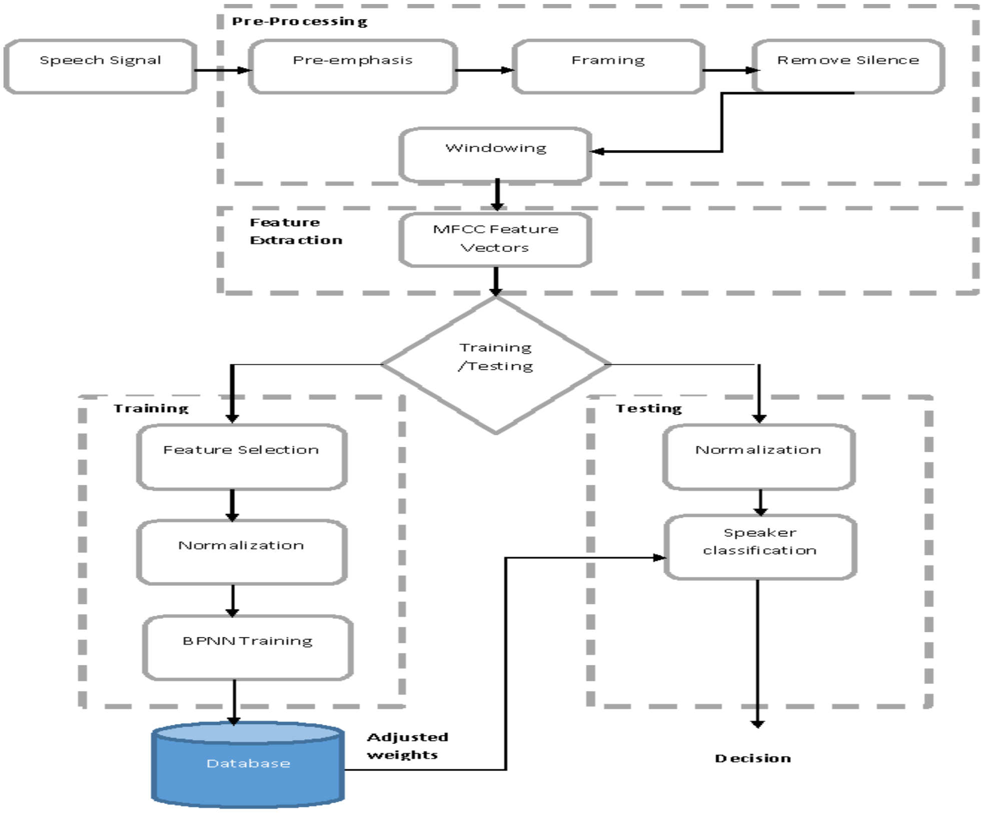


Figure 3.1 Layout of proposed system

3.1.2 Features extraction

It is achieved by MFCC technique and implemented as depicted in figure2. It includes the following steps:

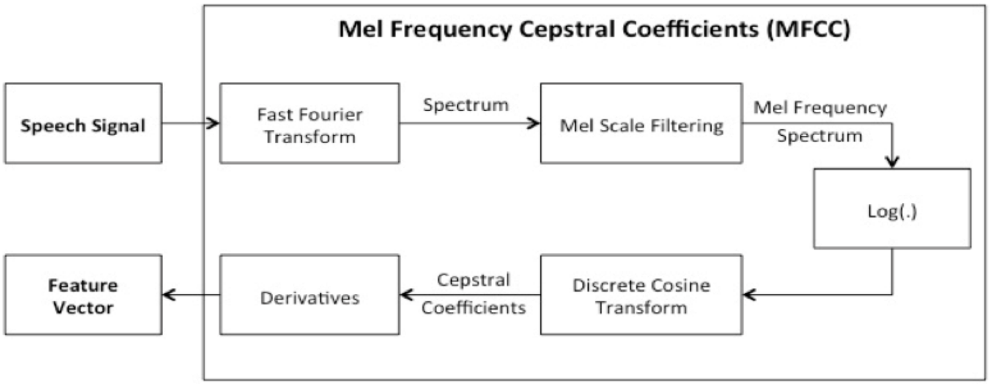


Figure 3.2 MFCC Flowchart

- Convert each frame into frequency domain by FFT.
- Map the power of spectrum of each frame into Mel-scale.
- Take logarithm of these energies.
- DCT is taken of the logarithm of these energies.

In our system, each MFCC frame vector consists of 20 features.

3.1.3 Normalization:

The vectors have been normalized because neural network training could be made more efficient by performing certain pre-processing steps on the network inputs and targets. Network input processing functions transform inputs to a better form for network use. Without normalizing, training the neural networks would be very slow. In this system, the values are normalized in the range of 0 to 1. Having used and compared min-max and our method of normalization, the latter gave a better accuracy and a faster train time. Therefore, the normalization technique used is:

1. Select the maximum value from the input data set.
2. Divide all data set values by the maximum value to get the normalized matrix.

Suppose X is the input matrix and x is the normalized matrix

n=max(X);

x=X/n;

We tried another normalizing technique which is given as follows:

1. The absolute value of the smallest element of the input set is taken
2. Add input features with the smallest value
3. Divide the input set by 100.

Suppose X is the input matrix and x is the normalized matrix

n=|min(X)|;

X=X + n;

x=X/100;

This technique led to a decrease in accuracy compared to min-max normalization and hence was discarded.

3.1.4 Back-propagation Neural Network:

For our implementation, Multilayer Feed forward Network (three layer neural network) has been created using MATLAB. For training the network, Back Propagation algorithm was used. The network consists of an input layer of 20 neurons, one hidden layer of 16 neurons and an output layer contains 3 neurons used to recognize 3 speakers each. We used a set of MFCC (Mel Frequency Cepstral coefficients) feature vectors as input pattern for the neural network. In our design we put all these input vectors in a 'Input Layer' variable matrix. As 3 output neurons, output matrix contains 3x1 unit matrix. Hyperbolic tangent sigmoidal activation function (tansig) is included in the hidden layer. For training the network, at first Randomized weights and biases are set using random function ranging values from -1 to 1. For each training pattern, network layers weights and biases are updated using back-propagation algorithms to reach the target. The weights and biases of the network are updated until the network error reaches almost zero. At testing phase, the trained network was simulated with unknown speech pattern. It was observed that the trained network performs very well and more than three speakers can be recognized by using the developed system.

3.2 Design approach

The current system will take audio stream as the input. The next step is the feature extraction from the input. The features extracted are MFCC. The next step is the feature matching and comparison phase in which the extracted features are compared with the features of the speakers that are stored in the database before-hand. Sound Isolation is done based on the extracted features. The individual speaker can be recognized after the features are matched. The different diagrams related to the project are:-

1) DFD

2) Use-Case diagram

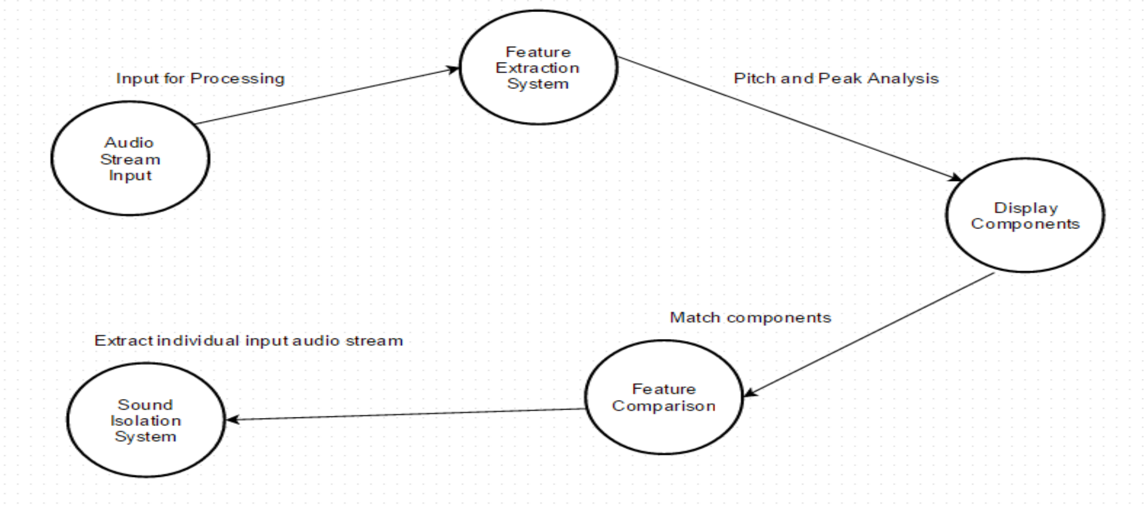


Figure 3.3 Data Flow Diagram

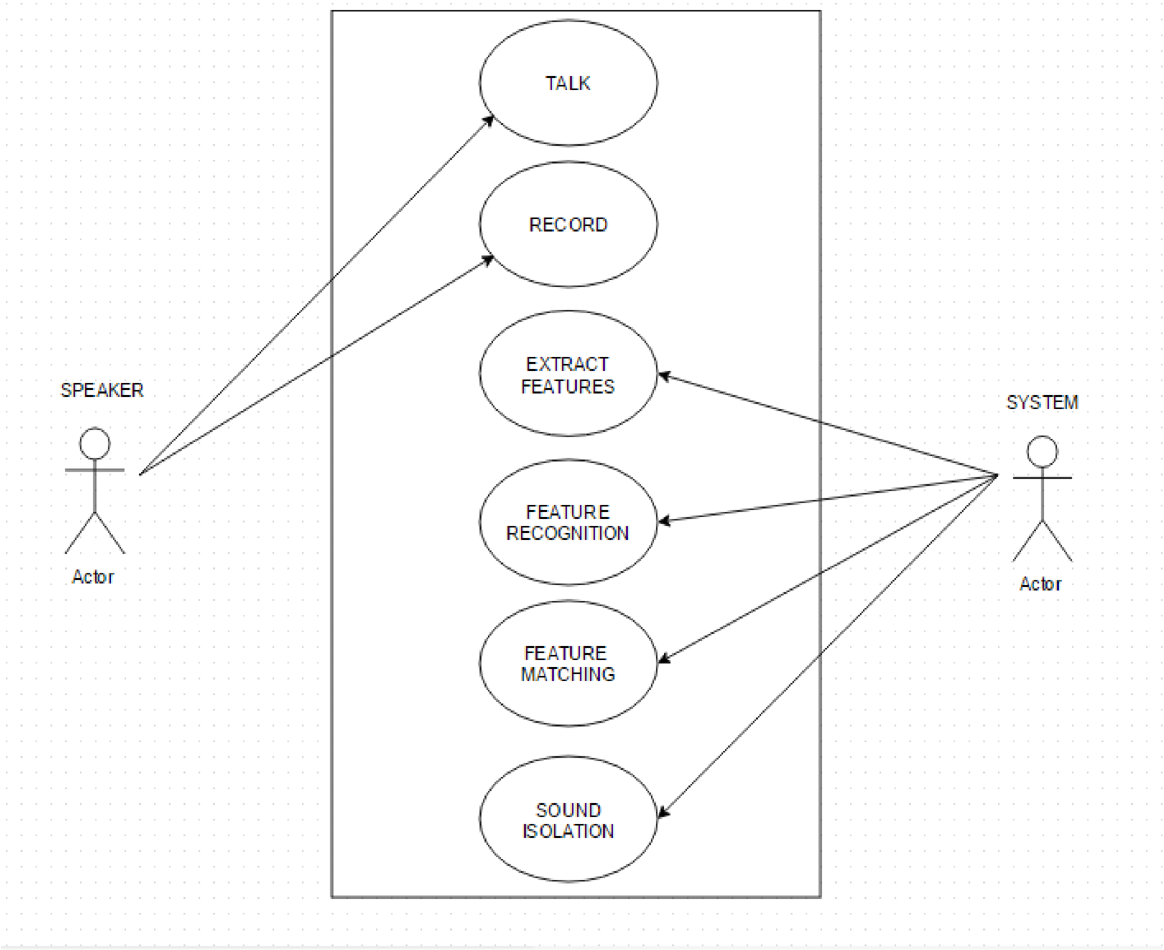


Figure 3.4 Use Case Diagram

3.3 Methodology and analysis

3.3.1 Front End Processing

The front-end processing stage is performed in Matlab. Front-end processing is vital as it off-loads the computational data from the rest of the system. The front-end processor is used to extract useful details about the speech data from the input audio file (in this case it extracts MFCCs). The speech signal is passed into the FEP, and Matlab reads the input speech from a audio file formats depending on what functions are used in Matlab to open such files. Next, the Frame length, sampling frequency, and number of channels which are required are passed into a function to compute a matrix of mel filters which will be used with the input signal to analyse the data. The characteristics of speech are relatively stationary over short periods of time, over longer periods of time an overflow of speech input makes extracting data more difficult. The input signal is then split into approximately 30ms frame lengths.

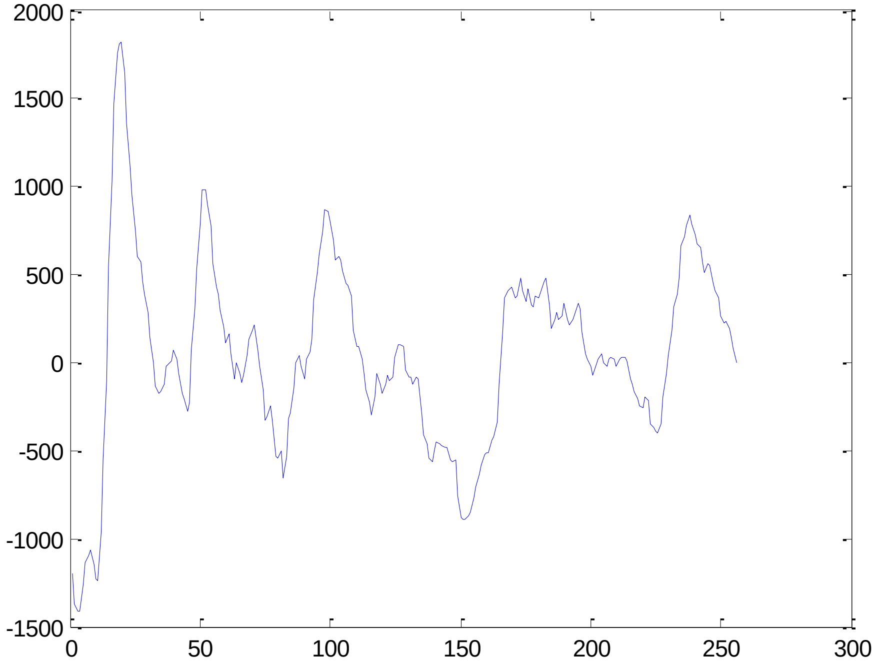


Figure 3.5 30ms Frame

The next step is to apply Hamming windowing to minimize the signal discontinuities at the beginning and end of each frame. This helps minimize the spectral distortion by using the window to taper the signal to zero at the beginning and the end of each frame.

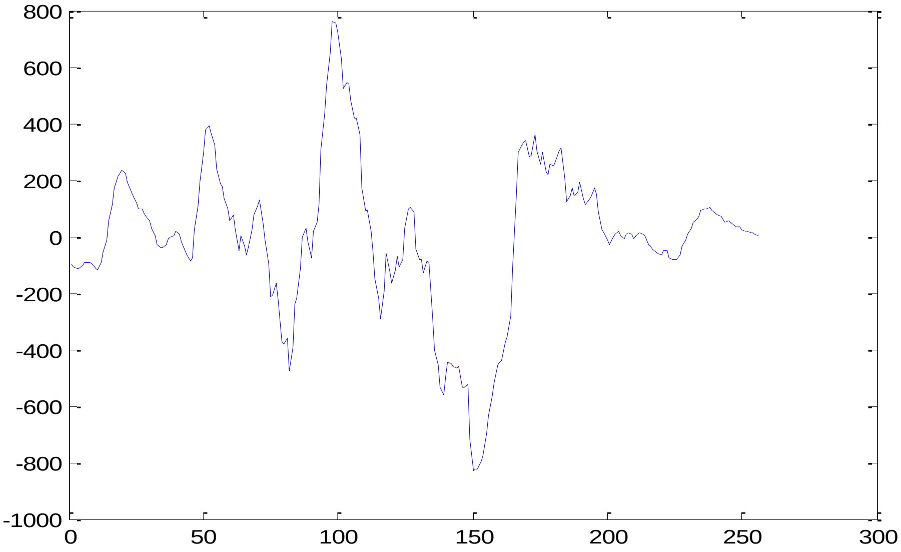


Figure 3.6 Signal with Hamming windowing

The signal is then converted to the frequency domain using a Fast Fourier transform. The first 128 frames are taken and plotted against the MFCC filter bank.

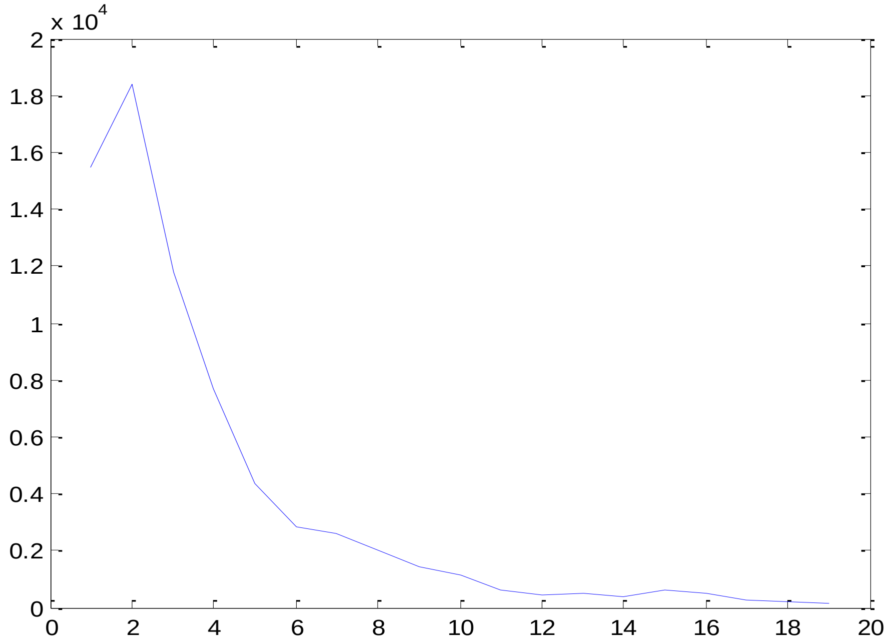


Figure 3.7 FFT of Signal analysed against MFCC

3.3.2 Mel Frequency Cepstral Coefficients (MFCCs)

Step 1**- Frame Blocking**

In this step the continuous speech signal is blocked into frames of *N* samples, with adjacent frames being separated by *M* (*M* < *N*). The first frame consists of the first *N* samples. The second frame begins *M* samples after the first frame, and overlaps it by *N* - *M* samples and Similarly, the third frame begins 2M samples after the first frame (or M samples after the second frame) and overlaps it by N - 2M samples. This process continues until all the speech is accounted for within one or more frames. Typical values for *N* and *M* are *N* = 256 and *M* = 100.

Step 2-**Windowing**

The next step in the processing is to window each individual frame so as to minimize the signal discontinuities at the beginning and end of each frame. The concept here is to minimize the spectral distortion by using the window to taper the signal to zero at the beginning and end of each frame. If we define the window as w(n), 0<n<N-1, where *N* is the number of samples in each frame, then the result of windowing is the signal

$y_l(n) = x_l(n)w(n),\ 0 \le n \le N-1$

Typically the *Hamming* window is used, which has the form:

$$w(n)=0.54-0.46\cos\left(\frac{2\pi n}{N-1}\right),\ 0\le n\le N-1$$

Step 3-**Fast Fourier transform**

The next processing step is the Fast Fourier Transform, which converts each frame of *N* samples from the time domain into the frequency domain. The FFT is a fast algorithm to implement the Discrete Fourier Transform (DFT), which is defined on the set of *N* samples {*xn*}, as follows:

$$X_k=\sum_{n=0}^{N-1} x_n e^{-j2\pi kn/N} \quad ,k=0,1,2\ldots.N-1$$

Step 4- **Mel-frequency Wrapping**

In sound processing, MFCC's are based on the known variation of the human ear's critical bandwidths. It is derived from the Fourier Transform of the audio clip. In this technique the frequency bands are positioned logarithmically, whereas in the Fourier Transform the frequency bands are not positioned logarithmically. As the frequency bands are positioned logarithmically in MFCC, it approximates the human system response more closely than any other system.These coefficients allow better processing of data. Each tone with an actual frequency t measured in Hz, a subjective pitch is measured on a scale called the 'Mel Scale'. The Mel frequency scale is linear frequency spacing below 1000 Hz and logarithmic spacing above 1 kHz. As a reference point, the pitch of a 1 kHz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 Mels. Therefore we can use the following formula to determine the Mels for a given frequency f in Hz. Mel (*f*) = 2595*log10 (1 + *f*/700).

To obtain the subjective spectrum we use a filter bank which is spaced uniformly on the Mel scale is described on the figure below. That filter bank has a triangular bandpass frequency response, and the spacing as well as the bandwidth is determined by a constant Mel frequency interval.

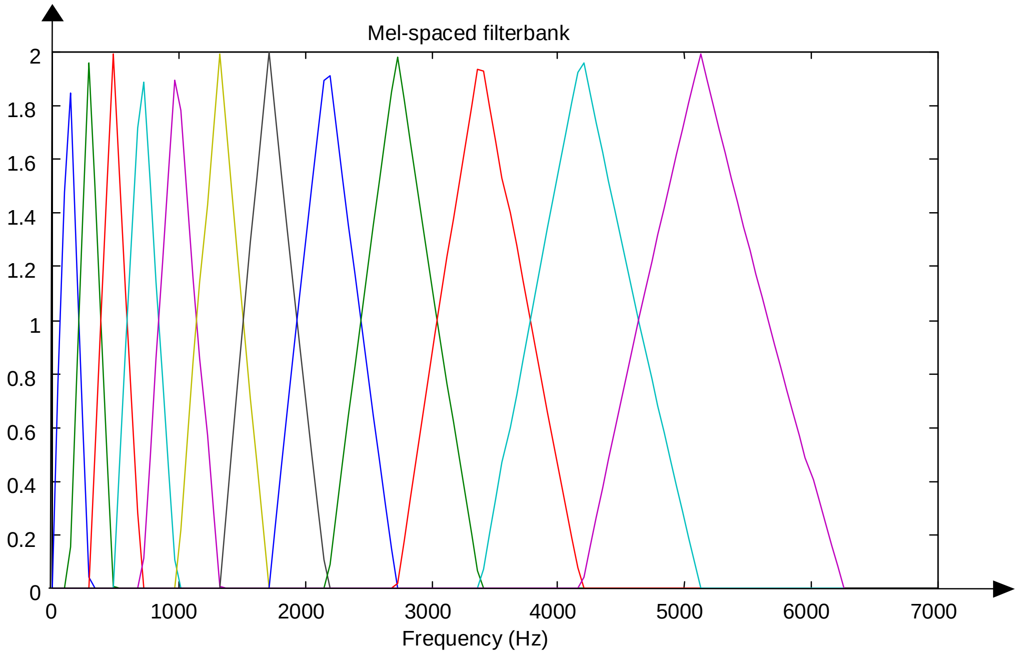


Figure 3.8 An example of Mel-spaced filter bank

Step 5-**Cepstrum**

Cepstrum name was derived from the spectrum by reversing the first four letters of spectrum. We can say Cepstrum is the Fourier Transformer of the log with unwrapped phase of the Fourier Transformer. Mathematically we can say Cepstrum of signal = FT (log (FT (the signal)) +j6.28m), Where m is the integer required to properly unwrap the angle or imaginary part of the complex log function. Algorithmically we can say – Signal - FT - log - phase unwrapping - FT -Cepstrum.We can calculate the Cepstrum by many ways. Some of them need a phase-warping algorithm, others do not.In this final step log Mel spectrum is converted back to time. The result is called the Mel Frequency Cepstrum Coefficients (MFCC).The discrete cosine transform is done for transforming the Mel coefficients back to time domain.

$$\tilde{c} = \sum_{k=1}^{K} (\log \tilde{S}k)\text{Cos}\left[n\left(k-\frac{1}{2}\right)\frac{\pi}{K}\right], \quad \text{n=0,1,….K-1}$$

3.3.3 Back-Propagation Training

Our network consists of an input layer, one hidden layer and an output layer. We use a set of Mel Frequency Cepstral coefficients as input for the neural network. The input neurons correspond to the features extracted per frame i.e., 20. In our design we put all these input ranges in an Input Layer matrix.

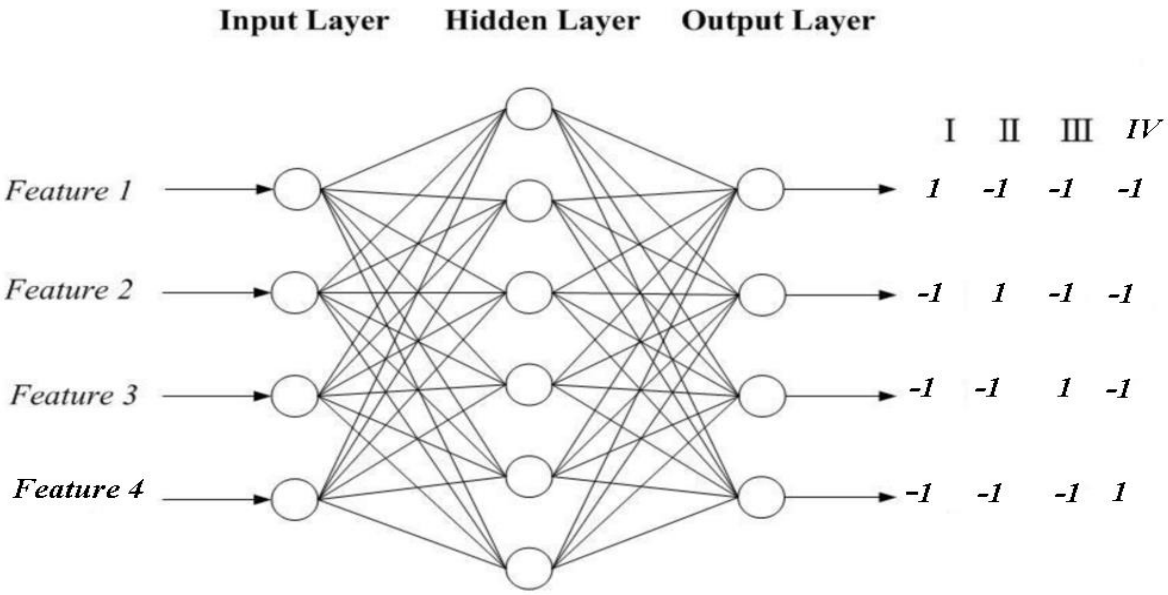


Figure 3.9 Back Propagation Neural Networks Methodology

The basic neural network having 4 output is as shown in figure 3. If the identified speaker is speaker 1, the first output neuron gives an output of 1 and the rest output neurons give an output of -1. Similarly, for second, third and fourth speaker, output neurons 2, 3 and 4 are fired and they give an output of 1 respectively.

Chapter 4 Implementation

We have implemented the speaker identification system in the following environment.

4.1 Software Requirements

4.1.1 MATLAB

MATLAB (**mat**rix **lab**oratory) is multi-paradigm numerical computing environment and fourth generation language Developed by Mathworks, MATLAB allows matrix manipulation ,plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, and Fortran. Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPAD symbolic engine, allowing access to symbolic computing capabilities. Package, Simulink, adds graphical multi-domain simulation and Model-Based Design additional for dynamic and embedded systems.

4.1.2 PRAAT

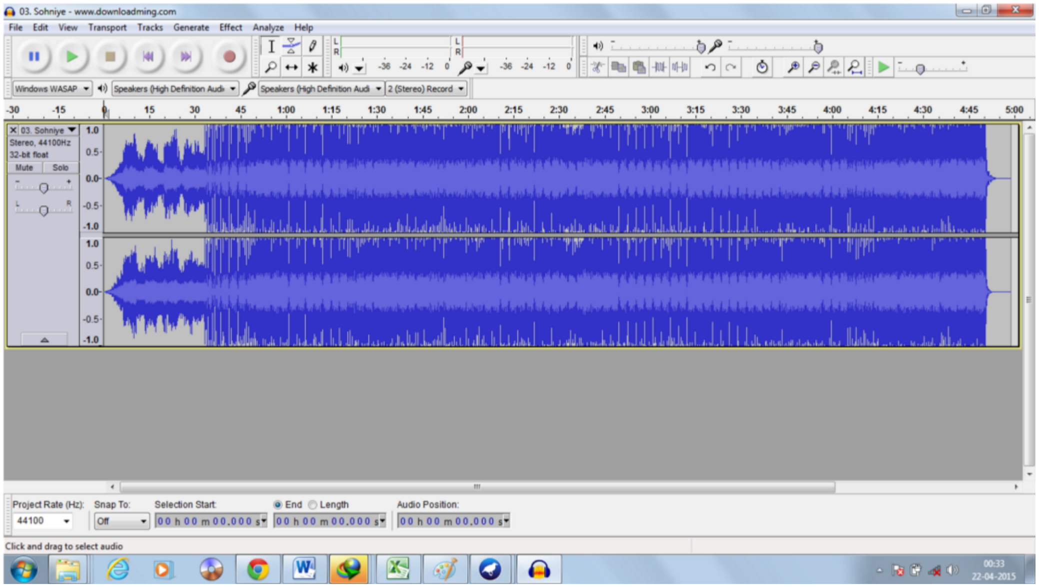
This is a freeware program for the analysis and reconstruction of acoustic speech signals. It offers a wide range of standard and non-standard procedures, including spectrographic analysis, articulatory synthesis, and neural networks.

4.1.3 VOICEBOX

VOICEBOX is a speech processing toolbox consists of MATLAB routines that are Maintained by and mostly written by Mike Brookes, Department of Electrical & Electronic Engineering, Imperial College, Exhibition Road, London SW7 2BT, UK. Several of the routines require MATLAB V6.5 or above and require (normally slight) modification to work with earlier versions.

4.1.4 AUDACITY

Audacity is the name of popular open source multilingual audio editor and recorder software that is used to record and edit sounds. It is free and works on Windows, Mac OS X, GNU/Linux and other operating system. It is basically an audio editor tool. It has been used in this project, in order to reduce the intensity of noise from the song, which makes the vocal part more clear and efficient for further processing.



Windows WASAP | Speakers (High Definition Audi | Speakers (High Definition Audi | 2 (Stereo) Record

Stereo, 44100Hz
32-bit float

Project Rate (Hz): 44100
Snap To: Off
Selection Start: 00 h 00 m 00.000 s
● End ○ Length: 00 h 00 m 00.000 s
Audio Position: 00 h 00 m 00.000 s

Click and drag to select audio

Figure 4.1 Audacity View diagram

4.2 Hardware Requirements

The system comprises of mostly software portion but has some hardware involved too. The hardware that has been used is:

- Microphone
- P.C.

4.2.1 Microphones

A quality microphone is a key, when utilizing Speaker Recognition. In most cases, a desktop microphone just won't do the job. Choice of microphone is critical for the success of Speaker Recognition system. Hand held microphones are also not the best choice as they can be cumbersome to pick up all the time.

4.2.2 Computers/Processors

Speaker recognition applications can be heavily dependent on processing speed. This is because a large amount of digital filtering and signal processing can take place in speaker recognition.

Because of the processing required, most software packages list their minimum requirements. The processor with 1.6 GHz and 3 GB RAM has been used in the system implementation.

Chapter 5 Results and Conclusion

5.1 Result

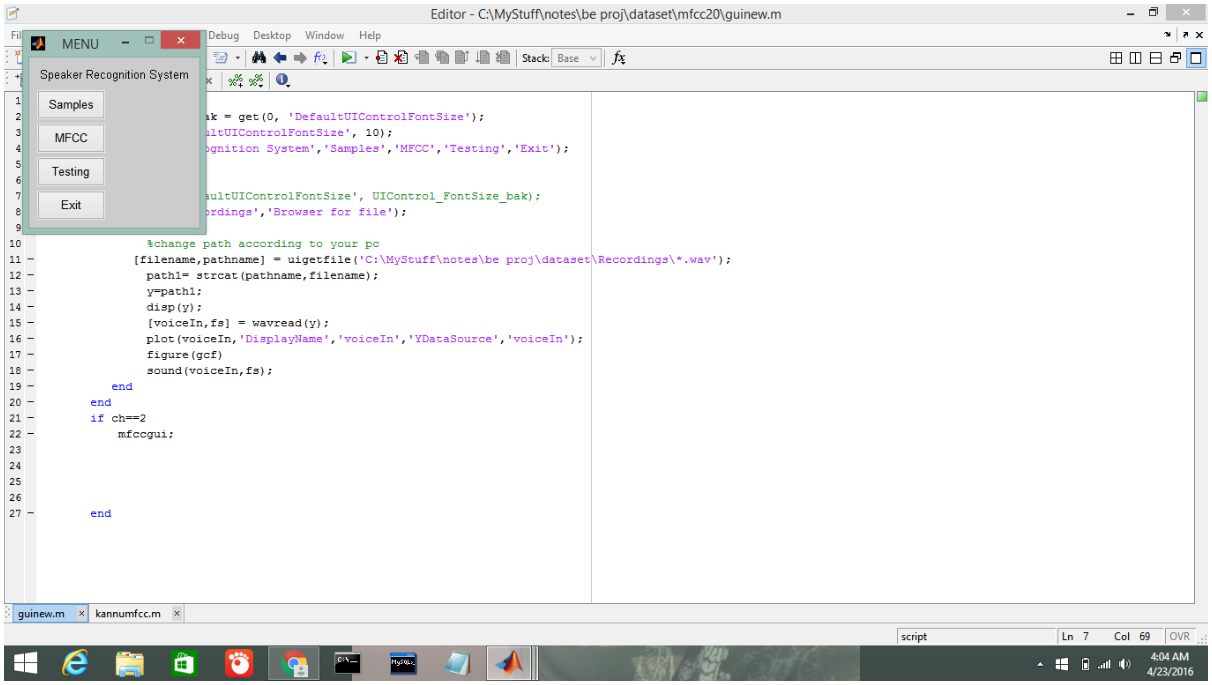
5.1.1 Analysis

The system was implemented in Matlab and trained using voice signals by Audacity, Speech analysis software at 16000 Hz. Figure shows the training set, and average accuracy.

Sr. No	Training Set(no. of recordings)	Test Set(no. of recordings)	Accuracy (%)
1. Single Speaker	3	30	90(27 recordings identified correctly)
2. Multi-Speaker	3	5	100(5 recordings identified correctly)

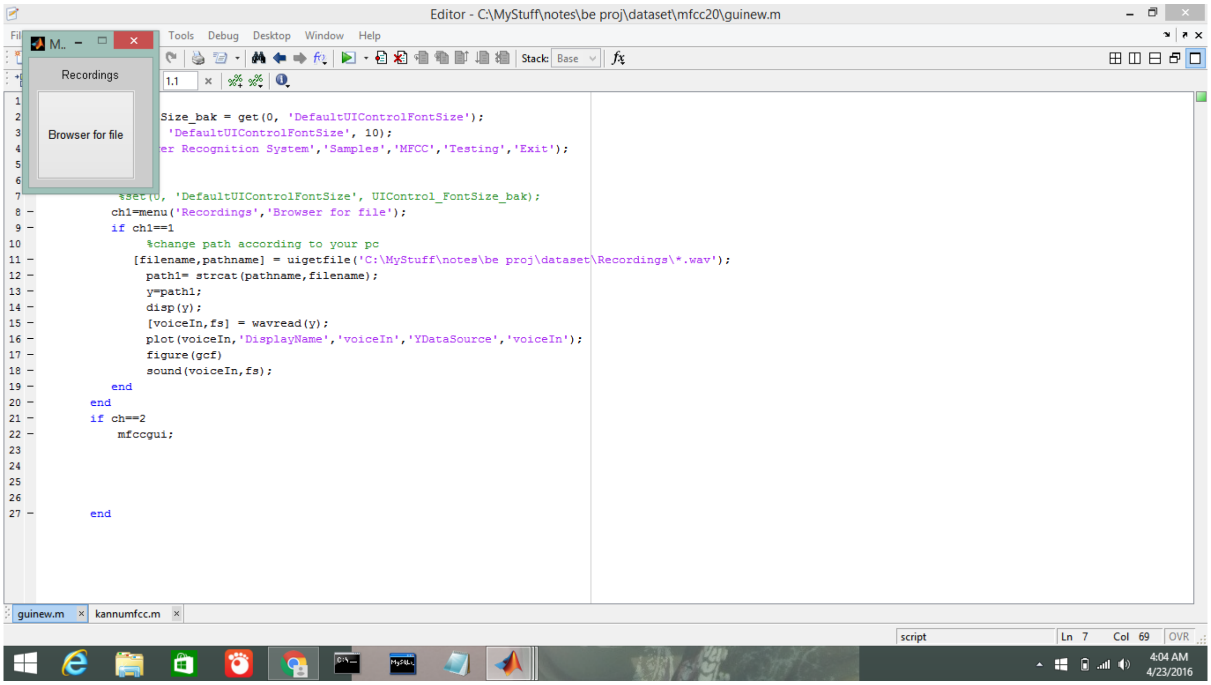
Table 5.1 Result

5.1.2 GUI and Screenshots



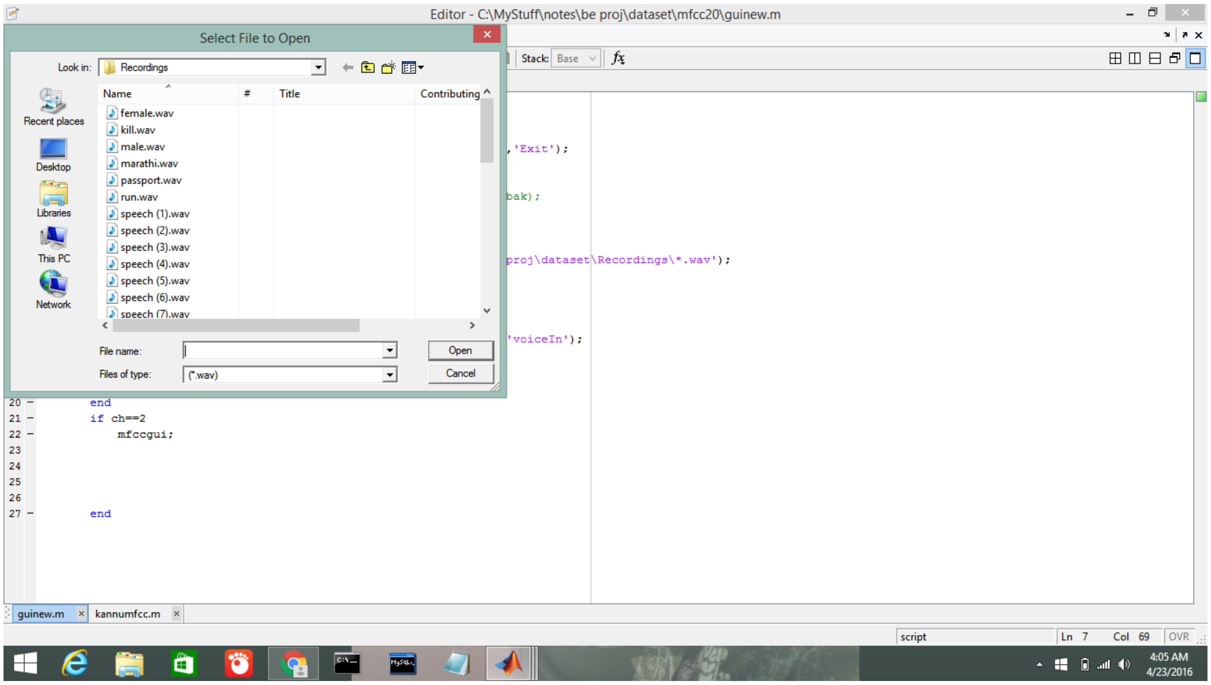
```
ak = get(0, 'DefaultUIControlFontSize');
iltUIControlFontSize', 10);
ognition System','Samples','MFCC','Testing','Exit');


aultUIControlFontSize', UIControl_FontSize_bak);
ordings','Browser for file');

            %change path according to your pc
          [filename,pathname] = uigetfile('C:\MyStuff\notes\be proj\dataset\Recordings\*.wav');
            path1= strcat(pathname,filename);
            y=path1;
            disp(y);
            [voiceIn,fs] = wavread(y);
            plot(voiceIn,'DisplayName','voiceIn','YDataSource','voiceIn');
            figure(gcf)
            sound(voiceIn,fs);
        end
    end
    if ch==2
        mfccgui;




    end
```

Figure 5.1 Screenshot of GUI main screen



```
Size_bak = get(0, 'DefaultUIControlFontSize');
 'DefaultUIControlFontSize', 10);
er Recognition System','Samples','MFCC','Testing','Exit');


    %set(0, 'DefaultUIControlFontSize', UIControl_FontSize_bak);
   ch1=menu('Recordings','Browser for file');
   if ch1==1
        %change path according to your pc
     [filename,pathname] = uigetfile('C:\MyStuff\notes\be proj\dataset\Recordings\*.wav');
        path1= strcat(pathname,filename);
        y=path1;
        disp(y);
        [voiceIn,fs] = wavread(y);
        plot(voiceIn,'DisplayName','voiceIn','YDataSource','voiceIn');
        figure(gcf)
        sound(voiceIn,fs);
   end
end
if ch==2
    mfccgui;




end
```

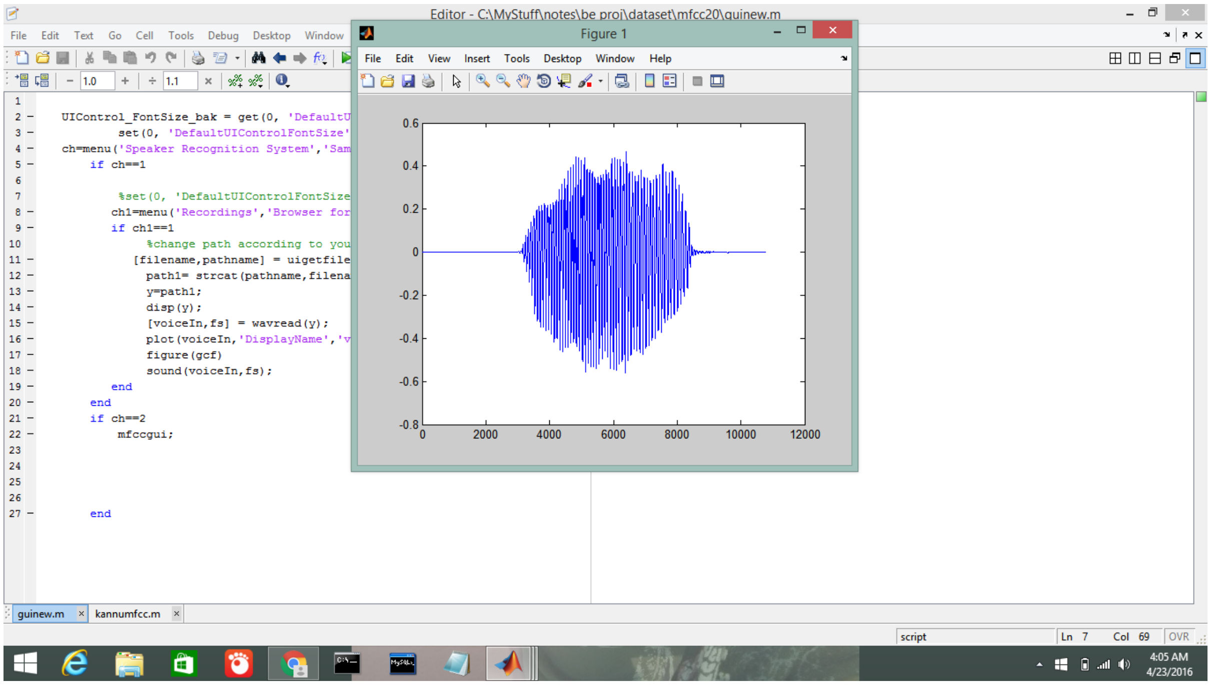
Figure 5.2 Screenshot for browsing audio file



female.wav
kill.wav
male.wav
marathi.wav
passport.wav
run.wav
speech (1).wav
speech (2).wav
speech (3).wav
speech (4).wav
speech (5).wav
speech (6).wav
speech (7).wav

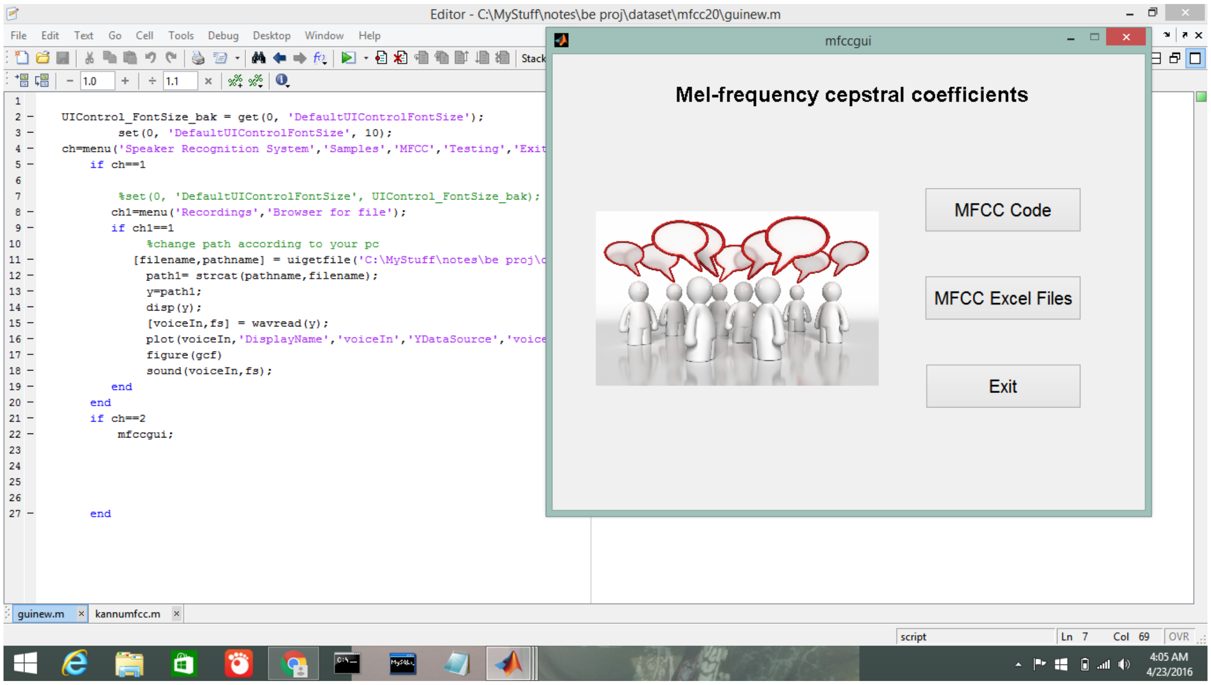
```
,'Exit');

bak);

proj\dataset\Recordings\*.wav');

'voiceIn');

end
if ch==2
    mfccgui;




end
```

Figure 5.3 Screenshot for selecting audio file


Editor - C:\MyStuff\notes\be proj\dataset\mfcc20\guinew.m
File Edit Text Go Cell Tools Debug Desktop Window
Figure 1
File Edit View Insert Tools Desktop Window Help
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
2000
4000
6000
8000
10000
12000
guinew.m
kannumfcc.m
script
Ln 7
Col 69
OVR
4:05 AM
4/23/2016

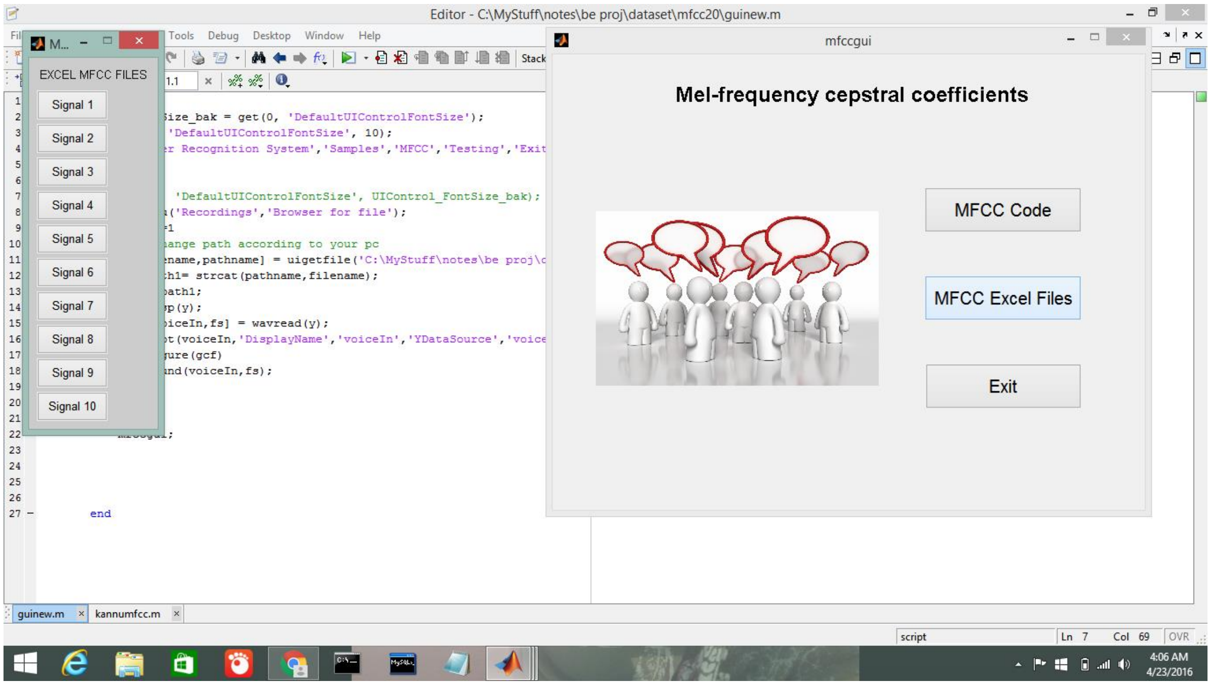
```
    UIControl_FontSize_bak = get(0, 'DefaultU
            set(0, 'DefaultUIControlFontSize'
    ch=menu('Speaker Recognition System','Sam
        if ch==1

            %set(0, 'DefaultUIControlFontSize
           ch1=menu('Recordings','Browser for
           if ch1==1
                %change path according to you
              [filename,pathname] = uigetfile
                path1= strcat(pathname,filena
                y=path1;
                disp(y);
                [voiceIn,fs] = wavread(y);
                plot(voiceIn,'DisplayName','v
                figure(gcf)
                sound(voiceIn,fs);
           end
        end
        if ch==2
            mfccgui;




        end
```

Figure 5.4 Screenshot of the waveform of the chosen audio file


Editor - C:\MyStuff\notes\be proj\dataset\mfcc20\guinew.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack
mfccgui
Mel-frequency cepstral coefficients
MFCC Code
MFCC Excel Files
Exit
guinew.m
kannumfcc.m
script
Ln 7
Col 69
OVR
4:05 AM
4/23/2016

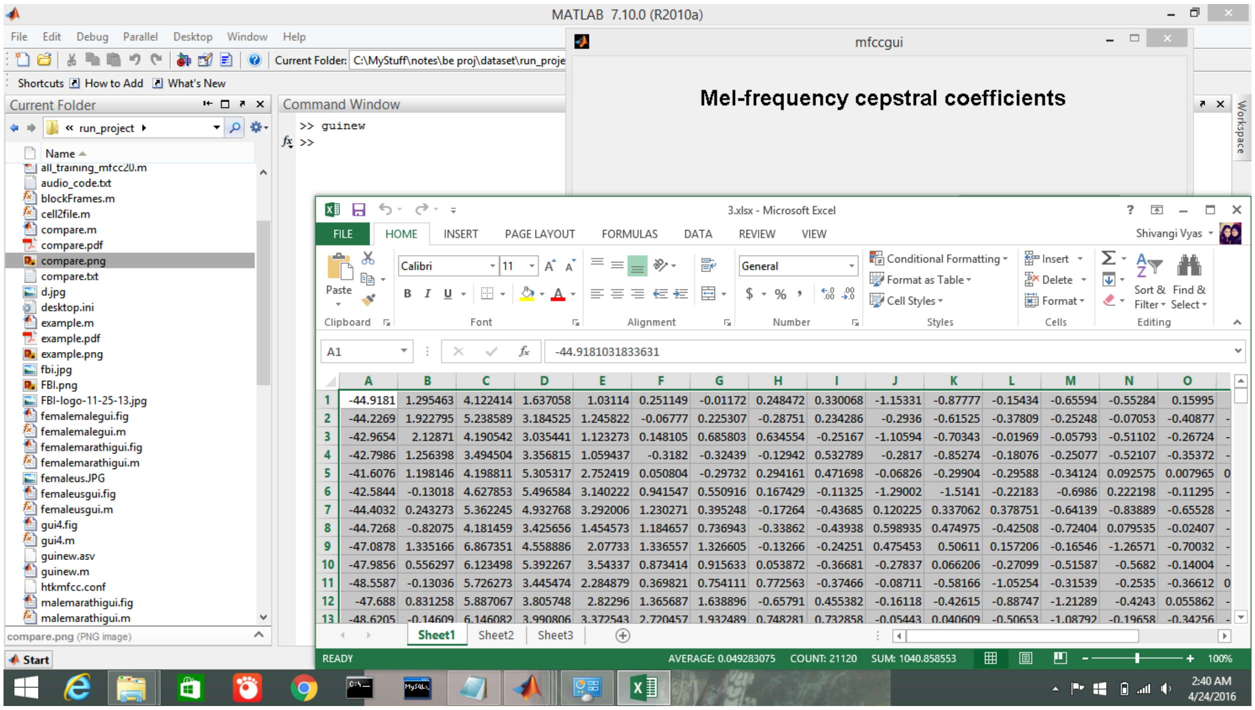
```
    UIControl_FontSize_bak = get(0, 'DefaultUIControlFontSize');
            set(0, 'DefaultUIControlFontSize', 10);
    ch=menu('Speaker Recognition System','Samples','MFCC','Testing','Exit
        if ch==1

            %set(0, 'DefaultUIControlFontSize', UIControl_FontSize_bak);
           ch1=menu('Recordings','Browser for file');
           if ch1==1
                %change path according to your pc
              [filename,pathname] = uigetfile('C:\MyStuff\notes\be proj\c
                path1= strcat(pathname,filename);
                y=path1;
                disp(y);
                [voiceIn,fs] = wavread(y);
                plot(voiceIn,'DisplayName','voiceIn','YDataSource','voice
                figure(gcf)
                sound(voiceIn,fs);
           end
        end
        if ch==2
            mfccgui;




        end
```

Figure 5.5 Screenshot for browsing MFCC



```
ize_bak = get(0, 'DefaultUIControlFontSize');
 'DefaultUIControlFontSize', 10);
r Recognition System','Samples','MFCC','Testing','Exit


 'DefaultUIControlFontSize', UIControl_FontSize_bak);
('Recordings','Browser for file');
1
ange path according to your pc
name,pathname] = uigetfile('C:\MyStuff\notes\be proj\c
h1= strcat(pathname,filename);
ath1;
p(y);
iceIn,fs] = wavread(y);
t(voiceIn,'DisplayName','voiceIn','YDataSource','voice
ure(gcf)
nd(voiceIn,fs);




      end
```

Figure 5.6 Screenshot for running MFCC of audio file

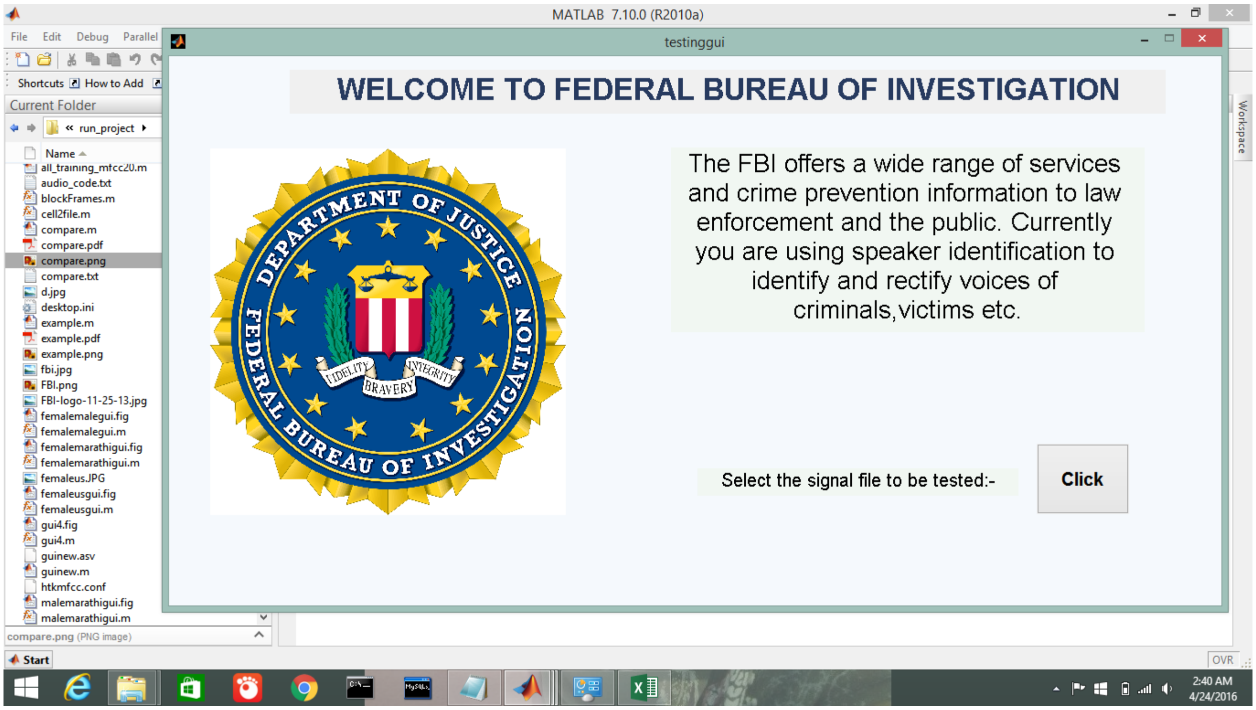


```
>> guinew
>>
```

A1 -44.9181031833631

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	-44.9181	1.295463	4.122414	1.637058	1.03114	0.251149	-0.01172	0.248472	0.330068	-1.15331	-0.87777	-0.15434	-0.65594	-0.55284	0.15995
2	-44.2269	1.922795	5.238589	3.184525	1.245822	-0.06777	0.225307	-0.28751	0.234286	-0.2936	-0.61525	-0.37809	-0.25248	-0.07053	-0.40877
3	-42.9654	2.12871	4.190542	3.035441	1.123273	0.148105	0.685803	0.634554	-0.25167	-1.10594	-0.70343	-0.01969	-0.05793	-0.51102	-0.26724
4	-42.7986	1.256398	3.494504	3.356815	1.059437	-0.3182	-0.32439	-0.12942	0.532789	-0.2817	-0.85274	-0.18076	-0.25077	-0.52107	-0.35372
5	-41.6076	1.198146	4.198811	5.305317	2.752419	0.050804	-0.29732	0.294161	0.471698	-0.06826	-0.29904	-0.29588	-0.34124	0.092575	0.007965
6	-42.5844	-0.13018	4.627853	5.496584	3.140222	0.941547	0.550916	0.167429	-0.11325	-1.29002	-1.5141	-0.22183	-0.6986	0.222198	-0.11295
7	-44.4032	0.243273	5.362245	4.932768	3.292006	1.230271	0.395248	-0.17264	-0.43685	0.120225	0.337062	0.378751	-0.64139	-0.83889	-0.65528
8	-44.7268	-0.82075	4.181459	3.425656	1.454573	1.184657	0.736943	-0.33862	-0.43938	0.598935	0.474975	-0.42508	-0.72404	0.079535	-0.02407
9	-47.0878	1.335166	6.867351	4.558886	2.07733	1.336557	1.326605	-0.13266	-0.24251	0.475453	0.50611	0.157206	-0.16546	-1.26571	-0.70032
10	-47.9856	0.556297	6.123498	5.392267	3.54337	0.873414	0.915633	0.053872	-0.36681	-0.27837	0.066206	-0.27099	-0.51587	-0.5682	-0.14004
11	-48.5587	-0.13036	5.726273	3.445474	2.284879	0.369821	0.754111	0.772563	-0.37466	-0.08711	-0.58166	-1.05254	-0.31539	-0.2535	-0.36612
12	-47.688	0.831258	5.887067	3.805748	2.82296	1.365687	1.638896	-0.65791	0.455382	-0.16118	-0.42615	-0.88747	-1.21289	-0.4243	0.055862
13	-48.6205	-0.14609	6.146082	3.990806	3.372543	2.720457	1.932489	0.748281	0.732858	-0.05443	0.040609	-0.50653	-1.08792	-0.19658	-0.34256

Figure 5.7 Screenshot displaying MFCC coefficients

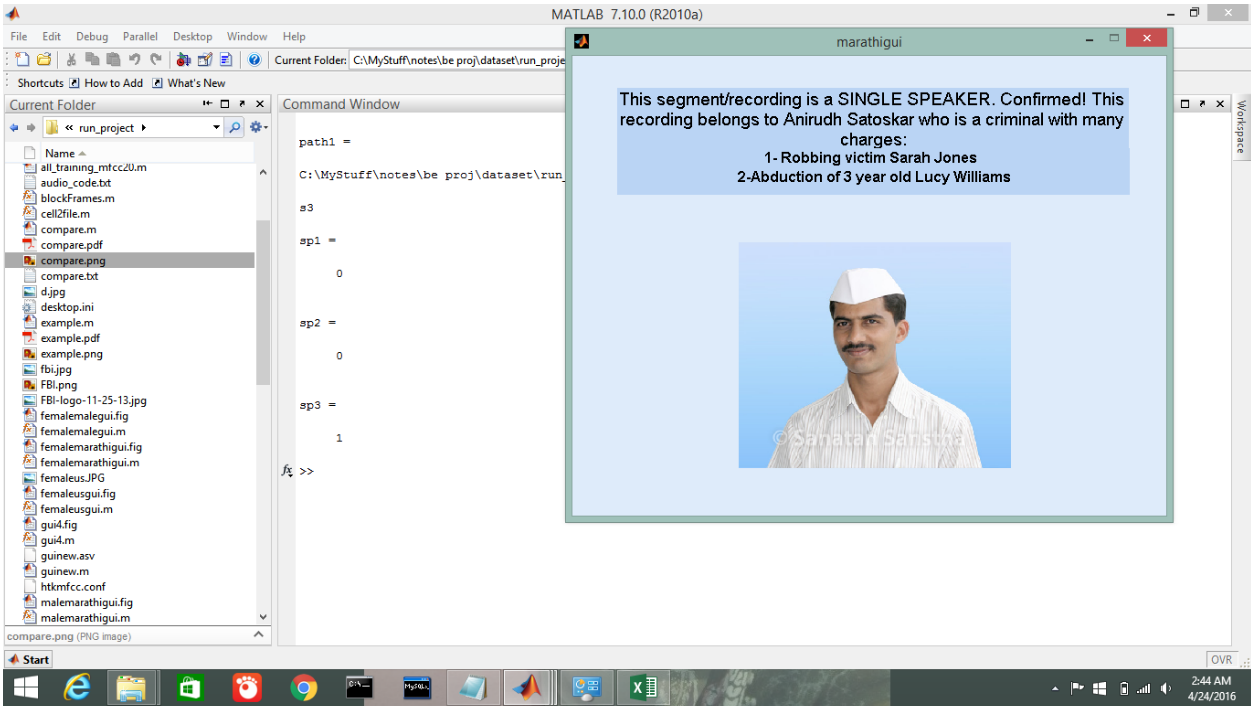


WELCOME TO FEDERAL BUREAU OF INVESTIGATION

The FBI offers a wide range of services and crime prevention information to law enforcement and the public. Currently you are using speaker identification to identify and rectify voices of criminals,victims etc.

Select the signal file to be tested:- Click

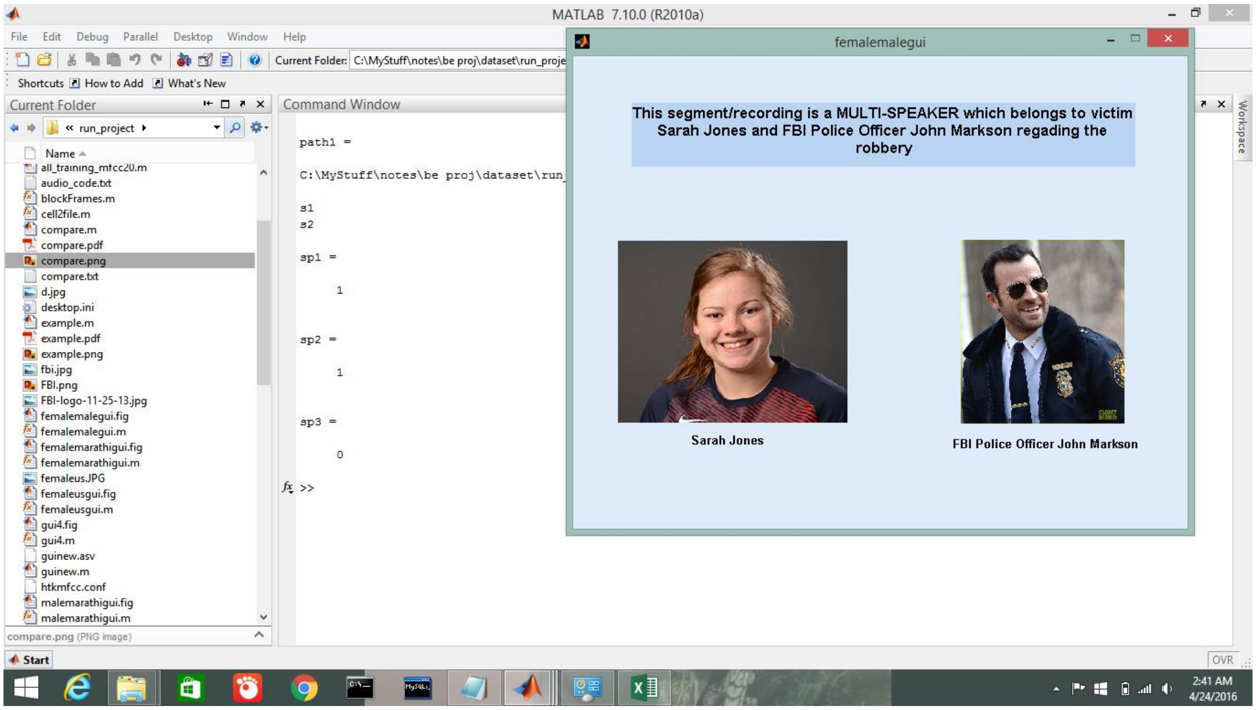
Figure 5.8 Screenshot of the testing option



```
path1 =

C:\MyStuff\notes\be proj\dataset\run_

s3

sp1 =

     0


sp2 =

     0


sp3 =

     1

fx >>
```

This segment/recording is a SINGLE SPEAKER. Confirmed! This recording belongs to Anirudh Satoskar who is a criminal with many charges:
1- Robbing victim Sarah Jones
2-Abduction of 3 year old Lucy Williams

Figure 5.9 Screenshot of single speaker result



```
path1 =

C:\MyStuff\notes\be proj\dataset\run_

s1
s2
sp1 =

     1


sp2 =

     1


sp3 =

     0

fx >>
```

This segment/recording is a MULTI-SPEAKER which belongs to victim Sarah Jones and FBI Police Officer John Markson regading the robbery

Figure 5.10 Screenshot of multi-speaker result

5.2 Conclusion:

5.2.1 Challenges faced while developing Speaker Identification System

1. Estimating the number of hidden neurons for training the system required a lot of trials.
2. Getting the adjusted weights for the EBPTA algorithm required a lot of training cases to be run.
3. Classifying the speech segments as single and multi-speaker with the trained weights required rigorous test cases to be run.
4. A lot of time was put into getting the error rate to 0.01 while training.

5.2.2 Proposed system which overcomes the above challenges

1. After a variety of train and test cases, the hidden neurons were found to be 25 and 27 for the single and multi-speaker segments respectively.

2. With the number of neurons obtained, the weight parameters were acquired and stored in the database.
3. The segments were classified as Single and Multi-Speaker segments successfully once the weights were captured in the database.
4. The average response time of the system was 79µs, once the error rate was brought down to zero.

Thus, by applying the Mel-Frequency Cepstral Coefficients technique used for feature extraction and using Error back Propagation training algorithm (EBPTA) for feature matching the speaker recognition system is made more robust and efficient and hence the performance of Speaker recognition system is improved.

References

[1] Wei-Ho Tsai and Shih-Jie Liao, “Speaker Identification in Overlapping Speech”,
Journal Of Information Science and Engineering paper published in 2010.

[2] Barry Arons, “A Review of The Cocktail Party Effect”, MIT Media Lab.

[3] Amit Sahoo and Ashish Panda, “Study of Speaker Recognition Systems”,National Institute of Technology,Rourkela, 2011.

[5] PPS Subhashini, Dr. M.Satya Sairam ,Dr. D Srinivasarao,”Speaker Identification with Back Propagation Neural Network Alogorithm”, International Journal of Engineering Trends and Technology paper published in 2014.

ACKNOWLEDGEMENTS

We are highly indebted to Thadomal Shahani Engineering College for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in making the project.

We would like to express our heartfelt gratitude towards our guide Prof.Shanthi Therese for her kind co-operation and encouragement which helped us in completion of this project. We thank her for giving us the liberty to be creative and innovative while implementing our project.

We thank our Head of Department (Information Technology) Mr. Arun Kulkarni for his cooperation and unconditional support. As our teacher he provided us with his useful insights and extended a helping hand whenever it was required.

We would like to express our special gratitude and thanks to faculty of Information Technology Department forgiving us such attention and time. Our thanks and appreciations also go to our colleagues in developing the project and people who have willingly helped us out with their abilities.

Manthan Thakker

Prachi Ved

Shivangi Vyas

Sahil Kathpal