SPEAKER VERIFICATION OVER VARIOUS CHANNELS

A PROJECT REPORT

Submitted by

K.SENTHIL KUMAR

S.SHYAM SUNDAR

B.VIGNESH

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

SRI SIVASUBRAMANIYA NADAR COLLEGE OF ENGINEERING
KALAVAKKAM – 603 110

ANNA UNIVERSITY :: CHENNAI 600 025
APRIL 2013

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “Speaker verification over various channels” is the bonafide work of “**K.SENTHIL KUMAR (31509104094), S.SHYAM SUNDAR (31509104097), B.VIGNESH (31509104111)”** who carried out the project work under my supervision.

SIGNATURE

Dr. CHITRA BABU
HEAD OF THE DEPARTMENT

Department of Computer Science
and Engineering,
Sri Sivasubramaniya Nadar
College of Engineering,
SSN Nagar, Kalavakkam - 603 110

SIGNATURE

Ms. B. BHARATHI
SUPERVISOR

ASSISTANT PROFESSOR

Department of Computer Science
and Engineering,
Sri Sivasubramaniya Nadar
College of Engineering,
SSN Nagar, Kalavakkam - 603 110

Submitted for the Viva-voce examination held _________________

INTERNAL EXAMINER

Date:

Official Seal:

EXTERNAL EXAMINER

Date:

ACKNOWLEDGEMENT

We are honored to acknowledge the below mentioned people who have been instrumental in the completion of this project.

We record our sincere thanks to our guide **Ms. B. BHARATHI**, Assistant Professor, Department of Computer Science and Engineering, for being instrumental in the completion of our project with her exemplary guidance.

We extend our heartfelt thanks to our Project Coordinator, **Ms. S. KAVITHA**, who provided constant support and valuable suggestions for our project.

We are deeply indebted to our beloved Head of the Department, **Dr. CHITRA BABU**, Department of Computer Science and Engineering, who moulds us both technically and morally for achieving greater success in life.

We are grateful to our Principal, **Dr. S. SALIVAHANAN**, SSN College of Engineering, for providing us an excellent environment to carry out our course successfully.

We express our deep respect to PADMA BHUSHAN **Dr. SHIV NADAR**, Chairman, SSN Institutions for working on a great mission and vision and providing excellent infrastructure.

We take this opportunity to extend our deep appreciation and gratitude to all our friends and family for their moral support and encouragement in all that we attempt to achieve.

ABSTRACT

Speaker verification is the verification of the person who is speaking by characteristics of their voices. Speaker verification over various channels involves recording the voice of the speaker through various channels. Text-independent systems are most often used for speaker verification as they require very little if any cooperation by the speaker. In this case the text during enrollment and test is different. In text independent systems both acoustics and speech analysis techniques are used.

In this project a new architecture for text independent speaker verification from samples of speech recorded over various channels is used. We give a full account of the algorithms needed to carry out a joint factor analysis of speaker variability in a training set in which each speaker is recorded over many different channels. A GMM model is estimated for each target speaker and these GMMs are used along with Joint Factor Analysis to make the speaker identification decision.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	DESCRIPTION
ASR	Automatic Speech Recognition
HMM	Hidden Markov Model
GUI	Graphical User Interface
GMM	Gaussian Mixture Model
SVM	Support Vector Machine
HTK	Hidden Markov Toolkit
PLP	Perceptually based Linear Predictive analysis
LPC	Linear Predictive Coding
DET	Detection Error Tradeoff
DCT	Discrete Cosine Transform
FFT	Fast Fourier Transform
JFA	Joint Factor Analysis
MFCC	Mel Frequency Cepstral Coefficients

LIST OF SYMBOLS

SYMBOL	DESCRIPTION
Σ	Summation
γ_t	Posterior probability
Σ^{-1}	Transposed mean of posterior distribution

CHAPTER 1

INTRODUCTION

1.1 SPEECH PROCESSING

Speech processing is the study of speech signals and the processing methods of these signals. The signals are usually processed in a digital representation, so speech processing can be regarded as a special case of digital signal processing, applied to speech signal. Aspects of speech processing includes the acquisition, manipulation, storage, transfer and output of digital speech signals.

It is also closely tied to natural language processing (NLP), as its input can come from / output can go to NLP applications. E.g. text-to-speech synthesis may use a syntactic parser on its input text and speech recognition's output may be used by e.g. information extraction techniques. The main applications of speech processing are the recognition, synthesis and compression of human speech.

Speech processing includes the following areas of study:

- **Speech recognition** (also called voice recognition), which deals with analysis of the linguistic content of a speech signal and its conversion into a computer-readable format.
- **Speaker recognition**, where the aim is to recognize the identity of the speaker.
- Speech coding, a specialized form of data compression, is important in the telecommunication area.
- **Voice analysis** for medical purposes, such as analysis of vocal loading and dysfunction of the vocal cords.

- **Speech synthesis**: the artificial synthesis of speech, which usually means computer-generated speech. Advances in this area improve the computer's usability for the visually impaired.
- **Speech enhancement**: enhancing the intelligibility and/or perceptual quality of a speech signal, like audio noise reduction for audio signals.
- **Speech compression** is important in the telecommunications area for increasing the amount of information which can be transferred, stored, or heard, for a given set of time and space constraints.
- **Speaker diarization** is the process of determining who spoke when in a signal.

1.2 SPEAKER RECOGNITION

Speaker recognition is the identification of the person who is speaking by characteristics of their voices, also called voice recognition. There is a difference between speaker recognition and speech recognition where, in speaker recognition who is speaking is recognized and in speech recognition what is being said is recognized. Recognizing the speaker can simply the task of translating speech in systems that have been trained on specific person’s voice or it can be used to authenticate or verify the identity of a speaker as a part of security process. Speaker recognition uses the acoustic features of the speech that have been found to differ between individuals. Speaker recognition refers to two fields:

Speaker Verification - If the speaker claims to be of a certain identity and the voice is used to verify this claim

Speaker Identification - On the other hand, identification is the task of determining an unknown speaker’s identity.

1.2.1 PHASES IN SPEAKER RECOGNITION SYSTEM

Speaker Recognition System has two phases:

Training Phase

During enrolment, the speaker’s voice is recorded and typically a number of features are extracted to form a voice print.

Testing Phase

In the verification phase, a speech sample or utterance is compared against a previously created voice print. For identification systems, the utterance is compared against multiple voice prints in order to determine the best match while verification systems compare an utterance against a single voice print which makes them faster than identification.

1.2.2 CLASSIFICATION OF SPEAKER RECOGNITION SYSTEMS

Speaker recognition systems fall into two categories:

Text-Dependent Speaker Recognition System

If the text must be the same for enrollment and verification this is called text-dependent recognition. In a text-dependent system, prompts can either be common across all speakers (e.g. a common pass phrase) or unique. In addition, the use of shared-secrets (e.g. passwords and PINs) or knowledge-based information can be employed in order to create a multi-factor authentication scenario.

Text-Independent Speaker Recognition System

Text-independent systems are most often used for speaker identification as they require very little if any cooperation by the speaker. In this case the text during enrollment and test is different. In fact, the enrollment may happen without the user's knowledge, as in the case for many forensic applications. As text-independent technologies do not compare what was said at enrollment and verification, verification applications tend to also employ speech recognition to determine what the user is saying at the point of authentication. In text independent systems both acoustics and speech analysis techniques are used.

1.2.3 ISSUES IN SPEAKER RECOGNITION

1) Extreme emotional states

For example, the sympathetic arousal associated with an anger state often produce changes in respiration and an increase in muscle tension, which influence the vibration of the vocal folds and vocal tract shape, affecting the acoustic characteristics of the speech thus making speaker identification difficult.

2) Microphone placement

There are several classes of microphone placement for recording and amplification like close miking and distant miking.

3) Poor or inconsistent room acoustics

Acoustic parameters are affected by both room properties and distance between source and mike.

4) Sickness

Sickness like cold can affect the vocal tract and produce a different voice. This makes speaker identification difficult.

5) Aging

With ageing, the subsystems which make up the human speech production system undergo progressive physiological change, bringing about significant changes in the voice. This makes the speaker identification process more difficult.

6) Channel Mismatch

Channel mismatch between enrollment and test data is one of the top performance degrading factors in speaker recognition applications. Mismatch is particularly true over public telephone networks, where input speech data is collected over different handsets and transmitted over different channels from one trial to the next.

This project addresses the various issues and problems caused by channel mismatch.

1.2.4 APPLICATIONS OF SPEAKER RECOGNITION

- **Speaker Recognition for Surveillance -** Security agencies have several means of collecting information. One of these is electronic eavesdropping of telephone and radio conversations. As this result in high quantities of data, filter mechanisms must be applied in order to find the relevant information. One of these filters may be the recognition of target speakers that are of interest for the service.

- **Forensic Speaker Recognition -** Proving the identity of a recorded voice can help to convict a criminal or discharge an innocent in court. Although this task is probably not performed by a completely automatic speaker recognition system, signal processing techniques can be of use in this field nevertheless.

- **Making reservations -** Voice recognition could also be used for computers for making airline and hotel reservations. A user can simply state his needs, to make reservation, cancel reservation, or make enquiries about schedule.

- **Telephony and other domains -** ASR in the field of telephony is now common place and in the field of computer gaming and simulation is becoming more widespread some PBX/Voice Mail systems allow callers to speak commands instead of pressing buttons to send specific tones.

1.3SPEAKER VERIFICATION SYSTEM OVER VARIOUS CHANNELS

Speaker verification over various channels involves recording the voice of the speaker through various channels.

1.4 PROBLEM STATEMENT

To develop a method to perform speaker verification over various channels using generative models like Gaussian Mixture Models based on Universal Background Model (GMM-UBM) and Joint Factor Analysis (JFA).

Input - Speech signal containing voice input of the speaker who is to be verified.

Output - Speaker is accepted or rejected based on the obtained scores.

Objective - To verify the speaker who is speaking, over various channels.

1.5ORGANIZATION OF THE REPORT

This section describes about various chapters included in this report.

Chapter 2: Literature review

This chapter presents some background information on the various areas involved in this research. These include various speaker recognition techniques, feature extraction techniques and classifiers used in speech recognition systems. It gives introduction to Hidden Markov Toolkit.

Chapter 3: Proposed methodology

This chapter gives the complete architecture of the system. It explains the detailed design of training and testing phases. This section also describes about the hardware and software used to develop the system.

Chapter 4: Evaluation and results

This chapter explains about the results of various phases involved in the development of the system. It describes about the Graphical User Interface (GUI) that is used for testing the system. Performance of the system in different environments is explained in this section.

Chapter 5: Conclusions and future directions

This chapter summarizes the main components of this system as well as the major conclusions gathered from the results of the evaluation. This section also covers how the system can be further improved.

CHAPTER 2

LITERATURE REVIEW

Speaker verification over various channels involves verifying the true identity of the speaker by the characteristics of their voices across various channels. This verification process involves the following steps:

1) Feature Extraction
2) Classification
3) Joint Factor Analysis

2.1 FEATURE EXTRACTION

Feature extraction is an essential step in machine learning. If we want a machine to identify the spoken word, it will have to differentiate between different kinds of sound the way the humans perceive it. The point to be noted in case of humans is that although, one word spoken by different people produces different sound waves humans are able to identify the sound waves as same because of their common features .On the other hand two sounds which are different are perceived as different by humans.

A good feature extractor should extract the features and use them for further analysis and processing. The features represent the global statistics, which means the values were estimated over the whole utterance. The features are statistical functions derived from acoustical parameters. The acoustical parameters are pitch, intensity and formants. These features help in comparing the speech segments.

2.1.1 Linear predictive coding (LPC) analysis:

The basic idea behind the linear predictive coding (LPC) analysis is that a speech sample can be approximated as linear combination of past speech samples. By minimizing the sum of the squared differences (over a finite interval) between the actual speech samples and the linearly predicted ones, a unique set of predictor coefficients is determined. Speech is modelled as the output of linear, time-varying system excited by either quasi-periodic pulses (during voiced speech), or random noise (during unvoiced speech). The linear prediction method provides a robust, reliable, and accurate method for estimating the parameters that characterize the linear time-varying system representing vocal tract. The higher order cepstra are numerically quite small and this results in a very wide range of variances when going from the low to high cepstral coefficients.

ADVANTAGES

- More efficient since LPC doesn’t require explicit computation of cepstral coefficients.
- Cepstral coefficients are generally decorrelated and this allows diagonal covariances to be used in the HMMs.
- It better approximates the perception of the human ears to frequencies.

DISADVANTAGE:

The higher order cepstra are numerically quite small and this results in a very wide range of variances when going from the low to high cepstral coefficients.

2.1.2 Perceptually Based Linear Predictive Analysis (PLP)

PLP analysis models perceptually motivated auditory spectrum by a low order all pole function, using the autocorrelation LP technique. It involves two major steps: obtaining auditory spectrum, approximating the auditory spectrum by an all pole model. Auditory spectrum is derived from the speech waveform by critical-band filtering, equal loudness curve pre-emphasis, and intensity loudness root compression.

ADVANTAGE

Provides better performance to cross speaker ASR .

DISADVANTAGE

Just like most other short-term spectrum based techniques this method is vulnerable when the spectral values are modified by the frequency response of the communication channel.

2.1.3 Mel Frequency Cepstral Coefficients (MFCC)

The use of Mel frequency cepstral coefficients can be considered as another standard method for feature extraction. Mel Frequency Cepstral Coefficients (**MFCC**s) are features that represent audio. They are derived from a type of cepstral representation of the audio clip. The difference between the cepstrum and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum. This frequency warping can allow for better representation of sound, for example, in audio compression. This method reduces the frequency information of the speech signal into a small number of coefficients that emulate the separate critical bands in the basilar membrane of the ear, i.e., it tries to code the information in a similar way as the human cochlea does. Additionally, the logarithmic operation attempts to model loudness perception in the human auditory system. MFCC is a very simplified model of auditory processing, but it is easy and relatively fast to compute.

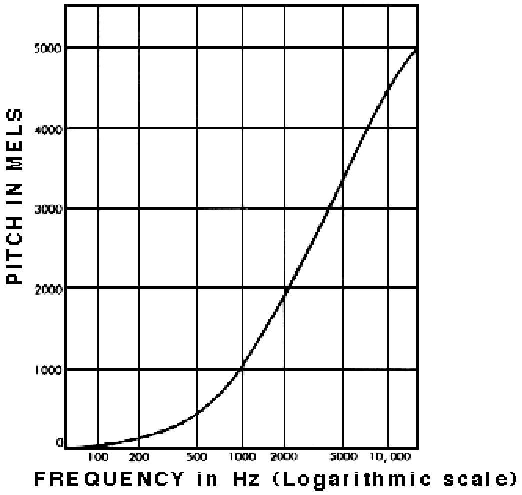
MFCCs are commonly derived as follows:

1. Take the Fourier transform of (a windowed excerpt of) a signal.
2. Map the powers of the spectrum obtained above onto the mel scale, using triangular overlapping windows.
3. Take the logs of the powers at each of the mel frequencies.
4. Take the discrete cosine transform of the list of mel log powers, as if it were a signal.

The MFCCs are the amplitudes of the resulting spectrum

2.1.3.1 MEL SCALE

The mel scale is a scale of pitches judged by listeners to be equal in distance from one another. The reference point between this scale and normal frequency measurement is defined by equating a 1000 Hz tone, 40 dB above the listener's threshold, with a pitch of 1000 mels. Below about 500 Hz the mel and hertz scales coincide; above that, larger and larger intervals are judged by listeners to produce equal pitch increments. As a result, four octaves on the hertz scale above 500 Hz are judged to comprise about two octaves on the mel scale. Many musicians and psychologists prefer a two-dimensional representation of pitch by chroma or tone colour and tone-height.



Source: Fundamentals of speech recognition

Figure 2.1: Mel Scale

The use of about 20 MFCC coefficients is common in Automatic Speech Recognition, although 10-12 coefficients are often considered to be sufficient for coding speech.

ADVANTAGE:

- Useful in classification of stress in speech.
- Delta and acceleration coefficients obtained from cepstral coefficient
- Provides temporal information and information about spectral changes from frame to frame.

DISADVANTAGE:

MFCC are not very robust in the presence of additive noise.

From the above survey, it is clear that MFCC analysis gives better performance than any other feature extraction techniques. Hence, MFCC features have been extracted from the wave file in our project.

2.2. CLASSIFIERS FOR SPEECH RECOGNITION

2.2.1 HIDDEN MARKOV MODEL (HMM)

Hidden Markov Model (HMM) is a finite state machine. The states of the model are represented as nodes and the transition are represented as edges. The difference in case of HMM is that the symbol does not uniquely identify a state. In HMM based speech recognition, it is assumed that the sequence of observed speech vectors corresponding to each word is generated by a Markov model as shown in the figure. A Markov model is a finite state machine which changes state once every time unit and each time *t* that a state *j* is entered, a speech vector o_t is generated from the probability density $b_j(o_t)$. Furthermore, the transition from state *i* to state *j* is also probabilistic and is governed by the discrete probability a_{ij}.

Source : HTK book

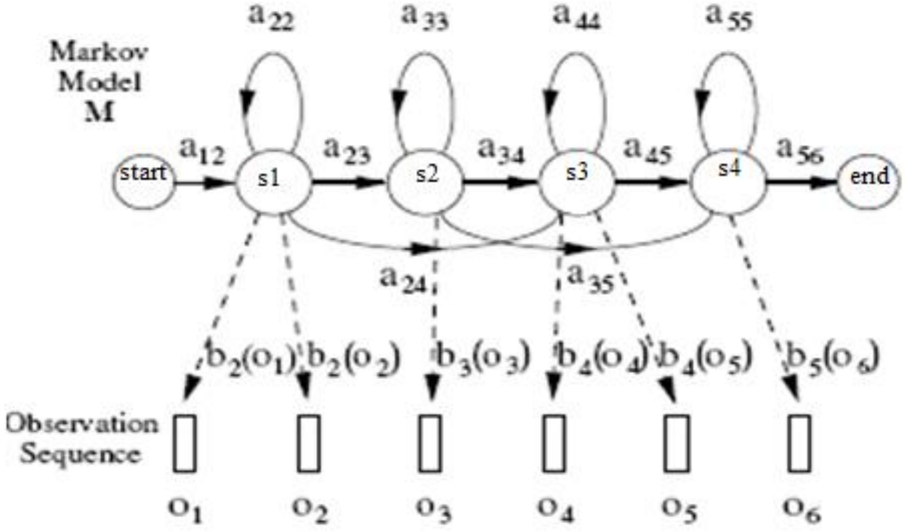


Figure 2.2 Diagramatic representation of HMM

Nodes denoted as circles are states. $O1$ to $O6$ are observations. Observation $O1$ takes us to states $S1$. a_{ij} defines the transition probability between S_i and S_j . It can be observed that the states also have self transitions. If we are at state S_1 and observation O_2 is observed, we can either decide to go to state S_2 or stay in state S_1. The decision is made depending on the probability of observation at both the states and the transition probability. [14]

Thus HMM Model is defined as:

$\lambda = (Q,O, A,B, \pi)$

Where Q is $\{q_i\}$ (all possible states)

O is $\{v_i\}$ (all possible observation)

A is $\{a_{ij}\}$ where $a_{ij} = P(X_{t+1} = q_j \,|X_t = q_i)$ (transition probabilities)

B is $\{b_i\}$ where $b_i(k) = P(O_t = v_k|X_t = q_{it})$ (observation probabilities of observation k at state i)

π is $\{\pi_i\}$ where $\pi_i = P(X_0 = q_i)$ (initial state probabilities)

X_t denotes the state at time t.

O_t denotes the observation at time t.

HMMs can be employed for classification in the following way: given several models, it is possible to determine the model which will produce a given sequence of observations with the highest probability. Thus, if for each class there is a model with the states, transitions and probabilities set appropriately, the Viterbi algorithm [2] can be used to calculate the model that most probably generated the sequence of observations.

ADVANTAGES

- The main advantage is its ease and availability of training algorithms for estimating the parameters of the models.
- The implementation of the recognition system using this method is easy.

DISADVANTAGE

- The number of parameters that need to be set in an HMM is huge
- The amount of data that is required to train an HMM is very large HMMs only use positive data to train.

2.2.2 SUPPORT VECTOR MACHINE (SVM)

A Support Vector Machine (SVM) is a decision-based prediction algorithm which can classify data into several groups. It is based on the concept of decision planes where the training data is mapped to a higher dimensional space and separated by a plane defining the two or more classes of data. Support Vector Machines have been used in a variety of classification tasks, such as isolated handwritten digit recognition, speaker identification, object recognition, face detection, and vowel classification. In a support vector machine, input vectors are mapped into a very high-dimension feature space through a non-linear mapping. Then a linear classification decision surface is constructed in the high-dimension feature space. This linear decision surface can take a non-linear form when it is mapped back into the original feature space.

ADVANTAGE

- SVMs deliver a unique solution, since the optimality problem is convex.

- SVM performs well on data sets that have many attributes, even if there are very few cases on which to train the model.

DISADVANTAGE

- Biggest limitation of the support vector approach lies in choice of the kernel
- SVMs have the high algorithmic complexity and extensive memory requirements of the required quadratic programming in large-scale tasks.
- Another limitation of SVMs, and machine learning algorithms in general, was that they assume that data samples are independent.

2.1.3 GAUSSIAN MIXTURE MODEL

GMM are commonly used in feature modeling for speech applications. They model the probability density function of observed variables using a multivariate Gaussian mixture density. Given a series of inputs, it refines the weights of each distribution through expectation – maximization algorithms.

To describe the GMM classifier more accurately, we should not that it belongs to the "unsupervised" classifiers category. This means that the training samples of a classifier are not labelled to show their category membership. More precisely, what makes GMM unsupervised is that during the training of the classifier, we try to estimate the underlying probability density functions (pdf's) of the observations.

ADVANTAGES

- GMMs have shown to be useful if features are particularly weak or missing.
- They are computationally efficient and can be easily implemented.

DISADVANTAGES

- GMM over-fits the data.
- Rely on low-level acoustic information [8].
- Performance of these systems degrades when the microphone or acoustic environment changes between training and recognition data [8].

From the above survey, it is clear that the GMM provides the flexibility of the resulting recognition system in which one can easily change the size, type, or architecture of the models to suit particular words, sounds. Hence we chose GMM as a classifier model for our project.

2.3 JOINT FACTOR ANALYSIS

Joint Factor Analysis is a statistical approach to extract the speaker dependent, speaker independent, channel dependent component from the given supervector.

ADVANTAGE

- Joint Factor Analysis (JFA) method achieves relative improvement on equal error rate and minimum value of detection cost function decreases [5].

- Joint Factor Analysis model can perform very well in speaker verification using a computationally inexpensive decision rule that steers a middle course between the exact and simplified scoring rule [10].

DISADVANTAGES

- Joint Factor Analysis is mathematically and computationally demanding in many respects [9].

- JFA requires a well balanced training set in which a typical training speaker is recorded under a variety of channel conditions that is sufficiently broad to cover most of the channel variation that is likely to be encountered during the recognition time [9].

Hence we choose Joint Factor Analysis because of its relative improvement on equal error rate, low value of detection cost function and inexpensive decision rule compared to other methods.

CHAPTER 3

SYSTEM DESIGN

Speaker verification is the computing task of validating a user's claimed identity using characteristics extracted from their voices. Speaker verification system has two phases: training phase and testing phase. During the training phase the speaker’s voice is recorded and typically a number of features are extracted to form a voice print. In the testing phase a speech sample or utterance is compared to previously determined voice print to determine the best match.

3.1 ARCHITECTURE DIAGRAM

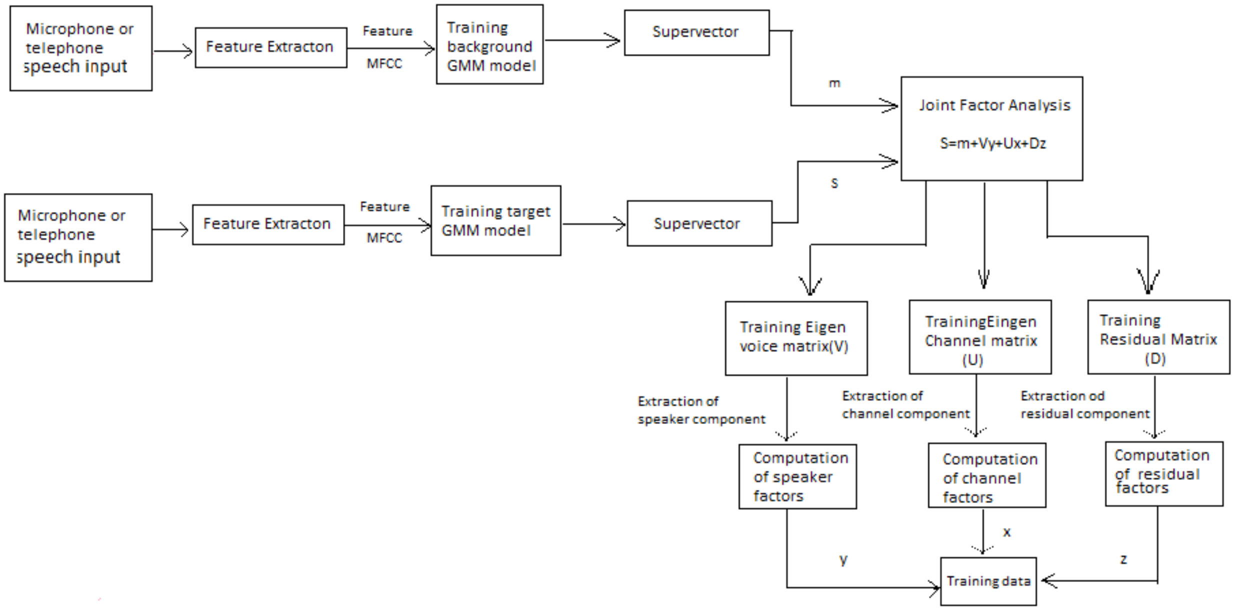

Microphone or telephone speech input
Feature Extracton
Feature
MFCC
Training background GMM model
Supervector
m
Joint Factor Analysis
S=m+Vy+Ux+Dz
s
Microphone or telephone speech input
Feature Extracton
Feature
MFCC
Training target GMM model
Supervector
Training Eigen voice matrix(V)
TrainingEingen Channel matrix (U)
Training Residual Matrix (D)
Extraction of speaker component
Extraction of channel component
Extraction od residual component
Computation of speaker factors
Computation of channel factors
Computation of residual factors
x
y
z
Training data

Figure 3.1 Architecture of Training Phase

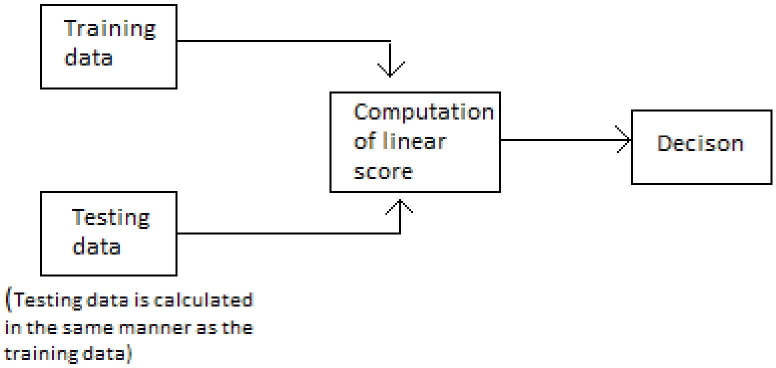

Training data
Computation of linear score
Decison
Testing data
(Testing data is calculated in the same manner as the training data)

Figure 3.2 Architecture of Testing Phase

3.2 TRAINING PHASE

Figure 3.1 shows various modules used in the training phase such as feature extractor, trained GMM and UBM model etc.

Telephone/Microphone Data

Each speaker's utterance is recorded using wavesurfer software. The recorded file is in wav format which is given as input to the Hidden Markov Model toolkit (HTK toolkit).

Input: Speaker's utterance
Output: .wav file (recorded file)

Feature Extraction

The wav file containing the speaker's telephone or microphone speech data is given as input to this module. The Mel Frequency Cepstral Coefficients (MFCC) are extracted using HTK toolkit. The mfcc features are stored in a file with extension .mfc.

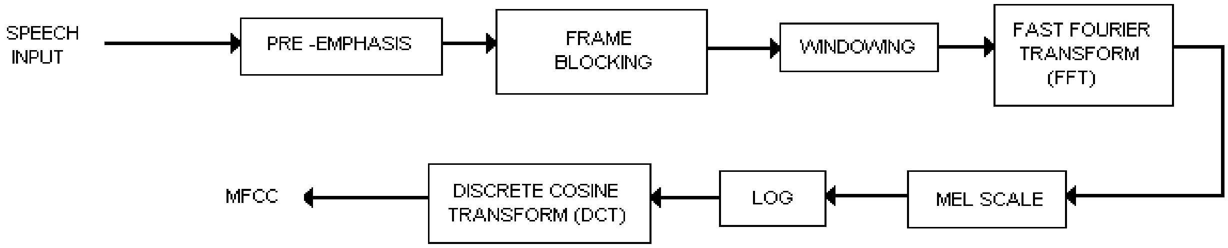


Figure 3.3 MFCC Extraction Process

The various steps involved in the extraction of mfcc features are shown in figure 3.3 and are explained as follows,

1. Pre-Emphasis: The speech signal is sent to a high-pass filter. Pre-emphasis is used to compensate the high frequency part which is suppressed during the sound production mechanism of humans.

2. Frame-Blocking: The input signal is segmented into frames of 20-30 ms. Usually the frame size (sample points) is in the powers of two to facilitate the use of FFT.

3. Windowing: Each frame has to be multiplied with a hamming window in order to keep the continuity of the first and the last points in the frame.

4. Fast Fourier Transform (FFT): Magnitude frequency response of each frame is obtained by performing FFT.

5. Mel Scale: The powers of the spectrum obtained above is mapped on to the mel scale.

6. Log: The logs of the powers at each of the mel frequencies is obtained.

7. Discrete Cosine Transform (DCT): The DCT of the mel log powers are taken considering it as a signal. The MFCCs are the amplitude of the resulting spectrum.

Input : .wav file
Output: .mfc file (file containing MFCC features)

Trained Target GMM Model

Using the mfcc features from the .mfc files, a speaker dependent GMM model is created using the HTK toolkit. The GMM model contains parameters such as mean, covariance and weight. Mean vectors represent the speaker dependent supervector (s).

Input: .mfc file (file containing MFCC features of the particular speaker)
Output: Gaussian Mixture Model of the particular speaker.

Supervector

The values of the mean vector represent the supervector(s) which is used in the JFA model.

Input: Gaussian Mixture Model of the particular speaker.
Output: Supervector of speaker dependent GMM.

Universal Background Model

Using the same procedure Universal Background Model is created which contains the speaker independent charecteristics of all speakers. The speaker independent supervector m is obtained from the universal background model.

Input: .mfc file (file containing MFCC features of all speakers)
Output: Speaker independent superverctor (m)

Joint Factor Analysis

The speaker dependent supervector is decomposed into speaker independent, speaker dependent, channel dependent, and residual components. Each component can be represented by a low-dimensional set of factors, which operate along the principal dimensions (i.e. eigen-dimensions) of the corresponding component.

The speaker GMM supervector s is decomposed as follows:

$$s = m + Vy + Ux + Dz$$

where m is the speaker-independent supervector (from UBM)

V is the eigenvoice matrix

y represents the speaker factors

U is the eigenchannel matrix

x represents the channel factors

D is the residual matrix

z represents the speaker residual factors

Training Eigenvoice Matrix V

1. The Eigenchannel (U) and residual (D) matrix are assumed to be zero.

2. The 0th, 1st, and 2nd order sufficient statistics for each speaker (s) and Gaussian mixture component (c) are accumulated

0th order statistic $\longrightarrow N_c(s) = \sum_{t \in s} \gamma_t(c)$

1st order statistic $\longrightarrow F_c(s) = \sum_{t \in s} \gamma_t(c) Y_t$

2nd order statistic $\longrightarrow S_c(s) = diag\left(\sum_{t \in s} \gamma_t(c) Y_t Y_t^* \right)$

where Y_t^* denotes hermitian transpose of vector or matrix:

diag - represents only the diagonal entries.

3. The 1st and 2nd order statistics are centered.

$$\tilde{F}_c(s) = F_c(s) - N_c(s) m_c$$

$$\tilde{S}_c(s) = S_c(s) - diag\left(F_c(s)m_c^* + m_cF_c(s)^* - N_c(s)m_cm_c^*\right)$$

where $N_c(s)$ represents the 0th order statistic

m_c represents UBM mean for mixture component c

$\tilde{F}_c(s)$ represents centered first order statistic

$\tilde{S}_c(s)$ represents centered second order statistic

4. The statistics are expanded into matrices.

$$NN(s) = \begin{bmatrix} N_1(s) * I & & \\ & \ddots & \\ & & N_C(s) * I \end{bmatrix} \quad FF(s) = \begin{bmatrix} \tilde{F}_1(s) \\ \vdots \\ \tilde{F}_C(s) \end{bmatrix}$$

$$SS(s) = \begin{bmatrix} \tilde{S}_1(s) & & \\ & \ddots & \\ & & \tilde{S}_C(s) \end{bmatrix}$$

where I - represents identity matrix.

C - represents total Gaussian mixtures.

5. Initial estimate of the speaker factors y is calculated as follows,

$$l_V(s) = I + V^* * \Sigma^{-1} * NN(s) * V$$

$$y(s) \sim Normal(l_V^{-1}(s) * V^* * \Sigma^{-1} * FF(s), l_V^{-1}(s)) \Rightarrow$$

$$\overline{y}(s) = E[y(s)] = l_V^{-1}(s) * V^* * \Sigma^{-1} * FF(s)$$

where Σ^{-1} represents inverse UBM covariance matrix

Normal - represents gaussian normal distribution.

$\bar{y}$(s) represents the expected value

6. Some additional statistics across the speakers are accumulated.

$$N_c = \sum_s N_c(s)$$

$$\mathrm{A}_c = \sum_s N_c(s) l_V^{-1}(s)$$

$$\mathbb{C} = \sum_s FF(s) * (l_V^{-1}(s) * V^* * \Sigma^{-1} * FF(s))^*$$

$$NN = \sum_s NN(s)$$

where l_v^{-1} represents covariance of posterior distribution of y(s)

Σ^{-1} represents transposed mean of posterior distribution of y(s)

7. V estimates are computed.

$$V = \begin{bmatrix} V_1 \\ \vdots \\ V_C \end{bmatrix} = \begin{bmatrix} A_1^{-1} * \mathbb{C}_1 \\ \vdots \\ A_C^{-1} * \mathbb{C}_C \end{bmatrix} \quad \mathbb{C} = \begin{bmatrix} \mathbb{C}_1 \\ \vdots \\ \mathbb{C}_C \end{bmatrix}$$

where V_c represents the block matrix components of V corresponding to each Gaussian mixture.

C_c represents the block matrix components of C corresponding to each Gaussian mixture.

8. Steps from 5-7 are iterated as many times (say 20) and the final V estimate is calculated.

Training Eigenchannel Matrix U

1. The estimate of V from the above step is used to train the matrix U, assume D is zero.

2. The estimate of speaker factor y for each speaker, and 0th and 1st order statistics for each conversation side (conv) of each speaker (s) in JFA training data are computed as follows,

$$N_c(conv,s) = \sum_{t \in conv,s} \gamma_t(c)$$

$$F_c(conv,s) = \sum_{t \in conv,s} \gamma_t(c) Y_t$$

3. For each speaker (s), the speaker shift (along with speaker- independent shift) is computed using matrix V and speaker factors y .

$$spkrshift(s) = m + V * y(s)$$

4. Gaussian posterior-weighted speaker shift is subtracted from first order statistics for each conversation side of each speaker (used for JFA training).

$$\tilde{F}_c(conv,s) = F_c(conv,s) - spkrshift(s) * N_c(conv,s)$$

5. The statistics are expanded into matrices

$$NN(conv,s) = \begin{bmatrix} N_1(conv,s) * I & & \\ & \ddots & \\ & & N_C(conv,s) * I \end{bmatrix}$$

$$FF(conv,s) = \begin{bmatrix} \tilde{F}_1(conv,s) \\ \vdots \\ \tilde{F}_C(conv,s) \end{bmatrix}$$

6. NN(conv,s) and FF(conv,s) used to train U and x in exact same way that NN(s) and FF(s) was used to train V and y.

7. Training procedure for V and y are iterated as many times (say 20) using NN(conv,s) and FF(conv,s).

Training Residual Matrix D

1. The estimate of V and U are used to train the residual matrix D.

2. For each speaker (s), the speaker shift using matrix V and speaker factors
y is computed.

$$spkrshift(s) = m + V * y(s)$$

3. For each conversation side (conv) of speaker (s), the channel shift using matrix U and channel factors z is computed.

$$chanshift(conv, s) = U * x(conv, s)$$

4. Gaussian posterior- weighted speaker shift and channel shifts from first order statistics are subtracted for each speaker (used for JFA training).

$$\tilde{F}_c(s) = F_c(s) - spkrshift(s) * N_c(s) - \sum_{conv \in s} chanshift(conv, s) * N_c(conv, s)$$

where chanshift - represents the channel shift computed for V estimate.

5. The statistics are expanded into matrices.

$$NN(s) = \begin{bmatrix} N_1(s) * I & & \\ & \ddots & \\ & & N_C(s) * I \end{bmatrix}$$

$$FF(s) = \begin{bmatrix} \tilde{F}_1(s) \\ \vdots \\ \tilde{F}_C(s) \end{bmatrix}$$

6. Initial estimate of the residual factors z is calculated as follows

$$l_D(s) = I + D^2 * \Sigma^{-1} * NN(s)$$

$$z(s) \sim Normal(l_D^{-1}(s) * D * \Sigma^{-1} * FF(s), l_D^{-1}(s)) \Rightarrow$$

$$\overline{z}(s) = E[z(s)] = l_D^{-1}(s) * D * \Sigma^{-1} * FF(s)$$

where $\bar{z}$(s) is the expected value

7. Some additional statistics across the speakers are accumulated.

$$N_c = \sum_s N_c(s)$$

$$a = \sum_s diag(NN(s) * l_D^{-1}(s))$$

$$b = \sum_s diag(FF(s) * (l_D^{-1}(s) * D * \Sigma^{-1} * FF(s))^{*})$$

$$NN = \sum_s NN(s)$$

where l_D^{-1} represents covariance of posterior distribution of y(s)

Σ^{-1} represents transposed mean of posterior distribution of y(s)

8. D estimates are calculated as follows

$$D = \begin{bmatrix} D_1 \\ \vdots \\ D_C \end{bmatrix} = \begin{bmatrix} a_1^{-1} * b_1 \\ \vdots \\ a_C^{-1} * b_C \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_C \end{bmatrix}$$

where D_c represents block matrix components of D corresponding to each Gaussian mixture

b_c represents block matrix components of b corresponding to each Gaussian mixture

9. Steps from 6-8 are iterated as many times (say 20) and the final D estimate is calculated.

3.3 TESTING PHASE

Figure 3.2 shows the various modules used in the testing phase such as linear score calculator, decision maker etc.

Training data

Training data is collected from the training phase. It contains speaker factors (y), channel factors (x) and residual factors (z).

Testing data

Testing data is accumulated in the same way as the training data. It is collected during the time of testing. It contains the characteristics if the test subject.

Computing linear score

In this stage the V, D, U matrices are used to get the estimates of y,x,z respectively in terms of their posterior means given the observation. The final score is obtained using conversation side (tst) and target speaker conversation side(tar) via following linear product

$$Score = (V * y(tar) + D * z(tar))^{*} * \Sigma^{-1} * (FF(tst) - NN(tst) * m - NN(tst) * U * x(tst))$$

1) (V*y(tar)+D*z(tar)) - is centered around speaker and residual factors.

2)(FF(tst)-NN(tst)*m-NN(tst)*U*x(tst)) - has the speaker independent and channel factors removed and hence also centered around speaker and residual factors.

CHAPTER 4
IMPLEMENTATION

4.1 DATA SET

The data set contains 130 utterances of each speaker collected through microphone and telephone. Each utterance lasts for 3 seconds. The utterances are stored as wave files.

4.2 HCOPY

HCopy is a general-purpose tool for copying and manipulating speech files. This program will copy one or more data files to a designated output file, optionally converting the data into a parameterized form. It is used to extract features from the wave files. The features are stored as mfc files. The configuration details of the features are contained in the configuration file of the HCopy tool.

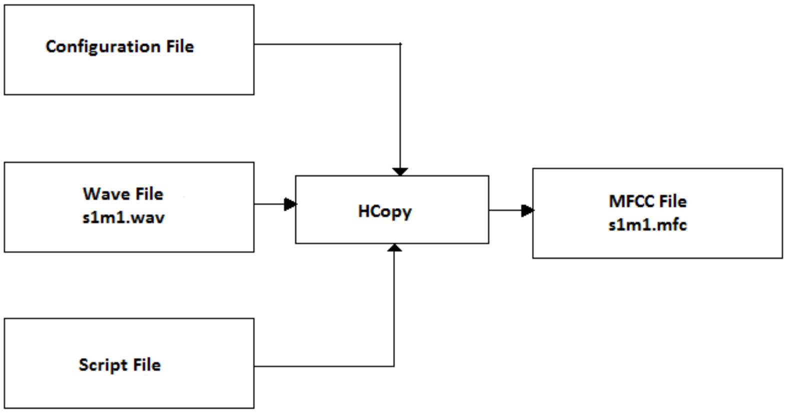
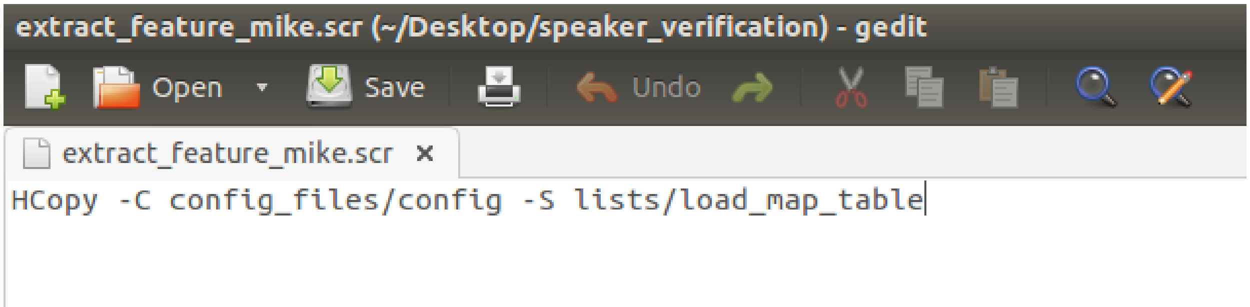
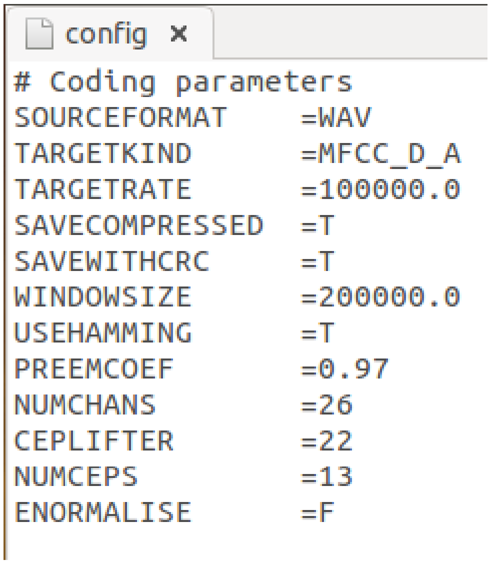


Figure 4.1 HCopy block diagram


extract_feature_mike.scr (~/Desktop/speaker_verification) - gedit
Open
Save
Undo
extract_feature_mike.scr

```
HCopy -C config_files/config -S lists/load_map_table
```

Figure 4.2 HCopy tool



```
# Coding parameters
SOURCEFORMAT     =WAV
TARGETKIND       =MFCC_D_A
TARGETRATE       =100000.0
SAVECOMPRESSED   =T
SAVEWITHCRC      =T
WINDOWSIZE       =200000.0
USEHAMMING       =T
PREEMCOEF        =0.97
NUMCHANS         =26
CEPLIFTER        =22
NUMCEPS          =13
ENORMALISE       =F
```

Figure 4.3 Configuration file of HCopy tool

The configuration file of the HCopy tool with the coding parameters is listed in figure 4.3. It describes the following parameters:

SOURCEFORMAT - Source file format (.wav)

TARGETKIND - Target file format (.mfc)

TARGETRATE - The rate of frames (HTK uses units of 100ns)

SAVECOMPRESSED - Should output be saved in compressed format(True)

SAVEWITHCRC - Should a CRC checksum be added (True)

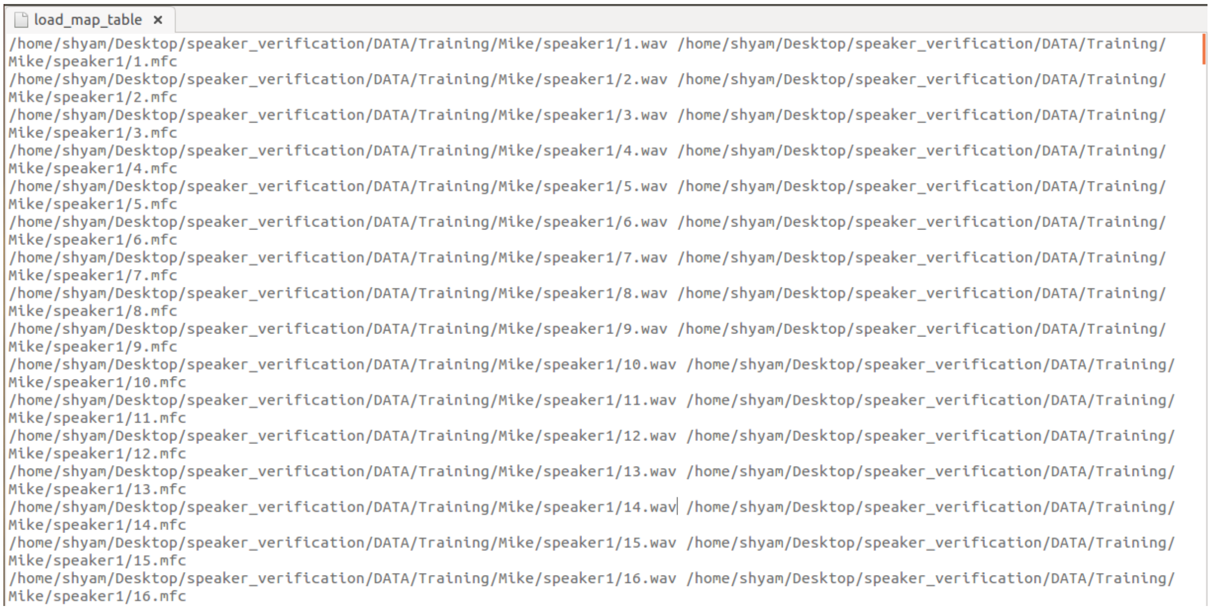
USEHAMMING - Should HammingWindow Transformation be made (True)

WINDOWSIZE - Specify the Hamming window size

NUMCHANS - Number of channels in filters

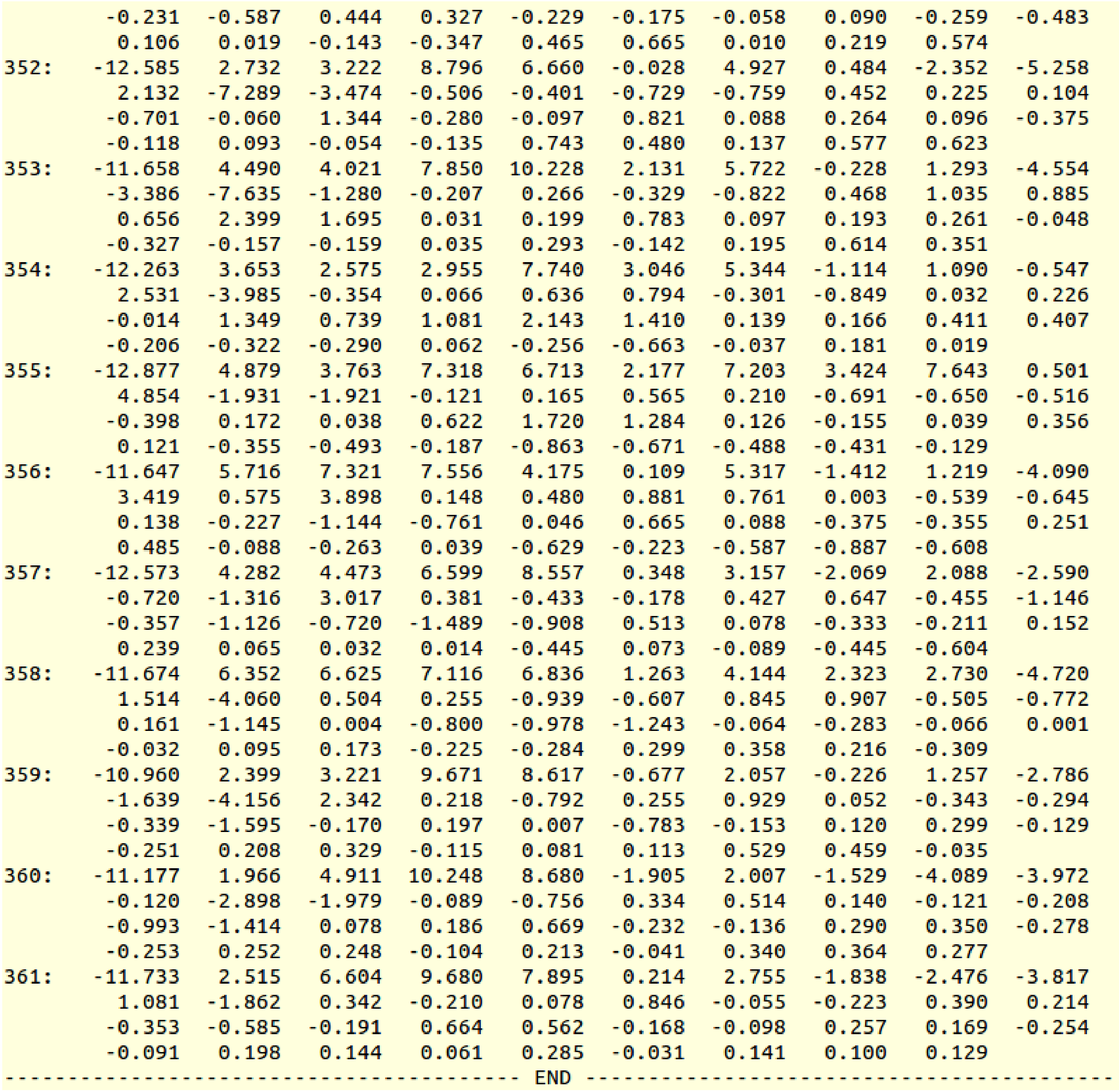
NUMCEPS - Number of MFC coefficients

The source files (.wav) and the target files (.mfc) are listed in the load map table file. A part of map table is shown in figure 4.4



```
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/1.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/1.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/2.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/2.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/3.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/3.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/4.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/4.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/5.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/5.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/6.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/6.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/7.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/7.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/8.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/8.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/9.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/9.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/10.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/10.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/11.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/11.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/12.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/12.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/13.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/13.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/14.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/14.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/15.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/15.mfc
/home/shyam/Desktop/speaker_verification/DATA/Training/Mike/speaker1/16.wav /home/shyam/Desktop/speaker_verification/DATA/Training/
Mike/speaker1/16.mfc
```

Figure 4.4 Map table containing source and destination



```
        -0.231  -0.587   0.444   0.327  -0.229  -0.175  -0.058   0.090  -0.259  -0.483
         0.106   0.019  -0.143  -0.347   0.465   0.665   0.010   0.219   0.574
352:   -12.585   2.732   3.222   8.796   6.660  -0.028   4.927   0.484  -2.352  -5.258
         2.132  -7.289  -3.474  -0.506  -0.401  -0.729  -0.759   0.452   0.225   0.104
        -0.701  -0.060   1.344  -0.280  -0.097   0.821   0.088   0.264   0.096  -0.375
        -0.118   0.093  -0.054  -0.135   0.743   0.480   0.137   0.577   0.623
353:   -11.658   4.490   4.021   7.850  10.228   2.131   5.722  -0.228   1.293  -4.554
        -3.386  -7.635  -1.280  -0.207   0.266  -0.329  -0.822   0.468   1.035   0.885
         0.656   2.399   1.695   0.031   0.199   0.783   0.097   0.193   0.261  -0.048
        -0.327  -0.157  -0.159   0.035   0.293  -0.142   0.195   0.614   0.351
354:   -12.263   3.653   2.575   2.955   7.740   3.046   5.344  -1.114   1.090  -0.547
         2.531  -3.985  -0.354   0.066   0.636   0.794  -0.301  -0.849   0.032   0.226
        -0.014   1.349   0.739   1.081   2.143   1.410   0.139   0.166   0.411   0.407
        -0.206  -0.322  -0.290   0.062  -0.256  -0.663  -0.037   0.181   0.019
355:   -12.877   4.879   3.763   7.318   6.713   2.177   7.203   3.424   7.643   0.501
         4.854  -1.931  -1.921  -0.121   0.165   0.565   0.210  -0.691  -0.650  -0.516
        -0.398   0.172   0.038   0.622   1.720   1.284   0.126  -0.155   0.039   0.356
         0.121  -0.355  -0.493  -0.187  -0.863  -0.671  -0.488  -0.431  -0.129
356:   -11.647   5.716   7.321   7.556   4.175   0.109   5.317  -1.412   1.219  -4.090
         3.419   0.575   3.898   0.148   0.480   0.881   0.761   0.003  -0.539  -0.645
         0.138  -0.227  -1.144  -0.761   0.046   0.665   0.088  -0.375  -0.355   0.251
         0.485  -0.088  -0.263   0.039  -0.629  -0.223  -0.587  -0.887  -0.608
357:   -12.573   4.282   4.473   6.599   8.557   0.348   3.157  -2.069   2.088  -2.590
        -0.720  -1.316   3.017   0.381  -0.433  -0.178   0.427   0.647  -0.455  -1.146
        -0.357  -1.126  -0.720  -1.489  -0.908   0.513   0.078  -0.333  -0.211   0.152
         0.239   0.065   0.032   0.014  -0.445   0.073  -0.089  -0.445  -0.604
358:   -11.674   6.352   6.625   7.116   6.836   1.263   4.144   2.323   2.730  -4.720
         1.514  -4.060   0.504   0.255  -0.939  -0.607   0.845   0.907  -0.505  -0.772
         0.161  -1.145   0.004  -0.800  -0.978  -1.243  -0.064  -0.283  -0.066   0.001
        -0.032   0.095   0.173  -0.225  -0.284   0.299   0.358   0.216  -0.309
359:   -10.960   2.399   3.221   9.671   8.617  -0.677   2.057  -0.226   1.257  -2.786
        -1.639  -4.156   2.342   0.218  -0.792   0.255   0.929   0.052  -0.343  -0.294
        -0.339  -1.595  -0.170   0.197   0.007  -0.783  -0.153   0.120   0.299  -0.129
        -0.251   0.208   0.329  -0.115   0.081   0.113   0.529   0.459  -0.035
360:   -11.177   1.966   4.911  10.248   8.680  -1.905   2.007  -1.529  -4.089  -3.972
        -0.120  -2.898  -1.979  -0.089  -0.756   0.334   0.514   0.140  -0.121  -0.208
        -0.993  -1.414   0.078   0.186   0.669  -0.232  -0.136   0.290   0.350  -0.278
        -0.253   0.252   0.248  -0.104   0.213  -0.041   0.340   0.364   0.277
361:   -11.733   2.515   6.604   9.680   7.895   0.214   2.755  -1.838  -2.476  -3.817
         1.081  -1.862   0.342  -0.210   0.078   0.846  -0.055  -0.223   0.390   0.214
        -0.353  -0.585  -0.191   0.664   0.562  -0.168  -0.098   0.257   0.169  -0.254
        -0.091   0.198   0.144   0.061   0.285  -0.031   0.141   0.100   0.129
---------------------------------------- END ----------------------------------------
```

Figure 4.5 Features extracted from microphone data

MFC file contains many frames with corresponding frame numbers. Each frame consists of 39 values. First 13 values represent cepstral coefficient values. The second 13 values represent first derivative or delta. Last 13 values represent second derivative or accelerator coefficients. The HInit and HRest tools of the HTK toolkit are used to generate the GMM model.

4.3 HINIT

Having set up a prototype, a HMM can be initialised using HTKtool HINIT. The basic principle of HINIT depends on the concepts of a HMM as a generator of speech vectors. Every training example can be viewed as output of the HMM whose parameter are to be estimated. Thus if the state that generated each vector in the training data was known, then the unknown means and variance could be estimated by averaging all the vectors associated with each state.

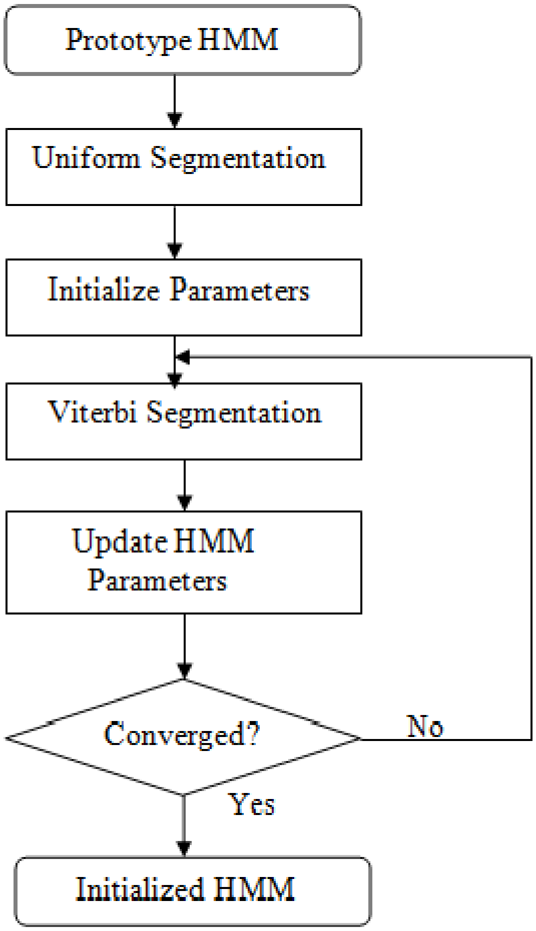


Figure 4.6 HInit Operation

The viterbi algorithm is used to find the most likely state sequence corresponding to each training example, then the HMM parameters are estimated.

As a side-effect of finding the Viterbi state alignment, the log likelihood of the training data can be computed. Hence, the whole estimation process can be repeated until no further increase in likelihood is obtained.

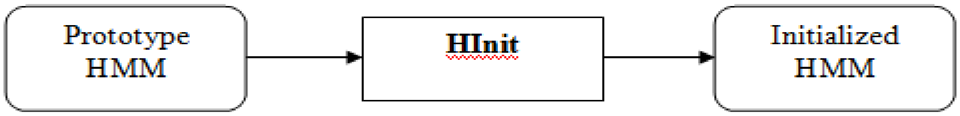


Figure 4.7 Initialization of HMM

4.4 HREST

HREST is the final tool to manipulate isolated unit HMMs. Its operation is very similar to the HINIT except that the HMM definition to have been initialized and it uses Baum-Welch re-estimation in place of Viterbi training. This involves finding the probability of being at each state at each time frame using the forward-backward algorithm. The generate speaker models mike.scr script is executed to generate the speaker models.

Re-estimation of models is explained in Figure 4.9.Using an initial parameter instantiation, the Baum-Welch algorithm iteratively re-estimates the parameters and improves the probability that given observation are generated by the new parameters.

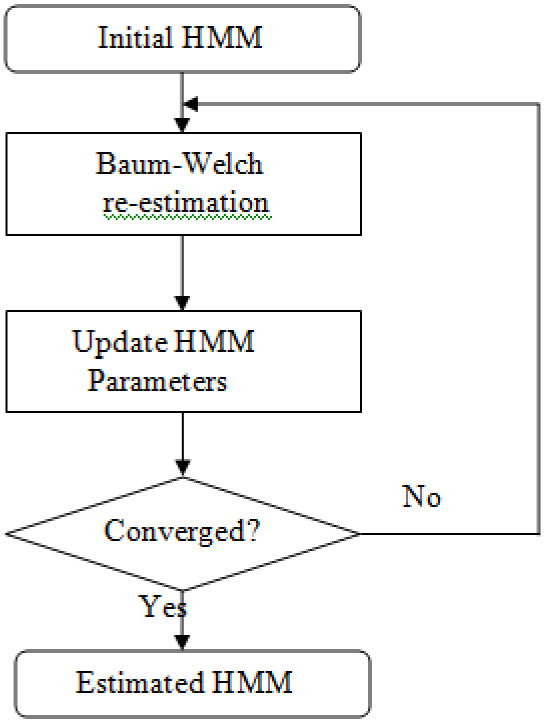


Figure 4.8 HRest Operation

Initialised models are re-estimated using “HREST”. Its operation is very similar to HInit except that it expects the input HMM definition to have been initialised and it uses Baum-Welch re-estimation. Baum-Welch algorithm is used for finding the probability of being in each state, transition probability, initial probability to enter a state at each time frame.

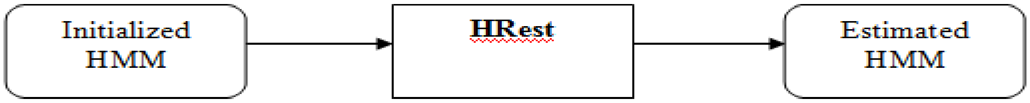
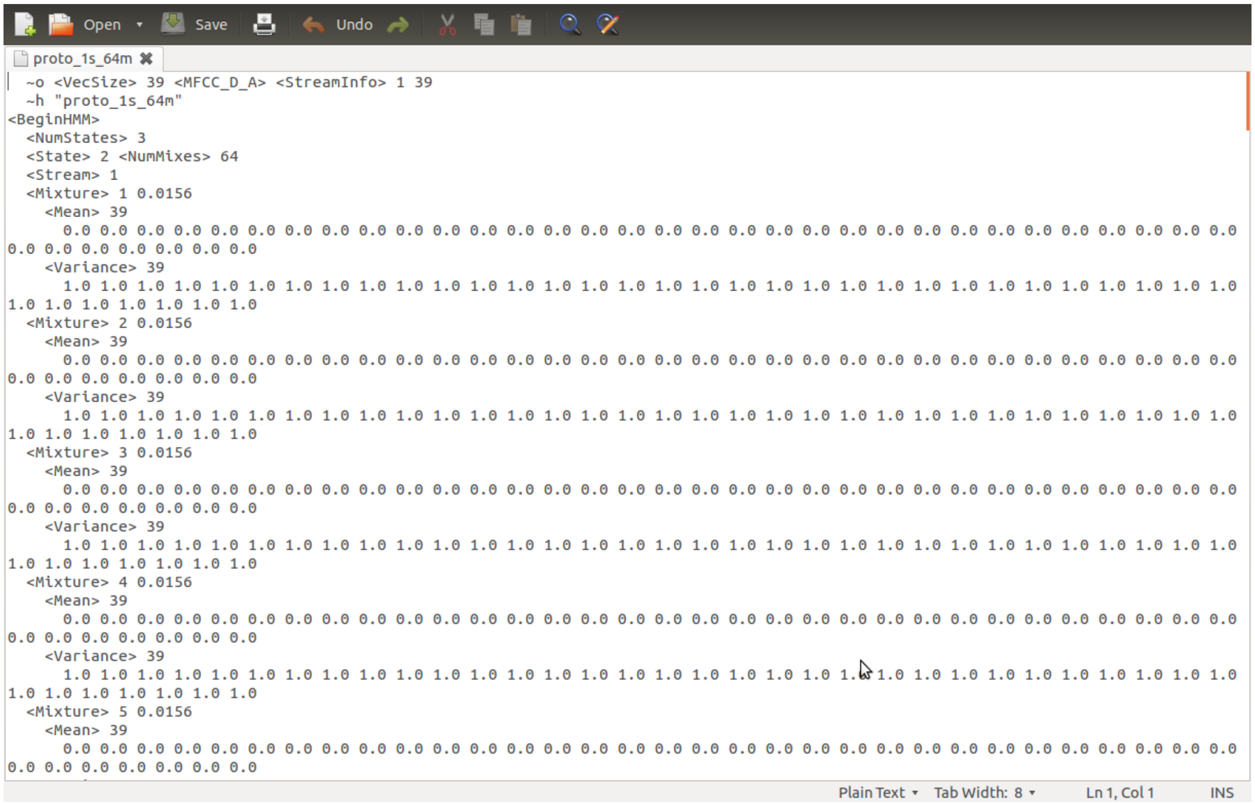


Figure 4.9 Re-estimation of HMM model

4.5 INITIAL PROTOTYPE GENERATION

The structure of the initial prototype generation is shown in figure 4.10. The mean values are initialized to 0, variance values are initialized to 1, the weight components are initialized to 0.0156. The sum of the weight components is equal to 1.

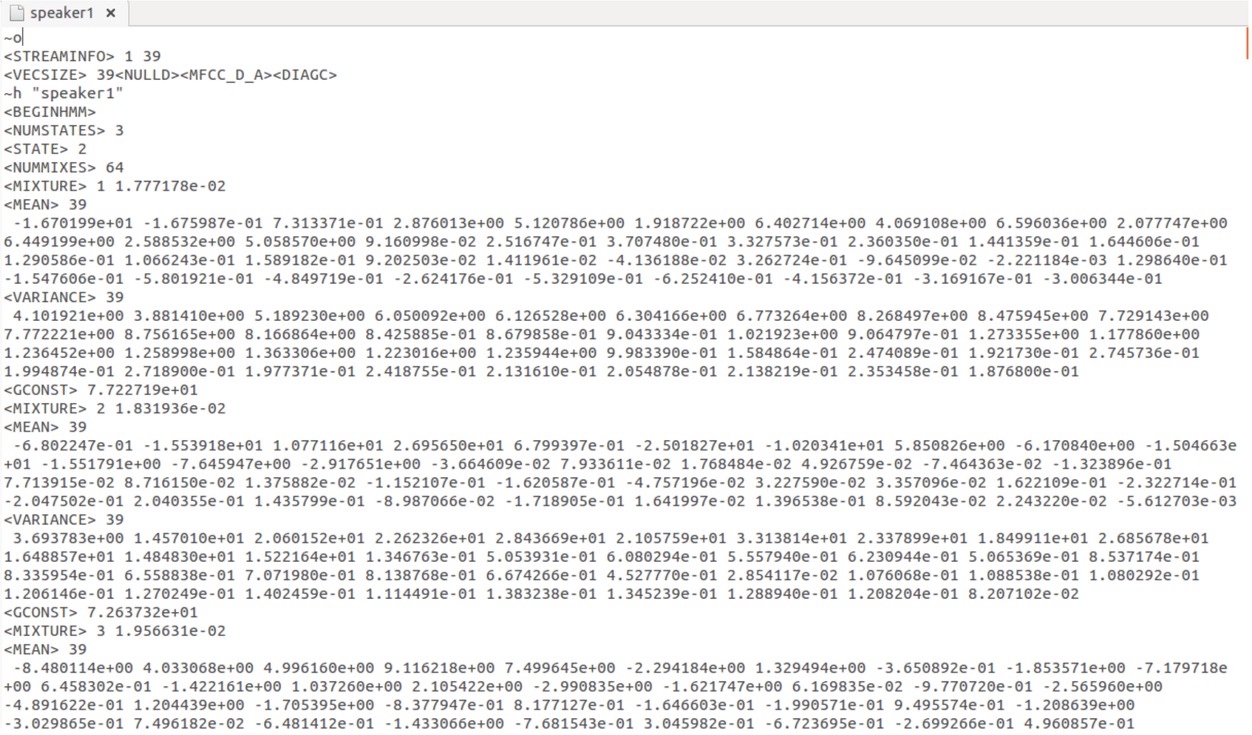


```
  ~o <VecSize> 39 <MFCC_D_A> <StreamInfo> 1 39
  ~h "proto_1s_64m"
<BeginHMM>
  <NumStates> 3
  <State> 2 <NumMixes> 64
  <Stream> 1
  <Mixture> 1 0.0156
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0
  <Mixture> 2 0.0156
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0
  <Mixture> 3 0.0156
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0
  <Mixture> 4 0.0156
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
    <Variance> 39
      1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0
  <Mixture> 5 0.0156
    <Mean> 39
      0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

Figure 4.10 Initial prototype generation

4.6 GMM Model

The speaker models for utterances recorded from telephone and microphone are created using 64 mixture component GMM. The features of all 15 speakers from both microphone and telephone channels are used to create the Universal Background Model (UBM).



```
~o
<STREAMINFO> 1 39
<VECSIZE> 39<NULLD><MFCC_D_A><DIAGC>
~h "speaker1"
<BEGINHMM>
<NUMSTATES> 3
<STATE> 2
<NUMMIXES> 64
<MIXTURE> 1 1.777178e-02
<MEAN> 39
 -1.670199e+01 -1.675987e-01 7.313371e-01 2.876013e+00 5.120786e+00 1.918722e+00 6.402714e+00 4.069108e+00 6.596036e+00 2.077747e+00
6.449199e+00 2.588532e+00 5.058570e+00 9.160998e-02 2.516747e-01 3.707480e-01 3.327573e-01 2.360350e-01 1.441359e-01 1.644606e-01
1.290586e-01 1.066243e-01 1.589182e-01 9.202503e-02 1.411961e-02 -4.136188e-02 3.262724e-01 -9.645099e-02 -2.221184e-03 1.298640e-01
-1.547606e-01 -5.801921e-01 -4.849719e-01 -2.624176e-01 -5.329109e-01 -6.252410e-01 -4.156372e-01 -3.169167e-01 -3.006344e-01
<VARIANCE> 39
 4.101921e+00 3.881410e+00 5.189230e+00 6.050092e+00 6.126528e+00 6.304166e+00 6.773264e+00 8.268497e+00 8.475945e+00 7.729143e+00
7.772221e+00 8.756165e+00 8.166864e+00 8.425885e-01 8.679858e-01 9.043334e-01 1.021923e+00 9.064797e-01 1.273355e+00 1.177860e+00
1.236452e+00 1.258998e+00 1.363306e+00 1.223016e+00 1.235944e+00 9.983390e-01 1.584864e-01 2.474089e-01 1.921730e-01 2.745736e-01
1.994874e-01 2.718900e-01 1.977371e-01 2.418755e-01 2.131610e-01 2.054878e-01 2.138219e-01 2.353458e-01 1.876800e-01
<GCONST> 7.722719e+01
<MIXTURE> 2 1.831936e-02
<MEAN> 39
 -6.802247e-01 -1.553918e+01 1.077116e+01 2.695650e+01 6.799397e-01 -2.501827e+01 -1.020341e+01 5.850826e+00 -6.170840e+00 -1.504663e
+01 -1.551791e+00 -7.645947e+00 -2.917651e+00 -3.664609e-02 7.933611e-02 1.768484e-02 4.926759e-02 -7.464363e-02 -1.323896e-01
7.713915e-02 8.716150e-02 1.375882e-02 -1.152107e-01 -1.620587e-01 -4.757196e-02 3.227590e-02 3.357096e-02 1.622109e-01 -2.322714e-01
-2.047502e-01 2.040355e-01 1.435799e-01 -8.987066e-02 -1.718905e-01 1.641997e-02 1.396538e-01 8.592043e-02 2.243220e-02 -5.612703e-03
<VARIANCE> 39
 3.693783e+00 1.457010e+01 2.060152e+01 2.262326e+01 2.843669e+01 2.105759e+01 3.313814e+01 2.337899e+01 1.849911e+01 2.685678e+01
1.648857e+01 1.484830e+01 1.522164e+01 1.346763e-01 5.053931e-01 6.080294e-01 5.557940e-01 6.230944e-01 5.065369e-01 8.537174e-01
8.335954e-01 6.558838e-01 7.071980e-01 8.138768e-01 6.674266e-01 4.527770e-01 2.854117e-02 1.076068e-01 1.088538e-01 1.080292e-01
1.206146e-01 1.270249e-01 1.402459e-01 1.114491e-01 1.383238e-01 1.345239e-01 1.288940e-01 1.208204e-01 8.207102e-02
<GCONST> 7.263732e+01
<MIXTURE> 3 1.956631e-02
<MEAN> 39
 -8.480114e+00 4.033068e+00 4.996160e+00 9.116218e+00 7.499645e+00 -2.294184e+00 1.329494e+00 -3.650892e-01 -1.853571e+00 -7.179718e
+00 6.458302e-01 -1.422161e+00 1.037260e+00 2.105422e+00 -2.990835e+00 -1.621747e+00 6.169835e-02 -9.770720e-01 -2.565960e+00
-4.891622e-01 1.204439e+00 -1.705395e+00 -8.377947e-01 8.177127e-01 -1.646603e-01 -1.990571e-01 9.495574e-01 -1.208639e+00
-3.029865e-01 7.496182e-02 -6.481412e-01 -1.433066e+00 -7.681543e-01 3.045982e-01 -6.723695e-01 -2.699266e-01 4.960857e-01
```

Figure 4.11 Speaker model for utterances recorded from microphone

4.7 JOINT FACTOR ANALYSIS

Joint Factor Analysis is a statistical approach to extract the speaker dependent, speaker independent, channel and residual components from a given speaker supervector. The Joint Factor Analysis is carried out using MATLAB as follows:

The 130 feature files of each speaker is split and given as follows. The first 20 files are given as enroll data, the next 20 files are given as test data. Files 41-70 are given as eigenchannel data and files 41-130 are given as eigenvoice data.

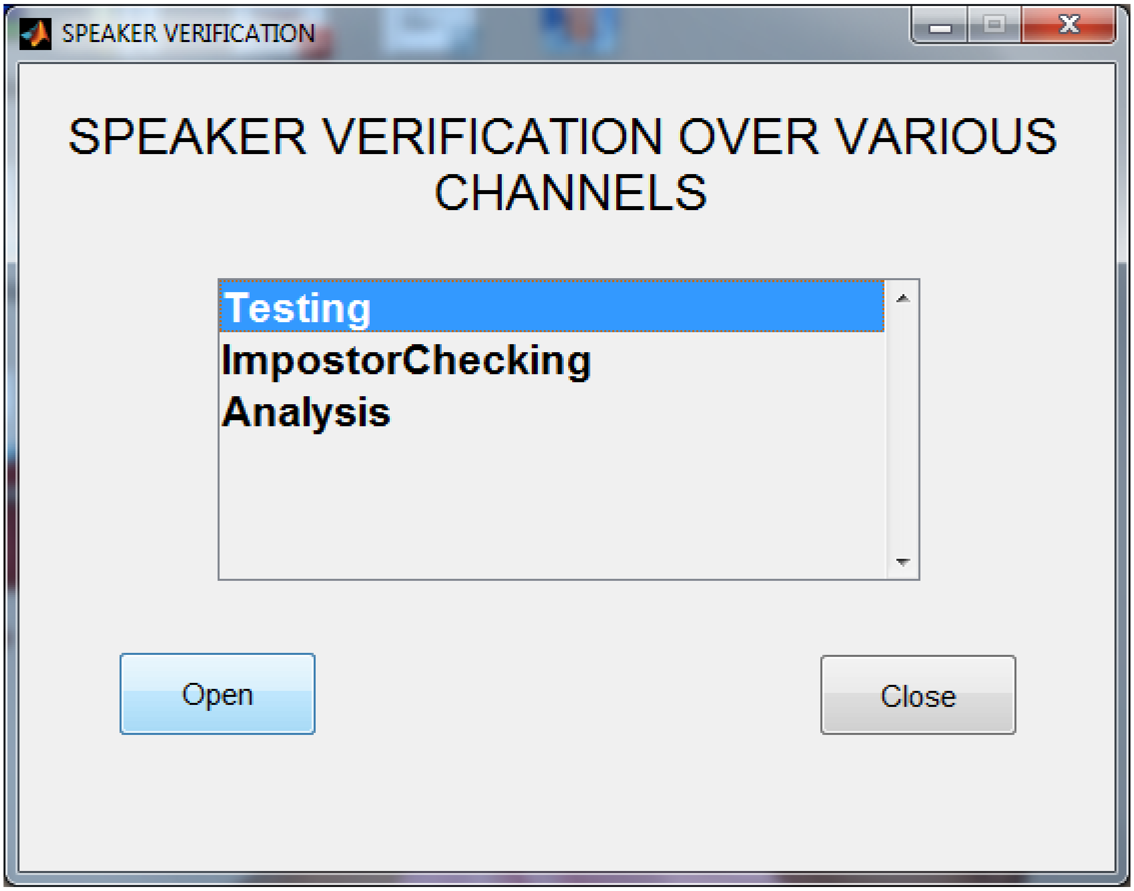
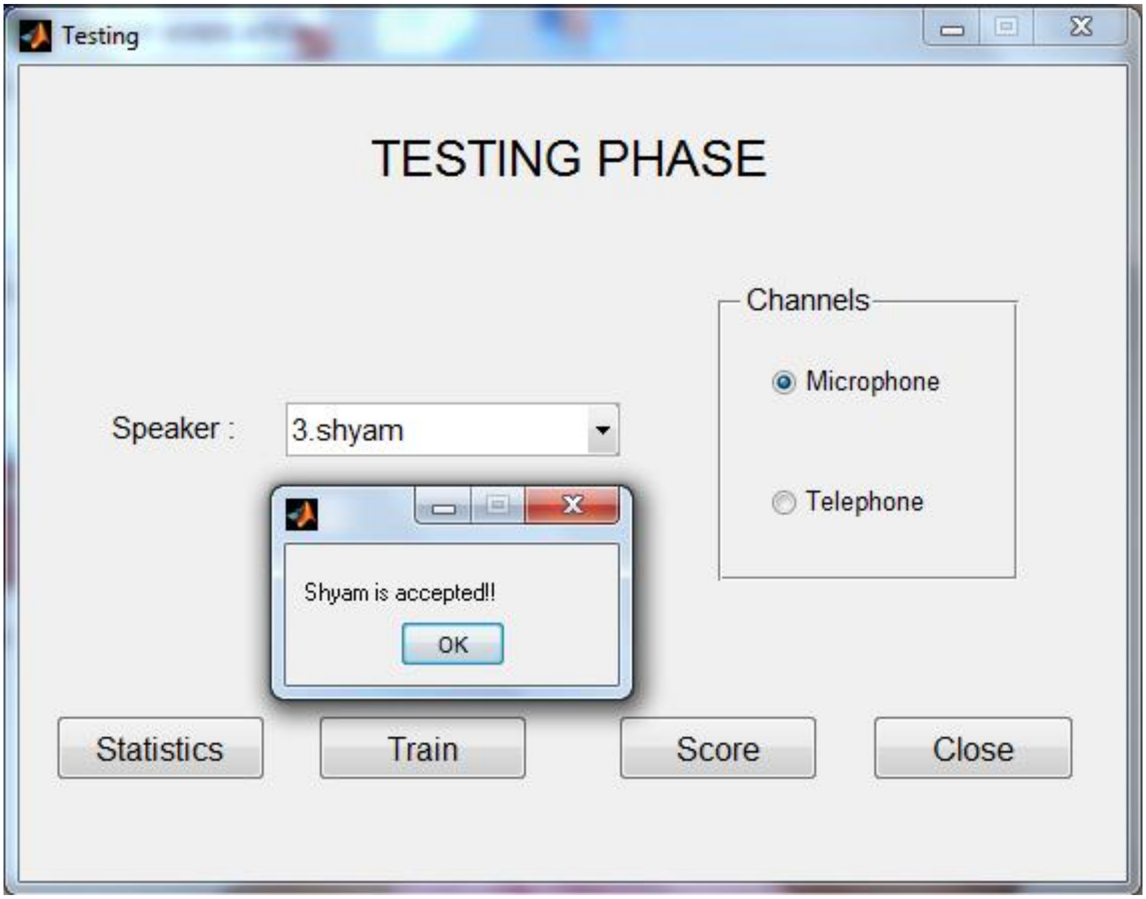


Figure 4.12 GUI Window of Speaker Verification System

The graphical user interface for speaker verification system is shown in figure 4.12. It consists of three phases: testing, impostor checking and performance analysis. The testing phase is used to verify a speaker’s identity. The impostor checking phase is used to identify impostor and the performance analysis phase is used to evaluate the performance of the system.



Speaker : 3.shyam

Channels

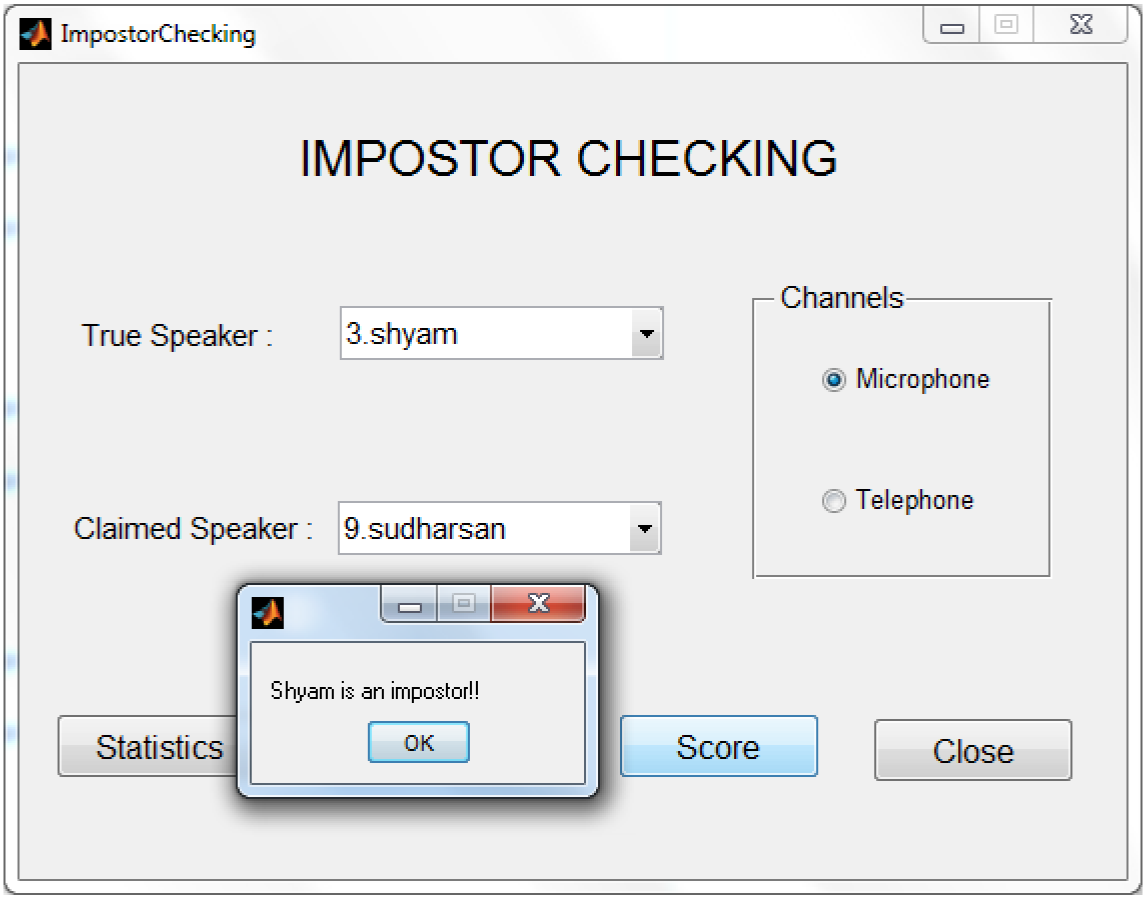
● Microphone

○ Telephone

Shyam is accepted!!

Figure 4.13 GUI Window of Testing Phase

The testing phase is used to verify the speaker’s claimed identity. It consists of a drop down list box to choose the speaker, a radio button to choose the channel and buttons to compute statistics, train matrices, compute the score and close the window. If the chosen speaker’s score is greatest, a message box showing the speaker is accepted is displayed. Else a message box showing the speaker is rejected is displayed.



True Speaker : 3.shyam

Claimed Speaker : 9.sudharsan

Channels

● Microphone

○ Telephone

Shyam is an impostor!!

Figure 4.14 GUI Window of Impostor Checking Phase

The impostor checking phase consists of two drop down list boxes to select the true speaker and the claimed speaker, a radio button to choose the channel, and buttons to compute statistics, train matrices, compute the score and close the window. If the true speaker's score is greater than the claimed speaker, a message box showing true speaker is an impostor is displayed. Else, a message box showing the true speaker is accepted is displayed.

When the statistics button is clicked the first order and zeroth order centered statistics are computed. The zeroth order statistic N consists of a 30x64 matrix. The number of rows (30) represent the number of enrolled speakers and the number of columns (64) represent the number of mixture components. The first order statistic

F consists of a 30x2496 matrix. The number of rows (30) represent the number of enrolled speakers and the number of columns (2496) is a product of the number of mixture components (64) and the number of dimensions (39).

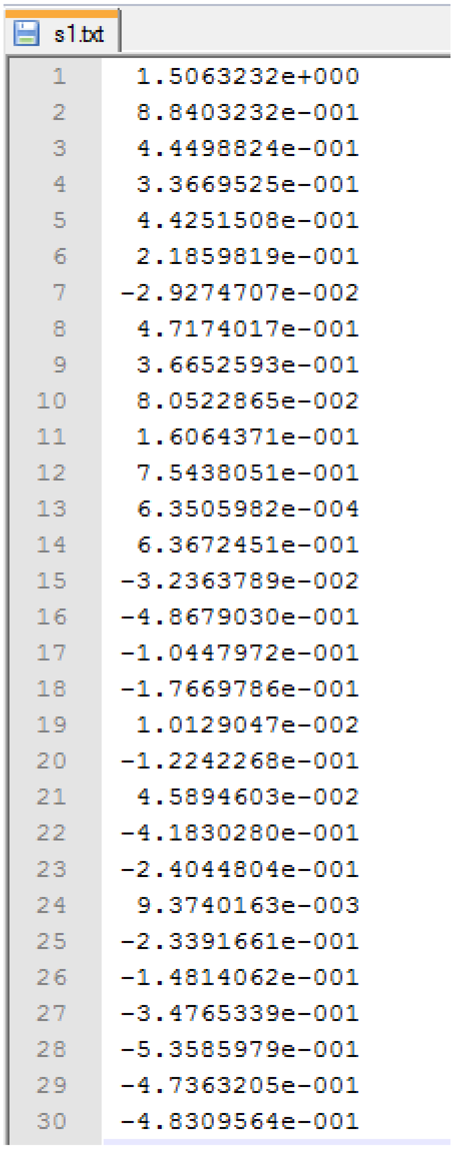
When the train button is clicked, the V, U and D matrices are trained.

The V matrix consists of a 4x2496. The number of rows (4) represent the initial number of eigenvoice components and the number of columns (2496) is a product of the number of mixture components (64) and the number of dimensions (39).

The U matrix consists of a 2x2496. The number of rows (2) represent the initial number of eigenchannel components and the number of columns (2496) is a product of the number of mixture components (64) and the number of dimensions (39).

The D matrix consists of a 1x2496. The number of rows (1) represent the initial number of residual components and the number of columns (2496) is a product of the number of mixture components (64) and the number of dimensions (39).

When the score is button is clicked, the score for the selected speaker is computed. The speaker is accepted if the corresponding row value is the greatest. The first 15 rows represent microphone and the next 15 rows represent telephone channel. The score computed for speaker1 through microphone channel is shown in figure 4.19



```
 1.5063232e+000
 8.8403232e-001
 4.4498824e-001
 3.3669525e-001
 4.4251508e-001
 2.1859819e-001
-2.9274707e-002
 4.7174017e-001
 3.6652593e-001
 8.0522865e-002
 1.6064371e-001
 7.5438051e-001
 6.3505982e-004
 6.3672451e-001
-3.2363789e-002
-4.8679030e-001
-1.0447972e-001
-1.7669786e-001
 1.0129047e-002
-1.2242268e-001
 4.5894603e-002
-4.1830280e-001
-2.4044804e-001
 9.3740163e-003
-2.3391661e-001
-1.4814062e-001
-3.4765339e-001
-5.3585979e-001
-4.7363205e-001
-4.8309564e-001
```

Figure 4.15 Output score computed for speaker1 through microphone channel

CHAPTER 5

PERFORMANCE ANALYSIS

Performance analysis is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. Performance analysis is performed using **Detection Error Tradeoff (DET) graph.**

A DET graph is a graphical plot of error rates for binary classification system, plotting false reject rate vs. false accept rate.

The false acceptance rate, or FAR, is the measure of the likelihood that the speaker verification system will incorrectly accept an imposter who claims to be the actual speaker. A system's FAR typically is stated as the ratio of the number of false acceptances divided by the number of verification attempts.

The false rejection rate, or FRR, is the measure of the likelihood that the speaker verification system will incorrectly reject a true speaker. A system's FRR typically is stated as the ratio of the number of false rejections divided by the number of identification attempts.

The speaker verification system predetermines the threshold values for its false acceptance rate and its false rejection rate, and when the rates are equal, the common value is referred to as the **equal error rate**. The value indicates that the proportion of false acceptances is equal to the proportion of false rejections. The lower the equal error rate value, the higher the accuracy of the speaker verification system.

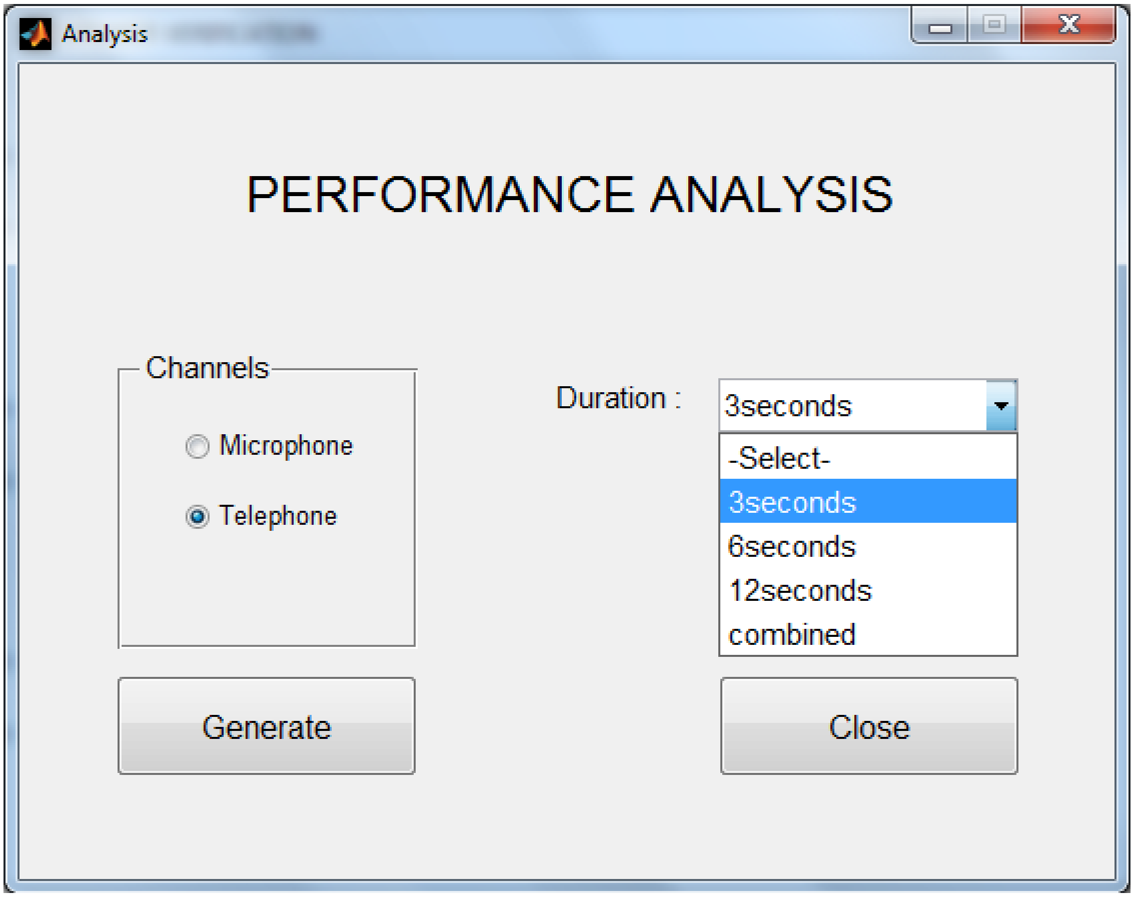


Figure 5.1 GUI Window for Performance Analysis Phase

The graphical user interface for performance analysis phase is shown in figure 5.1. It consists of a radio button to choose the channel, a drop down list box to choose the duration of data and buttons to generate the DET curve and close the GUI window.

The DET curve for 3 seconds utterance telephone data is shown in figure 5.2. From the DET curve the equal error rate can be computed. The ERR is computed by drawing a line from the origin to the point where the x coordinate and y coordinate values are the same. In the DET curve shown in figure 5.2 the ERR is 9%. The circle represents the lowest cost point.

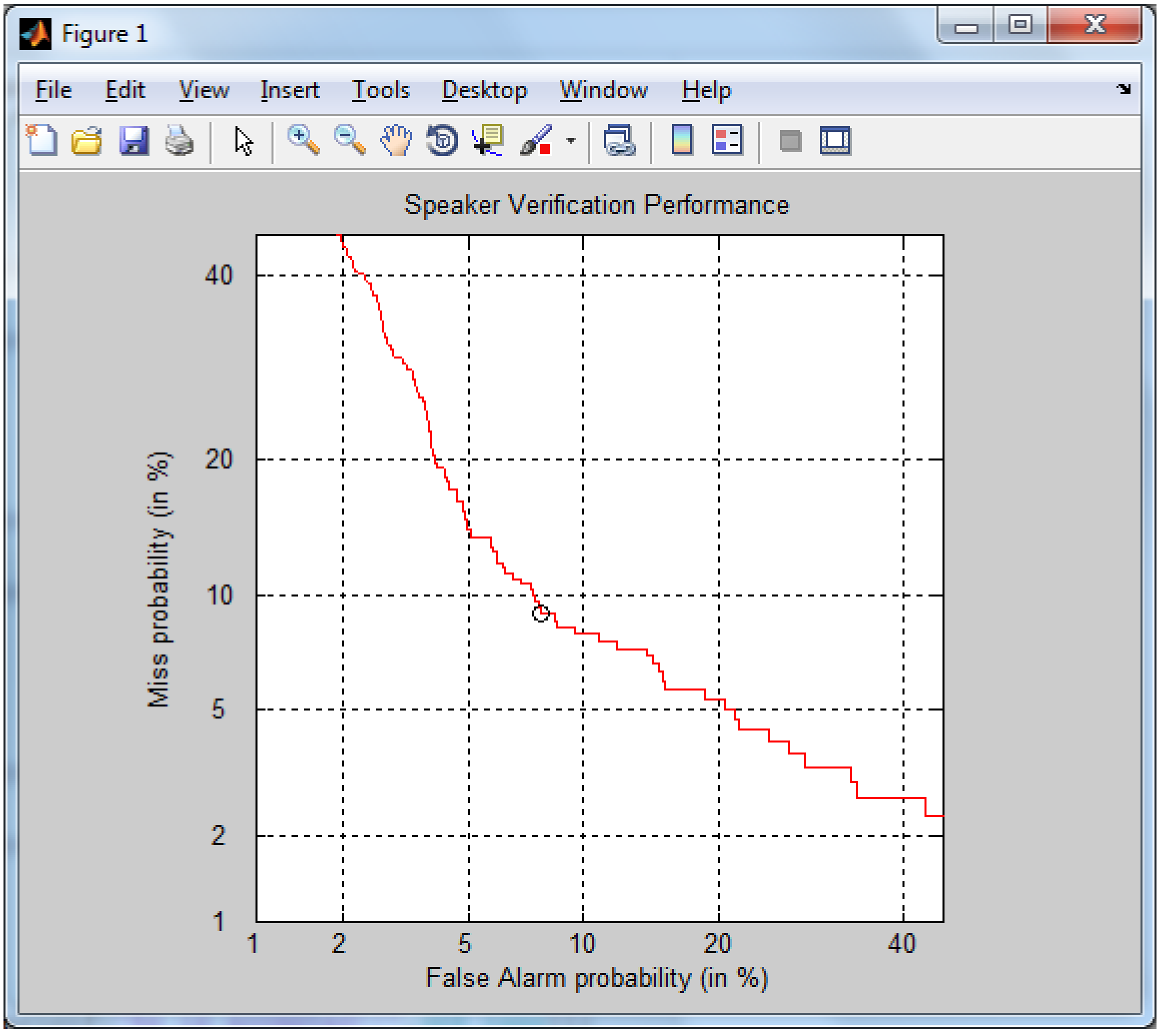


Figure 5.2 DET curve for 3 seconds utterance telephone data

It is easier to analyze the performance if we show the DET curves of all durations of data in a single graph. Figure 5.3 and 5.4 shows the DET curves of all durations of data in the microphone and telephone data respectively. The equal error rates and detection cost function values are noted for each duration of data in both channels. It is shown in Table 5.1

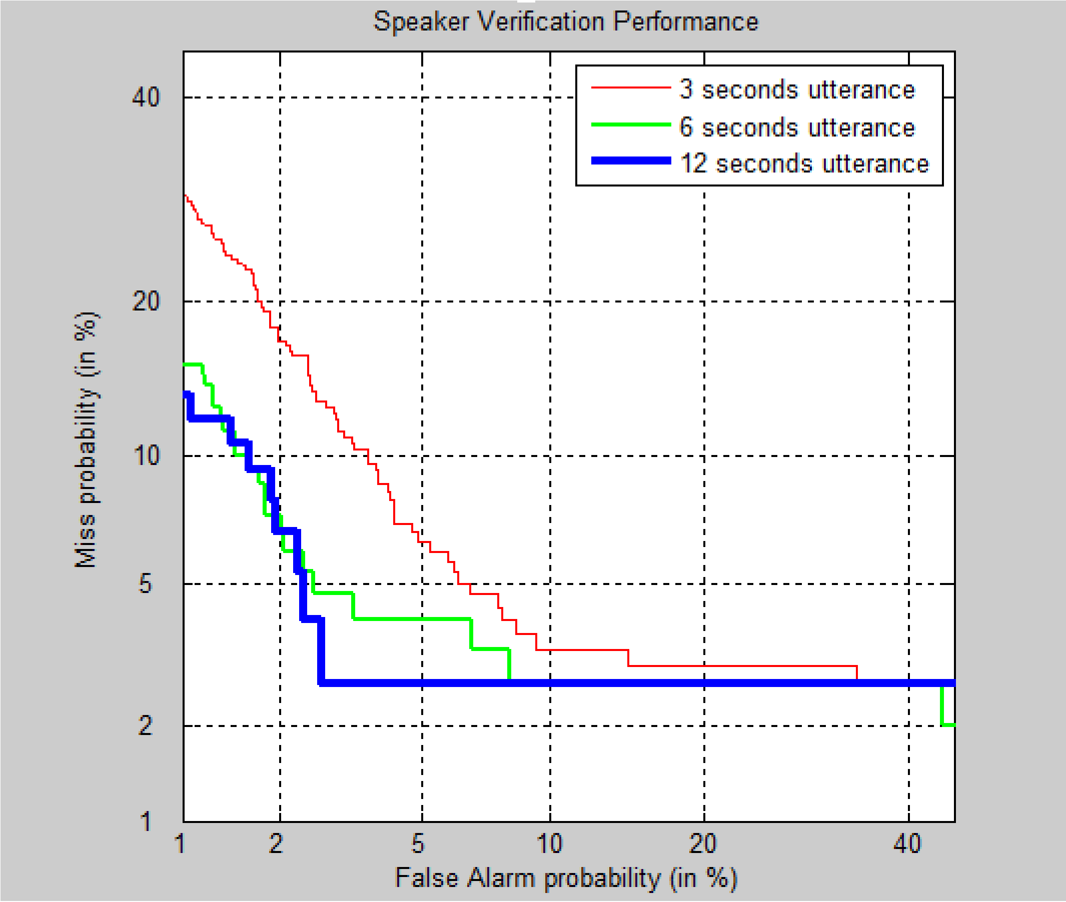


Figure 5.3 DET curves of all durations of microphone data

The ERR for 3 seconds utterance data is 6%. The ERR for 6 seconds utterance data is 4.3%. The ERR for 12 seconds utterance data is 3%. From the above observations it can be concluded that when the duration of the utterance data is increase the error rate decreases.

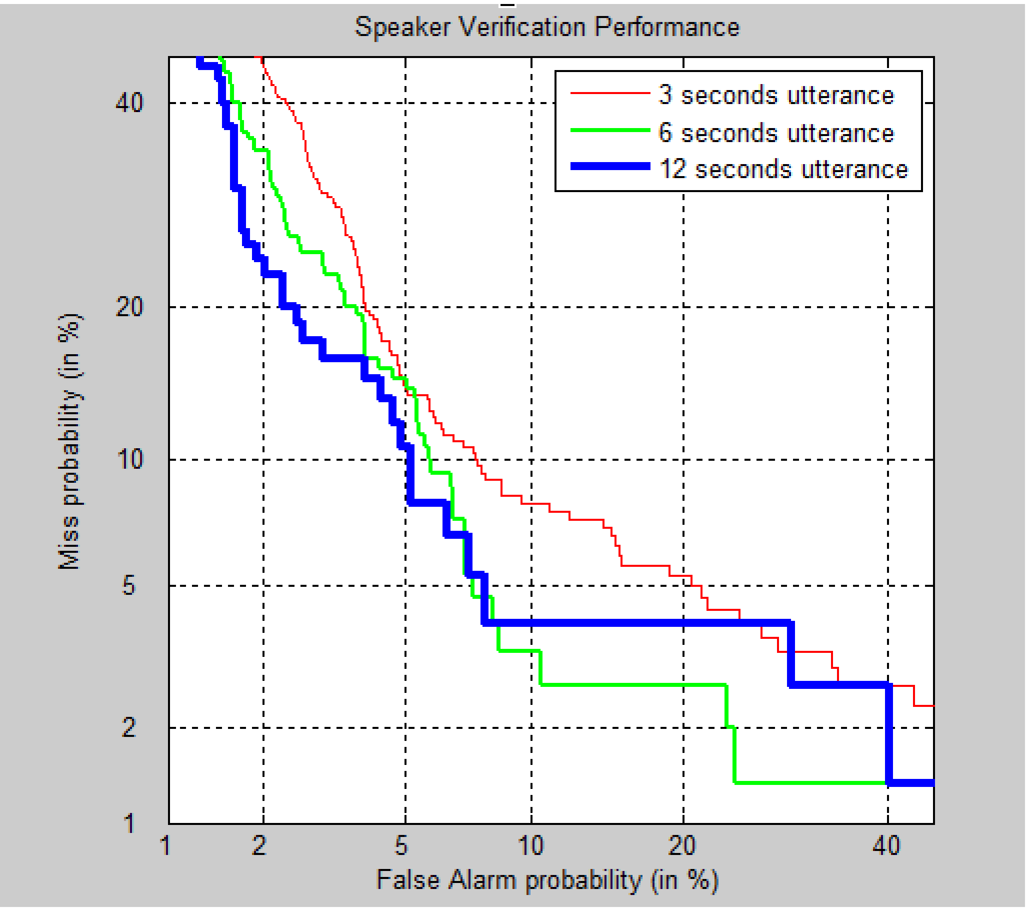


Figure 5.4 DET curves of all durations of telephone data

It can be observed from Figure 5.3 and Figure 5.4 that the microphone data show better performance compared to telephone data as indicated by the lower equal error rate values.

CHANNEL	**DATA DURATION**	**EQUAL ERROR RATE**	**DETECTION COST FUNCTION**
MIKE	12	3	0.0267
	6	4.3	0.0359
	3	6	0.0556
TELEPHONE	12	6.8	0.0595
	6	8	0.0592
	3	9.2	0.0843

Table 5.1 Performance Analysis table containing equal error rate and detection cost function values

CHAPTER 6

CONCLUSION AND FUTURE DIRECTIONS

This chapter summarizes the main components of this system as well as the major conclusions gathered from the results of the evaluation. This section also covers how the system can be further improved.

Thus, speaker verification system over various channels has been developed using Joint Factor Analysis. The utterances of 15 speakers are collected through microphone and telephone channels during training phase. In testing phase, true speaker is accepted or rejected. In impostor checking phase, if a speaker claims to be another speaker, he is either accepted or identified as an imposter. From the performance analysis phase it is understood that as the test utterance size increases the equal error rate decreases and detection cost function value increases.

In future, the speaker verification system can be improved to work across more channels. It can also be extended to address more issues such as room acoustics, change in speaker's voice due to sickness, aging, etc.

APPENDIX – 1

The source code, executables, data corpus, tools required and readme file are written onto a CD-ROM and it is attached with this report at the end page.

REFERENCES

1. A. Kanagasundaram, R. Vogt, D. Dean, S. Sridharan (2011) 'i-vector Based Speaker Recognition on Short Utterances', in Interspeech 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy, pp. 27-31.

2. Ann Lee (2012) 'A Comparison-based Approach to Mispronunciation Detection', Submitted to the Department of Electrical Engineering and Computer Science, MASSACHUSETTS INSTITUTE OF TECHNOLOGY.

3. D. A. Reynolds (1992) 'A gaussian mixture modeling approach to text - independent speaker identification', Ph.D. thesis, Georgia Institute of Technology.

4. D. Reynolds, T. Quatieri, and R. Dunn (2000) 'Speaker verification using adapted Gaussian mixture models', Digital Signal Processing, pp. 19 – 41.

5. M. Senoussaoui, P. Kenny, N. Dehak, P. Dumouchel (2010) 'An i-vector Extractor Suitable for Speaker Recognition with both Microphone and Telephone Speech', in Spoken language system, CSAIL MIT, Cambridge USA.

6. N. Dehak, P. Kenny, and P. Dumouchel (2007) ‘Modeling prosodic features with joint factor analysis for speaker verification’, IEEE Trans. Audio, Speech, Lang. Process., vol. 15, no. 7, pp. 2095 - 2103.

7. P.Kenny (2005) ‘Joint factor analysis of speaker and session variability: Theory and algorithms’, Tech. Report CRIM-06/08-13

8. P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel (2008) ‘A study of inter-speaker variability in speaker verification, IEEE Transactions on Audio, Speech and Language Processing’, vol. 16, no. 5, pp.980 - 988.

9. P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel (2007) ‘Joint factor analysis versus eigenchannels in speaker recognition’, submitted to IEEE Trans. Audio Speech and Language Processing.

10. P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel (2005) ‘Factor analysis Simplified’, Proc. ICASSP 2005, Philadelphia, PA.

11. P. Kenny and P. Dumouchel (2004) ‘Disentangling speaker and channel effects in speaker verification’, in Proc. ICASSP, Montreal, Canada, pp. 37 – 40.

12. R. Vogt, C. Lustri, and S. Sridharan (2008) ‘Factor analysis modeling for speaker verification with short utterances’, in Proc. IEEE Odyssey Workshop, Stellenbosch, South Africa.

13. S.-C. Yin, R. Rose, and P. Kenny (2007) ‘A joint factor analysis approach to progressive model adaptation in text independent speaker verification’, 23 IEEE Trans. Audio, Speech, Lang. Process., vol. 15, no. 7, pp. 1999 - 2010.

14. ‘The wavesurfer’, http://www.speech.kth.se/wavesurfer.html/.