ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS

TEXT-DEPENDENT SPEAKER VERIFICATION SYSTEM: AN EXPERIMENT USING HIDDEN MARKOV MODEL (HMM)

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

By

Fantahun Aberra Zikie

July 2004

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS

TEXT-DEPENDENT SPEAKER VERIFICATION SYSTEM: AN EXPERIMENT USING HIDDEN MARKOV MODEL (HMM)

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

By

Fantahun Aberra Zikie

July 2004

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
FACULTY OF INFORMATICS

TEXT-DEPENDENT SPEAKER VERIFICATION SYSTEM: AN EXPERIMMENT USING HIDDEN MARKOV MODEL (HMM)

By
Fantahun Aberra Zikie

Name and Signature of the Members of the Examining Board

Ato Nigusie Taddese, Chairman, Examining Board _____

Dr. Eneyew Adugna, Advisor _____

Dr. Kavi, N. Murthy, External Examiner _____

DEDICATION

This work is dedicated to my loving wife Netsanet Bogale and my Families.

I love you.

ACKNOWLEDGEMENT

All of the credits not only of working and completing this research, but also my life go to the Almighty God.

Some people deserve to be acknowledged. First of all my heart felt gratitude goes to my advisor Dr Eniyew Adugna, without whom the start to end of this research could have been a naughty.

I am greatly indebted to Ato Sebsibie, whom I met in the last but done above all. It is rarely encountered to get such a considerate person in the world. It is to my advantage.

I would also want to acknowledge Ato Zegaye Seifu and Ato Kinfe Taddese whose continuous encouragement and direction has helped me lots.

Greater sincere must also go to Ato Wondwossen Teshome, dearest friend, who has encouraged me throughout the painful time of working on the research.

Table of Contents

List of Tables and figures

List of Appendices

ABSTRACT

As the popularity of biometrics increase as a means of securing data and locations, the area of speaker recognition is emerging as an important field of current research. Since speech is our most natural means of communication having enough identifying characteristics, speaker recognition has become a desirable method of determining a person's identity. This thesis examines the field of speaker verification in which a speaker's identity claim is verified from a sample of speech. The experiment mainly focuses on HMM-based text-dependent speaker verification system experimented by using a word from the Amharic language. In the course of building the verification system, the popular Hidden Marcov Model Toolkit (HTK) is used.

For each client, six different utterances are taken in two sessions and phone-based HMM is developed. Speech of each client is collected at different sessions for testing purpose. Away from the client set a speech (of the word used in the experiment) is collected for testing purpose.

The results obtained are promising in that, only two false acceptances out of the ninety nine false claims, and one false reject out of the ninety true claims, errors are committed, giving a 0.02 FAR and 0.01 FRR. The details of the research and findings are discussed in the paper. Conclusions of the findings as well as future directions are recommended.

CHAPTER I

INTRODUCTION

1.1 Background

In the past four decades it has been observed that the experimentation and commercialization of speech processing systems has emerged and succeeded a lot. Computers can now 'understand' humans speaking in a natural manner in ordinary languages at least within a limited domain. Basic and applied research in signal processing, computational linguistics and artificial intelligence have been combined to open up new possibilities in human-computer interfaces [14]. Speech processing is the study of speech signals and the processing methods of these signals.

Speech signals are usually represented in a digital format, and the parameter vectors are matched with the phones in a given language. Thus, speech processing can be seen as the convergence of digital signal processing and natural language processing [14]. As a result, speech processing is a multidisciplinary field, encompassing information science, computer science, digital signal processing, psychology and linguistics among others [4]. The major study areas of speech processing systems are: speech recognition, speaker recognition and speech synthesis [3].

Speech recognition is a process by which a computer automatically distinguishes speech, in a sense that it transcribes speech in to text, or understands speech [1]. A speech to text system can be thought of as a voice actuated type writer [4]. Speaker recognition is the process of automatically recognizing who is speaking on the basis of

individual information contained in the speech waves [2]. And speech synthesis is a computer based program that is able to read any text audibly, whether it is directly introduced to the computer by an operator or scanned and submitted to an optical character recognition system.

The focus of this thesis is within the area of speaker recognition. It is a growing area of research, because, such systems will help in extracting information about individuals from their speech and thus good guesses to who an individual is can be arrived at [8]. From person's speech a better approximation about his or her sex, nationality, emotional and other states of affairs can also be obtained [10]. This shows that the identity of an individual can be determined or validated from the person's speech. The underlying assumption is that it is possible to collect important parameters from persons' speech so as to use the potentials of computers in automatic speaker recognition [10].

Based on the application areas for which they are intended, the design of speaker recognition systems enjoy certain tradeoffs. Speaker recognition systems can be sub classified: some of these classifications include:

- Speaker identification *vs.* speaker verification; and
- Text dependent *vs.* Text independent.

1.1.1 Speaker identification *vs.* speaker verification

Speaker identification is a process of determining which speaker among the registered ones provides a given utterance [2] [16]. This identification may be from a closed set where the one to be identified is in a set of known speakers; or open set, where the individual may or may not be in a set of known speakers [2]. Speaker verification, on the other hand, is the process by which identity claim of an individual is accepted or rejected by the system [7] [16]. Two major problems deserve to be dealt with in the design of speaker verification systems. These are false rejection and false acceptance. False rejection happens when a legitimate person is denied access and false acceptance occurs when an imposter is granted access [10] [15].

1.1.2 Text dependent vs. text independent

The other major classification of speaker recognition systems is as whether the system is Text-dependent or text-independent. Text-dependent speaker recognition systems are the one which are trained usually by short text uttered by each speaker [18]. This text is not more than some words, and the training period is short. In such systems, when an individual is to be recognized (testing phase), the unknown speaker must utter the same prescribed text as in the training. This method is suitable when the speakers are cooperative [10].

Text-independent systems use any text during both the training and recognition. In this method, more training data than the text-dependent is required because the full range of vocal sounds of a speaker must be captured during the training time so as to ensure good performance of the system. Both text-dependent and independent methods have a problem in common – because they can be deceived if some one plays back the recorded voice of a registered speaker, this can be accepted as registered speaker [2]. This problem can be resolved by using text-prompted speaker recognition method, which prompts the user with a new key sentence every time the system is used [14].

Methods of developing speaker recognition systems can also be classified with regard to the recording environment as quite or noisy, and as real time operation and batch operation based on the time of response [14].

The main concern of this paper is on the development of text-dependent speaker verification system, where the voice of individuals is recorded in a normal environment.

Text dependent speaker verification is a very promising domain of research, where real applications can be designed for telephone, computer access or banking services. The importance of text dependency lies in the fact that the input speech pronounced by the speaker can be controlled. This control is made by the recognizer which performs a

temporal segmentation of the input utterances, allowing a comparison of word utterances with pre-trained speaker models [16].

For speech, as an analog signal, in order to be utilized by a recognition (verification) system, it must be captured and converted to a digital format [15]. The two most important digital signal processing techniques (parameterization) used in the conversion process are Bank-of-filters and Linear Predictive Coding (LPC) [15]. Bank-of-filters coding uses a set of filters to segment each sample into a collection of frequency bands and each frequency band is converted into a vector of acoustic parameters using Fast Fourier Transform (FFT) [15]. LPC is predicated on the idea that it is possible to estimate the values of acoustic parameters from an incoming sample by using the parameter values from previous samples [15].

1.1.3 Application Areas of Speaker Verification Systems

The number of applications of speaker verification is increasing steadily. They perform a broad spectrum of functions, including monitoring convicted criminals, securing data and data networks, protecting buildings and other physical locations, monitoring time-and-attendance of employees, and securing transactions over the telephone.

Some of the application areas of speaker verification systems can be summarized as follows [2] [3] [13]:

- Secure access to services via the telephone, house shopping, and home banking.
- Detecting and tracking of speaker in a telephone conversation

- Secure access to information which can be obtained through the Internet.
- When identity verification is needed, speaker verification is can be used to improve existing security measures i.e. PIN or password.
- Speaker verification systems can be used as gate keepers.

1.2 Statement of the problem and Justification of the study

Because of the fact that speaker verification systems can substitute the traditional person verification strategies for accessing secure things, intensive researches has been and are still conducted throughout the world. The ultimate goal of such researches is to develop full fledged speaker verification systems capable of verifying, with minimal or no error, for the sake of authority granting to an individual.

Researches in speaker verification area, including this, are intended to address some problems. These are:

- Conventional ways of granting an individual an authority of accessing secure information, allow to secure places, operating secure hardware or software are done by using such identity figures as PIN or password. Other traditional ways use locks. The PIN or password can be forgotten or exposed to others. Traditional locks can be stolen, lost or broken.

- Visually disabled people can not enter their PINs or passwords to the place they want to get access of and may be unable to use traditional locks.
- Since speech of an individual is unique in characteristics for the fact that individuals differ greatly in behavioral and physiological aspects of speech production system, speech can be used alone or in combination to other mechanisms to verify an individual.
- Speaker verification systems are intended to be language independent which is a postulate of speaker verification systems. All literatures I could read on speaker verification systems in different countries, show that they are experimenting on their local languages.

Experiments are proceeding in different countries in different languages. No work has been reported in Amharic or any language in Ethiopia. This very fact, together with the academic requirement of the department justifies the study. As well, since the study is a first trial, it can be used as a good starting point for further researches or projects on speaker verification system development in the country.

It must be noted that researches conducted in one country have a tendency to be more practical for users in the country.

In the past four decades, intensive researches have been conducted through out the world in the area of speech processing in general and speaker verification in particular.

Many Dozens of speaker verification systems are developed for varying applications. Researches in speaker verification system development in different countries throughout the world are responses to the needs of the area.

In Ethiopia some researches in the area of Amharic speech processing have been conducted at Addis Ababa University by some post graduate students of the faculty of Informatics and technology. The works of Solomon [19], Kinfe [6], on Amharic speech recognition and works of Laine [9] on Amharic speech synthesis can be cited as encouraging researches in Amharic speech processing. To my knowledge, no work is reported on experiment on speaker verification system development in the country. **Therefore this research is intended to experiment HMM-based text-dependent speaker verification system using a word from Amharic language**.

1.3 Objectives of the Research

1.3.1 General Objective

The general objective of the research is to conduct an experiment on developing text-dependent speaker verification system in a word from Amharic language using Hidden Markov Model Toolkit (HTK) [20].

1.3.2 Specific Objectives

1. Acquisition of speech from some individuals
2. feature extraction, selection and storing feature vectors
3. Experiment the system development using the HMM toolkit (HTK)
4. Testing and evaluating the system
5. State recommendations for further study

1.4 Research Methods

To accomplish the above set objectives of the study, the following methods have been employed.

1.4.1 Review of related literature

As a starting point of any study, materials related to the problem at hand should be consulted as a basis of obtaining directions. Based on this reality, a thorough analysis of papers on speaker verification systems, biometrics, Hidden Marcov Modeling of speakers and the related toolkit (HTK) have been thoroughly reviewed. Emphasis has also been given to achievements obtained to date. Moreover, models developed by other researchers on speaker verification systems have been reviewed.

1.4.2 Planning and Data Preparation

Planning is a first step towards the study; and at this stage, the requirements of the system, what the system should look like, and how it should work was set. The data needed for the experiment, the speech data, was collected from different individuals at different sessions.

1.4.3 Selection of Models for the Experiment

There are two models that have been extensively used in speaker verification systems: **stochastic models** and **template models**. A very popular stochastic model for modeling speech production process is the Hidden Markov Model (HMM). In HMM the observations are the probabilistic function of state i.e. the model is a doubly embedded stochastic process where the underlying process is not directly observable (hidden). The template model attempts to model the speech production process in a non-parametric manner by retaining a number of sequences of feature vectors derived from multiple utterances of the same word by the same person. Template models dominated early work in speaker verification and speech recognition because the template model was intuitively more reasonable, because of its simplicity. However, recent work in stochastic models has demonstrated that these models are more flexible and hence allow for better modeling of the speech production process. A very popular stochastic model for modeling the speech production process is the Hidden Markov Model (HMM) [7].

In this research a left-to-right HMM model is used for speaker modeling. Appropriate tools from HTK were used for the experiment.

1.4.4 Testing and evaluation of the system

Testing the system is done for accuracy by using the test and training data of the individuals that participate in the study. At this stage, the pattern matching process involves the evaluation of the likelihood that a given set of input feature vectors are generated by the speaker model for the claimed identity and computing the matching score, since the speaker model is to be developed by HMM. Based on a threshold, the system is made to accept or reject the claimed identity.

The evaluation results are reported according to the errors committed by the system: the percentage of false acceptance and false rejection.

1.5 Purpose of the Research

Since the research is conducted as an academic exercise, it serves to fulfill the requirement of the Masters of Science program in which the researcher is enrolled in. Moreover, the system can be used by any one who can speak the word for which the system is trained.

As the research in this area is new in the country, the results of the research can be used as an input to the development of a full-fledged speaker verification system in the country.

1.6 Organization of the paper

The paper is organized in five chapters. The **first chapter** introduces the basic concepts of speech processing in general and speaker verification in particular including definition of basic terms and the major application areas. The statement of the problem as well as objectives of the study has been mentioned. In the last section the methods employed in conducting the research is briefed.

In **Chapter two**, the basic ideas of biometrics and the use of voice in speaker verification is thoroughly covered at a reasonable level of detail. In this chapter, the biometric recognition systems and the underlying assumptions, and biometric technologies are discussed. The chapter also discusses the subsystems of speaker verification as well as the human speech production system – the vocal tract.

Chapter three is devoted to the explanation of the basic principles of Hidden Markov Modeling and introduces the HTK toolkit.

In **chapter four** the experiment on developing speaker verification system using HTK is presented. The design and implementation details of the system are discussed. The results achieved in the experiment are also discussed.

In the last chapter, **chapter five**, the conclusion drawn from the experiment is presented. Recommendations for future work are also forwarded in this chapter.

forgotten and tokens can be lost [11]. Biometric person recognition operates by recognizing an individual by means of distinctive personal characteristics, such as face, fingerprint, iris and voice. The advantage of a biometrics approach is that these characteristics cannot be forgotten, lost or stolen, however may degrade overtime [11]. Moreover, clever use of biometrics may make access and checking procedures faster and friendlier, and the accuracy can be improved substantially. Still, this does not mean that, biometrics are a solution for all automatic recognition purposes. Especially, constructing a biometric feature set which is measurable, distinctive and robust is a non-trivial task. One major problem in biometric recognition is that one can not share his authority to others by providing his pass criteria.

The main idea behind using biometrics for recognition purposes is that a biometrics of an individual is completely different from others, so that if it is possible to extract the biometrics, it will be possible to identify a person uniquely.

2.1.1 Biometric recognition systems

Before a biometric recognition system is able to recognize an individual based on the characteristics, it is necessary that the system is acquainted with the individual's biometric characteristics. This logically divides each system into two separate modules: an enrollment (or training) module and a recognition (or testing) module [15]. Both the enrollment and the recognition module make use of a feature extraction sub-module,

2.1.2 Biometric technologies

We distinguish between two types of biometrics. One is based on the physical characteristics of an individual, whereas the second is based on an individual's behavior. The border between these two is not always strictly clear: physical characteristics can be modified by behavior, and behavior can be induced by physical characteristics. Nevertheless, we use the following distinction: [8]

2.1.2.1 Physical biometrics

Use concrete physical characteristics of an individual. Most physical characteristics are relatively stable over time and basically unalterable. Face, fingerprint, hand and finger geometry, and iris recognition belong to this category.

Let's consider finger print as an example to the physical biometrics;

Fingerprint

The skin on the inside surfaces of our hands, fingers, feet, and toes is ridged or covered with concentric raised patterns. These ridges are called friction ridges and they serve the useful function of making it easier to grasp and hold onto objects and surfaces without slippage. It is the many differences in the way friction ridges are patterned, broken, and forked which make ridged skin areas, including fingerprints, unique. Fingerprints of identical twins are different and even the prints on the fingers of one individual show no correlation. The fingerprint does not change over time as the individual becomes older. The most common (and accurate) recognition approach is to

analyze the details of the fingerprint: the end points, bifurcations and divergences of print ridges. The uniqueness and stability of the fingerprint make that performance of a fingerprint-based biometric system be very high. Still, errors do occur, mainly false rejects, not due to failing technology, but because of difficulties in obtaining a representative sample: incorrect placement of the finger on the sensor, rotational variance of the fingerprint image, or image distortions because of dirt or a bad physical condition of the fingertip.

One of the mentioned strength of a fingerprint biometrics it is easy to obtain. And its weakness is the quality of the samples and impossibility to use for remote recognition.

2.1.2.2 Behavioral biometrics

Behavioral biometrics use characteristics that reflect the behavior of an individual [11]. Voice and dynamic signature recognition are the most prominent examples. The essential property of behavioral biometrics is that the individual under test has to perform an act (e.g. write or speak) before recognition can take place. To a large extent, an individual has the capability to change his/her behavioral characteristics, although physical properties, such as size and gender, have a major influence. Additionally, behavioral characteristics are known to change over time, both long-term (aging) and short-term (health, psychological factors).

A good example of behavioral biometrics is voice. The human voice does not exist without the effort of a human, namely producing speech. So recording the human voice for recognition purposes requires a human to say something, in other words, to show some of his/her speaking behavior. Therefore, voice recognition fits within the category

of behavioral biometrics. Speech is a very rich medium of communication. When humans speak, they communicate not only a message (*what* is said), but also their voice characteristics (*who* said it) and attitude (*how* it is said). The voice characteristics of a speaker are determined largely by the size and shape of the vocal tract [8] [11]. The vocal tract is generally considered as the speech production organs above the vocal folds as shown in ***Fig. 1.1*** below. However, extraction of the voice characteristics from the speech is far from trivial, because these characteristics are obscured to some extent by the linguistic content and, more importantly, by the attitude of the speaker.

Differences in psychological and behavioral aspects of speech production system in human beings cause speaker-specific characteristics of speech. The shape of the vocal tract is the major physiological aspect playing major role in speaker-specific characteristics of speech. The vocal tract consists of the following, as illustrated by Fig.1.1.

1. Laryngeal pharynx
2. Oral pharynx
3. Oral cavity
4. Nasal pharynx
5. Nasal cavity

Speech is produced when acoustic wave passes through the vocal tract, and the spectral content of the wave is modified thereby. Biometric studies reveal that the shape of the vocal tract varies from individual to individual, and no two persons can have the same vocal tract shape [7].

Therefore, it is promising to make use of features derived only from vocal tract in speaker verification systems.

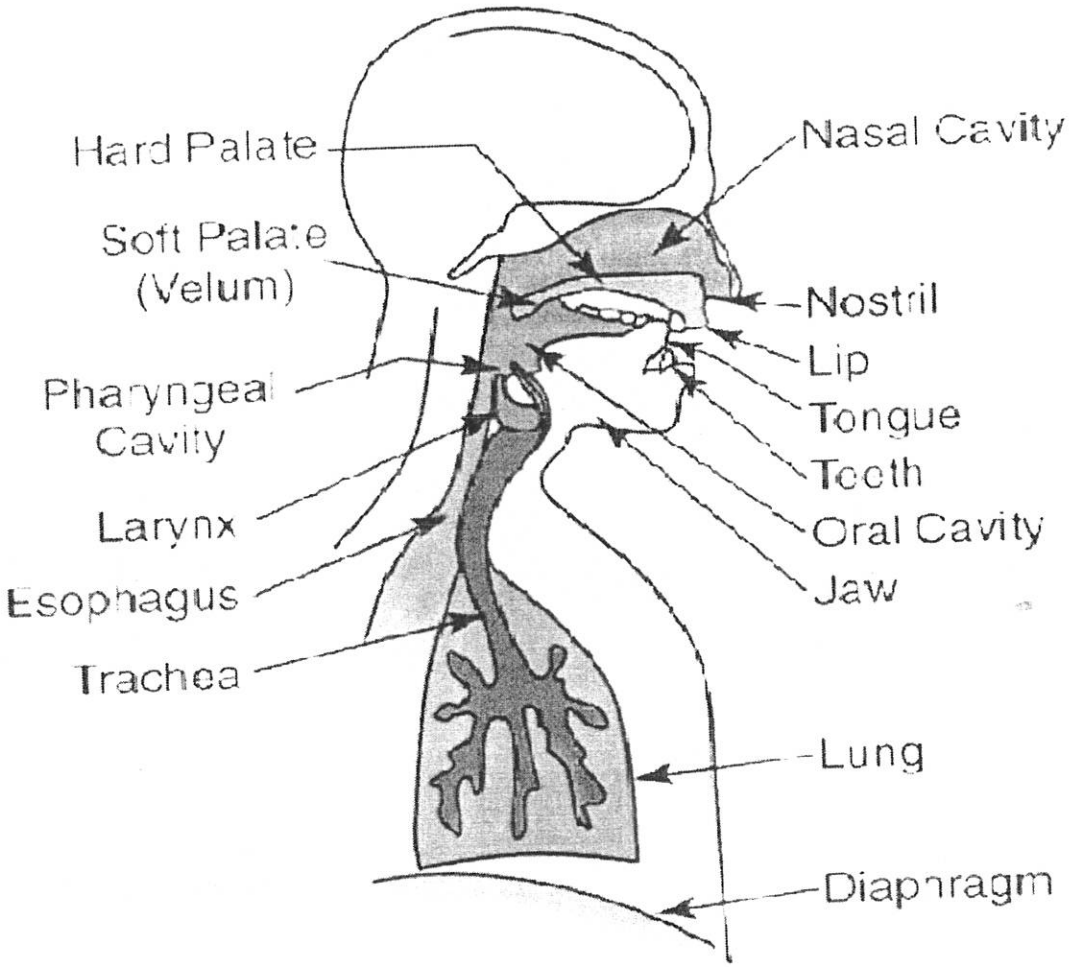

Fig.2. 1 Speech Production parts in Human Beings

Guided by the nervous system, the acoustic wave is produced when airflows from the lungs is carried by the trachea through the vocal folds [6][8]. (Fig.1.1). Excitation of the

acoustic wave produces phonation, whispering, frication, compression, vibration and any combination of these. When the air flow is modulated, by the vocal folds, the resulting excitation is said to be phonated. If the airflow rushes through a small triangular opening, the arytenoids cartilage at rear of the nearly closed vocal folds, a whisper is produced. Constriction in the vocal tract produces frication excitation. When a completely closed pressurized vocal tract is opened, compression excitation results. And at last, vibration excitation is caused by air being forced through a closure other than the vocal folds, especially at the tongue.

A phonated excitation produces speech which is called voiced, speech produced by phonated excitation and frication is called mixed voiced, while speech produced by other types of excitation is called unvoiced [8].

2.1.2.2.1 Intra-speaker Variations

Variation in linguistic content and attitude is also referred to as intra-speaker variation. Below is a brief discussion of two components of intra-speaker variation.

- The linguistic content is determined by the context of the dialogue, the choice of words, and by the way the words are pronounced, ranging from standard language, via regional variants, to dialects. Although, for example, the use of a specific word or a dialect might give a clue about who the speaker is, these types of clues are not very stable because they are under the speaker's control, and he/she can easily decide to use another wording or the standard language next time [2].

- The attitude of a speaker can serve a vast number of purposes, e.g. to emphasize specific words, to express emotions, or to communicate a subliminal message. Some aspects of attitude are within the speaker's control (speaking rate, sloppiness, loudness, emotion), others are not (health, aging, tiredness) [7].

Another source of variation emerges when speech is recorded [17]. Environmental noise might be added to the speech and also the transmission characteristics of the microphone or the telephone channel leave their mark on the recorded signal. This is schematically depicted in ***Fig.2.2.*** Where environmental noise e (t) is added to the original speech signal s (t) and the total signal is passed through a transmission channel with characteristic h (t). The recorded signal can be written as r (t) = (s (t) + e (t)) ⊗ h (t), with ⊗ denoting the convolution operation [15]. Effective removal of both the additive and the noise from the recorded speech signal is a real challenge to speaker recognition technology.

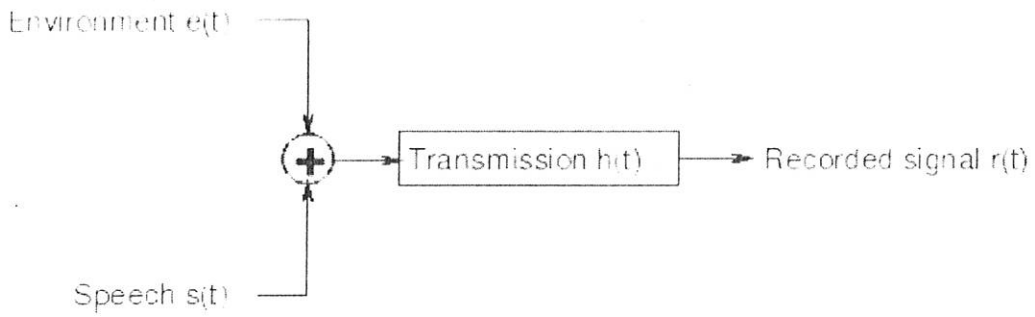

Fig.2. 2 Environment and transmission effects in recorded speech

Despite the existing challenges to technology, speaker verification has shown to be very effective especially in application areas where remote identity verification is necessary, such as secure access to services via the telephone [13]. When the intra-speaker variation in the speech is limited by allowing only words from a small vocabulary (with the most obvious example being the set of digits 0-9), performance can approach the 1% error rate, even in realistic acoustic conditions [8]. Speaker verification will probably never perform well in all possible recording conditions, using any amount of speech no matter how little, and with all possible kinds of linguistic and speaker attitude variation, but if one or a few of these standards are relaxed for a specific application of speaker verification, the Speaker Verification System performance could still be acceptable [10]. A cited strength of voice biometrics is its possibility of remote application (verification), including telephone and the Internet [13].

Ideally, the biometric features describing an individual's voice should be as insensitive as possible to intra-speaker variation. Additionally, the speaker modeling technique should be able to cope with feature variation due to intra-speaker variation [13].

2.2 Speaker Verification

Speaker verification means determining whether an unknown voice matches the known voice of the speaker whose identity is being claimed [2]. A binary comparison is involved in which the system returns either "accept" or "reject" according to whether it

believes the speaker is who he/she says he/she is [11]. The following figure, ***Fig.2.3.*** describes a block diagram of speaker verification system.

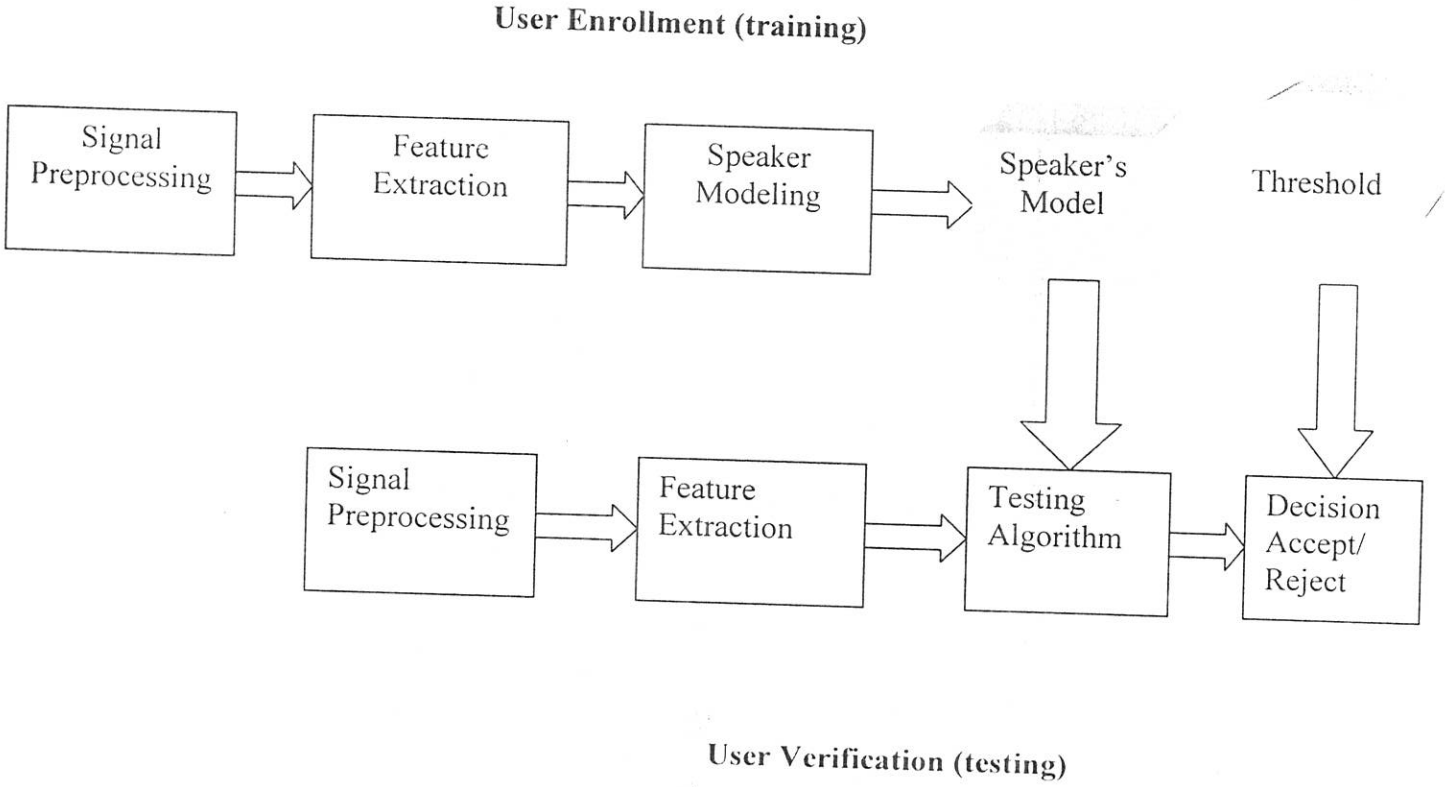

Fig.2. 3 High-level diagram of Speaker Verification system

There are three steps involved in a speaker verification system [7] [18].

The first step is referred to as enrollment. This is when a model for each speaker is created from a set of training data. The next step is the verification step in which a speech signal is input, necessary features are extracted and the system compares this with the reference model. A decision is then made as to whether to accept or reject the claimant based on the decision rule. Finally, the reference templates may be updated after verification is successful if changes are found in the user's speech and will help in future sessions [21].

If different individuals are given the same word to utter, the excitation parameters of all the individuals, under the same environment are almost similar, but vary considerably in parameters derived from vocal tract responses. This makes speech a promising biometrics to be used in speaker verification.

2.2.1. Speech processing

Speech parameters are obtained from speech of an individual. Before a speech signal can be processed by a computer it must be converted from an analogue to a digital form which is called sampling. The most common sampling technique is based on the ***Nyquist sampling theory*** [15]. The Nyquist sampling theory states that if an analog signal is sampled at regular intervals at a rate at least twice the highest frequency in the channel, the samples will contain sufficient information about the signal to allow its reconstruction. Thus the analogue signal must first be band-limited and then sampled at fixed time intervals. The sampling frequency generally ranges from 8 kHz for telephone quality speech to 48 kHz for digital audio tapes. The maximum spectral frequency that can be represented is half the sampling frequency, e.g. 4 kHz for telephone speech.

spectrum and the higher order coefficients are concerned with increasingly finer features in the spectrum [16]. The other approach is based on a linear predictive coding (LPC) scheme.

Properties of the Cepstrum

When a speech signal is passed through a linear filter, the cepstrum of the filtered signal is equal to the sum of the cepstrum of the input speech signal and the cepstrum of the filter. If we assume that the filter is time-invariant, and that the speech utterance is long enough such that the input speech energy is uniformly distributed over the entire range of the spectrum, the cepstrum of the filter can be obtained by averaging the cepstrum of the output signal over the entire signal duration. Subtraction of the time average from the output cepstrum delivers the cepstrum of the original signal.

$$\hat{c}_n(t) = c_n(t) - \frac{1}{T}\sum_{\tau=1}^{T} c_n(\tau)$$

This technique can be used to undo the linear filtering effects of the transmission channel in speaker verification and speech recognition over the telephone. The technique is known as Cepstral Mean Subtraction (CMS) [7]. However, in most practical applications the transmission channel also has non-linear components which are not removed by CMS. Moreover, the speech signal seldom has duration long enough to assume a flat long-term average spectrum, causing CMS to remove some speech and/or speaker information in addition to the channel component. Still, CMS is one of the most popular and efficient channel normalization techniques. Sometimes the task requirements make it impossible to calculate the cepstral mean over the whole utterance. In these cases, the cepstral mean can

CHAPTER III

HIDDEN MARKOV MODEL AND HTK

3.1 Hidden Markov Models

One of the best known strategies for generating word models of indviduals is the Hidden Markov Model. The Hidden Markov Model is a finite state machine which has associated probability distribution [4]. Transitions among the states depend on a set of probabilities called *transition probabilities*. In a particular state an outcome or *observation* can be generated, according to the associated probability distribution. It is only the outcome, not the state, that is visible to an external observer and therefore states are "hidden" to the observer; hence the name Hidden Markov Model.

HMMs are defined based on the following elements

- The number of states of the model, N.
- The number of observation symbols in the alphabet, M. If the observations are continuous then M is infinite. A = $\{a_{ij}\}$
- A set of state transition probabilities.

  $a_{ij} = p\{q_{t+1} = j \mid q_t\}, 1 \le i, j \le N$

  where q_t denotes the current state. Transition probabilities should satisfy the normal stochastic constraints,

  $a_{ij} \ge 0, 1 \le i, j \le N$ and

$$\sum_{j=1}^{N} a_{ij} = 1, \quad 1 \le i \le N$$

- A probability distribution in each of the states, $B = \{ b_j(k) \}$

$$b_j(k) = p\{o_t = v_k | q_t = j\}, \quad 1 \le j \le N, \quad 1 \le k \le M$$

where v_k denotes the k^{th} observation symbol in the alphabet, and u_t the current parameter vector.

The following stochastic constraints must be satisfied also.

$$b_j(k) \ge 0, \quad 1 \le j \le N, \quad 1 \le k \le M$$

and

$$\sum_{k=1}^{M} b_j(k) = 1, \quad 1 \le j \le N$$

If the observations are continuous then we will have to use a continuous probability density function, instead of a set of discrete probabilities. In this case we specify the parameters of the probability density function. Usually the probability density is approximated by a weighted sum of M Gaussian distributions $\mathcal{N}$.

$$b_j(o_t) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(\mu_{jm}, \Sigma_{jm}, o_t)$$

where,

c_{jm} = *weighting coefficients*

μ_{jm} = *mean vectors*

Σ_{jm} = *Covariance matrices*

c_{jm} should satisfy the stochastic constrains.

$$c_{jm} \geq 0, \quad 1 \leq j \leq N, \quad 1 \leq m \leq M$$

and

$$\sum_{m=1}^{M} c_{jm} = 1, \quad 1 \leq j \leq N$$

- The initial state distribution, $\Pi = \{\Pi_i\}$

.

where,

$$\pi_i = p\{q_1 = i\}, \quad 1 \leq i \leq N$$

Therefore we can use the compact notation

$$\lambda = (B, \pi)$$

to denote an HMM with discrete probability distributions, while

$$\lambda = (c_{jm}, \mu_{jm}, \Sigma_{jm}, \pi)$$

to denote one with continuous densities. .

3.2 The HTK toolkit

HTK, Hidden Markov Model Toolkit, was developed at the Speech Vision and Robotics Group of the Cambridge University Engineering Department (CUED) in 1989 by Steve Young as a first version. HTK was (and is) a set of C library modules and tools that was initially used for speech recognition research (using continuous density HMMs) within the Speech Vision and Robotics Group at Cambridge. It soon became apparent that the software was of general interest and the early versions were distributed in source form at some cost. Different versions of HTK have been released till version 3.2.1, which is a current stable release. The experiment exercised here depends on this release. This time HTK is an open source for experimental purposes [21].

HTK consists of a set of library modules and command line tools. Each tool in HTK uses these standard library modules to communicate with the outside world. The library modules in HTK are HSHEL, HMEM, HMAT, HSIGP, HLABLE, HLM, HNET, HDICT, HVQ, HMODEL, HAUDIO, HGRAF, HUTIL, HTRAIN, HFB, AND HREC while the command line tools include HBUILD, HCOMPV, HCOPY, HDMAN, HEADAPT, HEREST, HHED, HINIT, HLED, HLIST, HLSTATES, HPARSE, HQUANT, HREST, HRESULTS, HSGEN, HSLAB, ASMOOTH and HVITE. [20] [21].

Even if HTK is primarily developed for speech recognition research, it can be used for various researches in the area of speech processing and pattern matching. Speaker verification researches like the CAVE (Caller Verification) and MASV (**M**unich **A**utomatic **S**peaker **V**erification) have used the toolkit by extending the tools.

All of the operating system and user interface functions are provided by the HTK module HShell. Low level memory management in HTK is provided by the module HMem and the management of higher level structures such as vectors and matrices is provided by HMath. HTK tools are executed by giving commands to the operating system shell. Each command typically contains the names of the various files that the tool needs to function and a number of optional arguments which control the detailed behavior of the tool [19].

Every HTK tool uses a set of standard library modules to interface to the various file types and to connect with the outside world. Many of these modules can be customized by setting parameters in a *configuration file*. To this end, it is possible to specify a small number of parameters using environment variables.

3.2.1 The Command Line

The general form of command line for invoking a tool is:

tool [options] files ...

Options always consist of a dash followed by a single letter. Some options are followed by an argument as follows:

-t 6 - an integer valued option

-i - a switch option

-a 0.007 - a float valued option

-s ok - a string valued option

Option names consisting of a capital letter are common across all tools. Integer arguments may be given in any of the standard C formats in Decimal, Hexadecimal or Octal, for example, 14, 0xE and 016 all represent the same number. Typing the name of a tool on its own always causes a short summary of the command line options to be printed in place of its normal operation. Example typing HCOPY on the command line produces the following.

C:\>hcopy

USAGE: *HCopy [options] src [+ src ...] tgt ...*

Option		Default
-a i	Use level i labels	1
-e t	End copy at time t	EOF
-i mlf	Save labels to mlf s	null
-l dir	Output target label files to dir	current
-m t	Set margin of t around x/n segs	0
-n i [j]	Extract i'th [to j'th] label	off
-s t	Start copy at time t	0
-t n	Set trace line width to n	70
-x s [n]	Extract [n'th occ of] label s	off
-A	Print command line arguments	off
-C cf	Set config file to cf	default
-D	Display configuration variables	off
-F fmt	Set source data format to fmt	as config
-G fmt	Set source label format to fmt	as config
-I mlf	Load master label file mlf	
-L dir	Set input label (or net) dir	current
-O	Set target data format to fmt	as config
-P	Set target label format to fmt	as config
-S f	Set script file to f	none
-T N	Set trace flags to N	0
-V	Print version information	off
-X ext	Set input label (or net) file ext	lab

3.2.2 Data preparation tools

Data preparation is a process of collecting and smoothing speech data together with the corresponding transcription that is needed for system development. HTK tools are primarily made for speech recognition purposes. The data preparation tools, but, fits to any speech processing task [21].

In order to build a set of HMMs, which are needed to represent an individual, a set of speech data files for each individual and their associated transcriptions are required. For speaker verification, the transcriptions are required only to make use of the training and testing tools. Mostly speech data will be obtained from database archives, typically on CD-ROMs.

Before it can be submitted to the training tools, the speech data must be converted into the appropriate parametric form and any associated transcriptions must be converted to have the correct format and use the required – phone, syllable or word labels. When recording the speech data from scratch, the tool HSLab can be used both to record the speech and to manually annotate it with any required transcriptions.

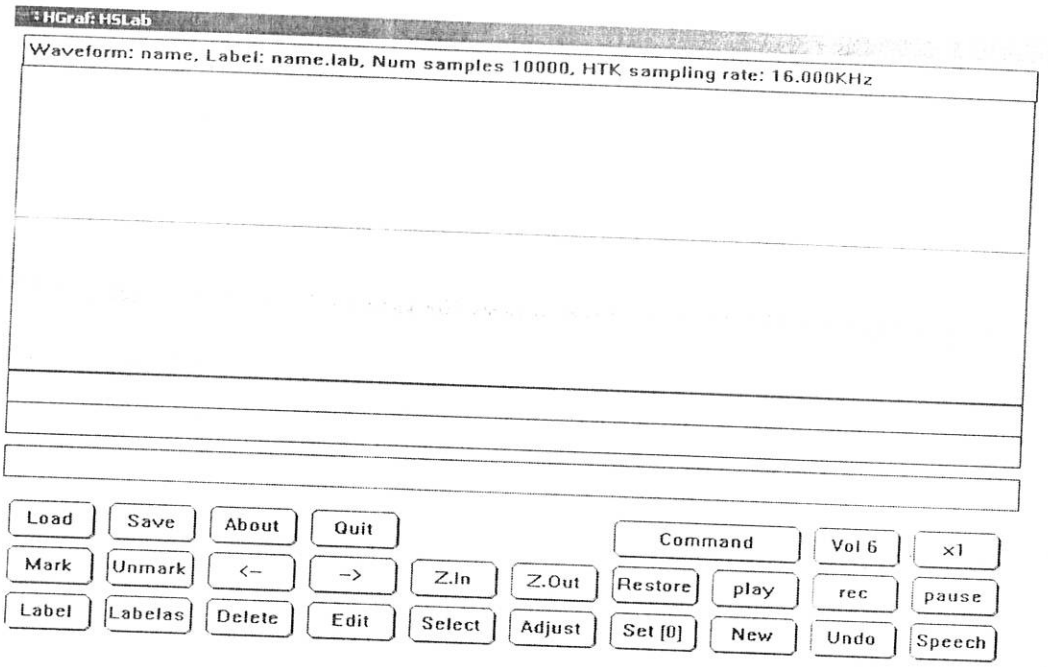

Fig.3. 2 Graphical interface of HSlab

The tool HCopy is used for parametrizing the wave forms. It is also possible that all HTK tools can parameterize waveforms *on-the-fly*. As the name implies that HCopy is used to copy one or more source files to an output file converting the source kind to a target kind based on the given configuration variables as they are read in. Thus, simply copying each file in this manner performs the required encoding.

The general form of HCOPY invocation is:

HCOPY src tgt

Copies from source to target file with corresponding formats. Valid parameter conversion is given in figure below.

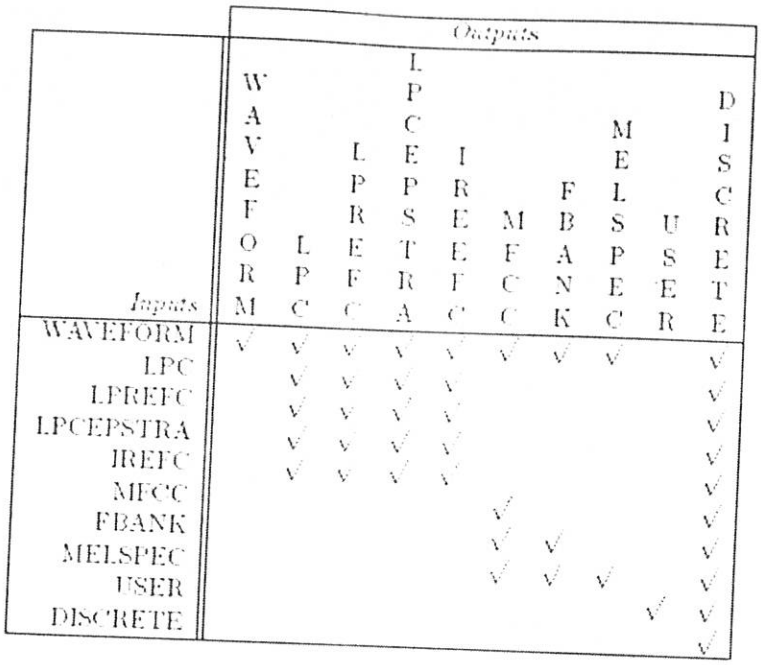

Fig.3. 3 Valid parameter conversion

The tool HList can be used to check the contents of any speech file and since it can also convert input on-the-fly, it can be used to check the results of any conversions before processing large quantities of data. Transcriptions will also need preparing. Typically the labels used in the original source transcriptions will not be exactly as required, for example, because of differences in the phone sets used. HLed is an editor tool for manipulating label files.

HLed is invoked by a command line

HLED [options] cmdFile labFiles ....

HLed is used to make word (or phone) level transcriptions of training data. It can also output files to a single *Master Label File* MLF which is usually more convenient for successive processing. The last stage in data preparation tools in HTK are HLStats, which can gather and display statistics on label files and where required, HQuant which can be

used to build a VQ codebook in preparation for building discrete probability HMM system.[21]

3.2.3 Training tools

The second step of system building is to define the topology required for each HMM by writing a prototype definition.

The process of parameter estimation is called *training*. HTK supplies four basic tools for parameter estimation: HCompV, HInit, HRest and HERest. HCompV and HInit are used for initialization parameter values, for mean, variance and transition probabilities. HCompV will set the mean and variance of every Gaussian component in a HMM definition to be equal to the global mean and variance of the speech training data. This is typically used as an initialization stage for *flat-start* training [21].

HCompV is invoked via the command line:

HCompV [options] hmm trainFiles, where hmm is the name of the physical HMM whose parameters are to be initialized.

HInit can alternatively be used as a more detailed initialisation to compute the parameters of a new HMM using a Viterbi style of estimation [21].

HRest and HERest are used to refine the parameters of existing HMMs using Baum-Welch Re-estimation. Like HInit, HRest performs *isolated-unit* training whereas HERest operates on complete model sets and performs *embedded-unit* training. In general, whole word HMMs are built using HInit and HRest, and sub-word based systems are built using HERest initialized by either HCompV or HInit and HRest [21].

HEREST is invoked via the command line

HEREST [options] hmmlist trainFile..., which causes the hmmlist to be loaded and its parameters to be re-estimated using data obtained from the trainfile. When need arises it is possible to store the training files in a script file. The first requirement to use HEREST is to create a file containing a list of all HMMs in the model set with each model name written on a separate line. The names of the models in this list must correspond to the labels used in the transcriptions and there must be a corresponding model for every distinct transcription label.

3.2.4 Recognition tools

HVite is the only recognition tool provided by HTK, which is a general purpose word recognizer.

HVite is invoked by the command line argument:

HVite [options] dictionary hmmlist testFiles....

Given a network of HMMs, it matches the HMM network with the speech file. At the simplest level, a label file is read and expanded making use of the dictionary.

HVite takes as input a network describing the allowable word sequences, a dictionary defining how each word is pronounced, and a set of HMMS. It operates by converting the word network to sub word network and attaching the appropriate HMM to the sub word units. Recognition can be performed on either a list stored speech files or on direct audio input.

As described at the start of the chapter, HTK tools can in similar fashion be used for speaker verification. All points discussed for training and recognition follow the same track.

If no obscuring options are used, a typical output of HVITE has the following format.

Start_time End_time word Log likelihood

For example:

0 1220000 one -34820.1543 can be an output of the tool HVite, in recognizing the word '*one*'. The first two columns are to do with time alignment. It is the fourth column that we are interested in. This column contains the log-likelihood values for a word spoken.

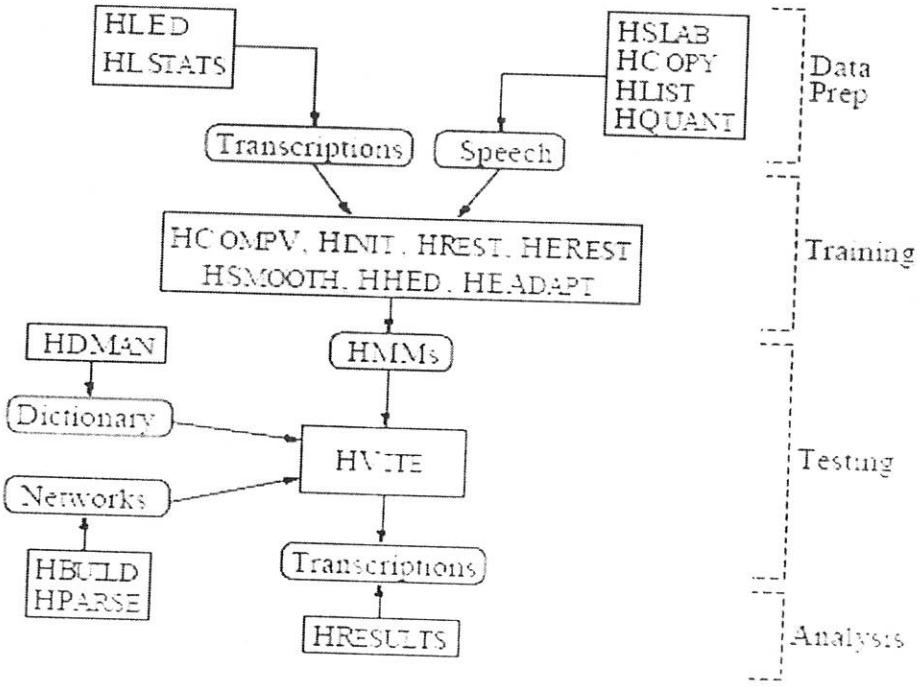

Fig.3. 4 HTK processing stages

CHAPTER 4
EXPERIMENTATION

4.1 Introduction

In order to select the best speaker model to be implemented in the current system, several baseline experiments were performed using the HTK toolkit [20]. Before discussing in depth the experiment that was conducted, a detailed description of the steps taken in each experiment is provided. Where appropriate, graphs are provided as a summary of the findings in the experiment.

For the sake of simplifying the research, so as to fit to the academic requirement, most of the parts of the system have been dropped to experiment on simple data. This chapter describes the basic issues considered during conducting the experiment.

4.2 The Experiment

4.2.1 Data Preparation

In any recognizer, data preparation is the first step. In speaker verification system development, speech data is needed both for training and testing. In this research, no already developed database is used and thus all of the speech data is recorded from scratch. All individuals are given the same word to utter, so as the excitation parameters of all the individuals under this same environment are almost similar, but vary considerably in parameters derived from vocal tract responses.

In this research which is an experiment on text dependent speaker verification, it is required to let the user select the password for him/her, where each speaker is modeled by a set of HMMs describing all speech units in the speakers vocabulary. But for simplicity's purpose every speaker is modeled by single word selected from the Amharic language 'ixnnegagerixbetalen'. The word is selected on the belief by the researcher that the word is long enough to help in extracting ample parameters from a person's speech that is unique to an individual.

In this experiment phone based HMMs are defined to build word model of an individual. This is considered because of the possible extensions of the research to multiple words for which phone based HMMs are suitable [6]. HTK training and recognition tools are mostly suitable for speech recognition purposes as indicated in the last chapter. Whilst, the recognition tool, HVite, produces a log likely hood together with the word which is recognized. In such experiment, which uses single for training purpose, it is trivial that the recognition tool will exactly recognize the word, but with varying log probabilities, which is a point of interest in HTK based speaker verification system. Thus to make use of the HTK tools, the following procedure is used:

4.2.2 Constructing the Task Grammar

The goal of developing task grammar here is to identify the word to be used in the experiment and to put it in a desired format for successive uses. The word 'ixnnegagerixbetalen' is used in this sense.

HTK provides grammar definition language for specifying simple task grammars. For single word, the case is rather very simple. A suitable grammar is thus:

(ixnnegagerixbetalen) put in parenthesis.

Even if we have only one word here, HTK tools expect a word network which is a Standard Lattice Format (SLF). For this purpose, the tool HParse is used to convert the grammar above into a word network (in our case a network of one node). Storing the above grammar in a file called gram, HParse can be invoked as:

HParse gram wdnet

To take the grammar as input and output the word network as file wdnet, which is an SLF.

4.2.3 The Dictionary

Building dictionary is another necessary step to be exercised to use HTK. For the problem at hand, the task of constructing dictionary is very simple. But in the case where the individual must be allowed to choose his/her own password, a large vocabulary sentence database must be used.

A sorted list of words must be constructed from the database, and stored in a file. A pronunciation dictionary for the words in the database must be obtained from standard source if available, or have to be constructed. In this experiment the list of words is stored in a file wordlist, containing only the word:

"*ixnnegagerixbetalen*" and

the pronunciation dictionary is wordpron_dict, containing the phone level pronunciation of the word as:

ixnnegagerixbetalen *ix n n e g a g e r ix b e t a l e n*

is used.

Then the HTK tool HDman is invoked as follows:

HDman –m –w wordlist –n monophones –l dlog dict wordpron_dict

To produce a dictionary called dict (a word with its pronunciation) by taking the word list and the corresponding pronunciation dictionary wordpron_dict. In our case the wlist is a file containing the word itself as described above. The –m option is used to merge all pronunciations from source dictionaries, which in our case is only one. –l option instructs HDman to output a log file dlog which contains various statistics about the constructed dictionary, for instance, whether or not the dictionary is constructed and about the missing words.

HDman also outputs list of phones used here called monophones. The ultimate goal of this step is to generate all those distinct phones for which HMMs are defined latter [21].

As the general format of the dictionary is:

Word [outputsym] p1 p2 p3 . . ., the output of HDman in this experiment in the file dict is

ixnnegagerixbetalen *ix n n e g a g e r ix b t a l e n*

just similar to wlist.

At this point it must be noted that no silence is entertained. since silence must be removed from the speech for use in the speaker verification system. Both the ***dict*** and ***monophones*** can be constructed manually. and this step is considered only for the sake of completeness.

The distinct phones output in ***monophones*** are:

ix n e g a r b t l

considered based on the works illustrated in [22], [23], [24], which uses 12 MFCC in their researches and found comparable results. This shows that even if some of the speaker variations might be suppressed while using only the first 12 MFCC coefficients, the remaining variations will still work in carrying speaker variations.

The output should be saved in compressed format, indicated by setting ***savecompressed*** to **T** and a **CRC** checksum should be added. The FFT should use a Hamming window and the signal should have first order preemphasis applied using a coefficient of 0.97, as in [23]. The variable ENORMALISE is by default true and performs energy normalization on recorded audio files. It cannot be used with live audio and since the target system can be used for live audio, this variable should be set to false.

Rather than running HCopy several times for each wav form, a script file (wav_to_mfc.scp) that contains list of all pairs of conversions is used. The script file is given in the Appendix **A.**

HCopy is invoked from the command line as

C:\>HCopy –C Config –S wav_to_mfc.scp

Based on the configuration parameters, the HCopy extract the needed parameters from the speech wave forms and copies them to a new file with the same name but different extension, **.mfc**.

This time, all ***.wav*** formats are now converted to ***.mfc***.

4.2.5 System Training

If we have a labeled speech data and thus the location of the sub-word units (in this case phone) boundaries are marked, this can be used as a bootstrap data. In the case where there

is no bootstrap data, a scheme called flat start is used, which makes all the models equal initially and move to embedded training using HEREST.

The first step in HMM training is to define a prototype model. Different parameter values are listed but the values of this model are not important as they will be set at latter time; thus the purpose of this definition is to describe the model topology.

As illustrated above, the feature vector used in this experiment is a 12 MFCC with one short-time energy and its delta and delta-delta (Acceleration) coeficients forming a total of 39 dimensions (vector size).

A good topology for phone-based system is a 3-state left-to-right [18]. In HTK based HMMS, since entry and exit states are non-emitting, they have no output probability distributions associated with them. Due to this fact, in this experiment a sub-word based HMM, with five states per phoneme is used; and thus the topology to be used here is a five state left-to-right HMM. The topologies defined for the phone-base by the name proto is attached in the Appendix B.

During the first cycle of embedded re-estimation, the word utterance will be uniformly segmented since it has enough of the sub-word models align with actual realizations of units so that on the second and successive iterations, the models align as intended.

Embedded re-estimation is started for the set of monophones, and every mean and variance is made identical. The HTK tool HCOMPV is used to compute the global mean and variance of the training set. The command

HCOMPV – C config –f 0.01 –m –S nametr.scp –M hmm0 proto

Creates a new version of proto in the directory hmm0 for the phone models in which the zero means and unit variances are replaced by the global means and variances. The – f option causes a variance floor macro (called vfloors) to be generated which is equal to 0.01 times the global variance. This is a vector of values which will be used to set a floor on the variances estimated in the subsequent steps. The – m option tells the HCOMPV to calculate the means and variances.

Having this new prototype model stored in directory hmm0, a HMM definition file *namehmmdefs* containing a copy for each of the required monophone HMMs was constructed by manually copying the newly generated prototype and re-labeling each HMM definition for each required monophone or monosyllable.

The above process is repeated for each speaker, and now we have namehmmdefs for each speaker in the directory hmm0.

The flat-start sub-word units for each speaker that are stored in the directory hmm0 are next re-estimated using the embedded re-estimation tool HEREST is invoked as:

HEREST – C Config –T 1 – I phones.mlf – S nametr.scp – H hmm0/namehmmdefs – M hmm1 monophones.

Embedded training using HEREST simultaneously updates all of the HMMs in a system using all of the training data. Each time HEREST is run, it performs a single Baum-Welch re-estimation of the whole set of HMM phone models. For each word model training, the corresponding sub-word units are concatenated and then the forward-backward algorithm is

used to accumulate the statistics of state occupation, means, variance, etc., for each HMM in sequence.

HEREST was run twice more, where after each run each new HMM set is stored in a new directory, changing the name of the input and output directories (set with the options –H and -M) each time till the directory hmm15 contains the final set of phone HMMs. 15 is considered here since convergence is achieved on average at the 15^{th} iteration.

4.2.6 Creating Background (Anti-Speaker) Model

As noted in the previous chapter, speaker verification does not only require a model describing who the speaker is, but also a model describing all other or competitive speakers. The model for anti-speakers can be either a "world model", a single model trained on many speakers, or a "cohort model" where one have several models of individual speakers having closer model parameters and calculate the sum of log-likelihood over a few of the models that best match the enrollment data from the target speaker.

Because of some advantages that can be drawn from the world model - efficiency in computation and less storage space, and the fact that there is no definite difference in performance between the two techniques in text-dependent speaker verification the world model approach is the commonly adopted one, and in this research the world model is used [7]. A set to develop the world model is taken such that one utterance from each client speaker, a total of nine MFC files and another nine individuals completely away from the client set, are considered based on convenience to construct the world model. The

corresponding label files are constructed in the same way as that of each client individual models. Similar mechanism is used to create the worldhmmdefs, and stored in the subsequent hmm folders.

This completes the training phase.

Table 4.1 shows the HTK tools used in the experiment together with their functions.

HParse	The HParse program generates word level lattice files (for use with HVite), from a grammar file.
HLEd	This program is a simple editor for manipulating label files. We use it to make word (or phone) level transcriptions of training data.
HSlab	This program is used to record speech of individuals, both for training and testing.
HCopy	This program is used for generation of MFCC (Mel Frequency Cepstral Coefficients), which are required by HTK for model building, from recorded WAV files.
HCompV	This Program calculates the global mean and covariance of a set of training data. We use this tool to create initial (HMM)values from the training data.
HERest	This program is used to perform a single re-estimation of the parameters of a set of HMMs using an embedded training version of the Baum-Welch algorithm.
HVite	HVite is a general-purpose Viterbi word recognizer. HVite is used to calculate log-likelihoods when testing.

Table4. 1 Summary of HTK tools used in the experiment

4.2.7 Recognizer Evaluation

HVITE is the only tool provided by HTK for recognizing speech. In this research, HVITE is used mainly to generate the log likelihood that an observation is generated by the speaker's word model. HVITE, run for a client called Abraham is given as:

HVITE –H hmm15/abrahamhmmdefs –S abrahamtest.scp –i recout.mlf –w wdnet dict monophones

Abrahamtest.scp is a script file containing a test file for the person by the name abrahamtest.mfc. The execution of this produces an mlf file on the current directory by the name recout. The content of recout is:

```
#!MLF!#
"abrahamtest.rec"
0 16300000 ixnnegagerixbetalen -10469.789063
.
```

The first two columns of the above output MLF file are to do with time alignment. The third column is the recognized word. It is the fourth column that is of interest in HTK based speaker verification. This column contains the log likelihood values for a word spoken which can be used as an input to a decision program.

This value is the log likelihood which can directly be used for comparison purposes. The log likelihood values are large negative numbers, which is a result of multiplying many factors < 1, i.e. it is the result of multiplying number of observations vectors * vector dimension * likelihood from a gaussian * a transition probability. This value can be normalized by using the –o N option, which will bring the value in a range of -50 to -100 or

so. This, however may round off some important digits to one, obscuring differences. For this very reason, in this research the log likelihood obtained from HVite without normalization is used.

The world model, discussed in the last section, is now to be used practically. The logic of decision lies behind the fact that given an observation, it will be evaluated against the model of the claimed person and model of the anti-speaker (the world model). Decision is made in such a way that if the value for the model is larger than that of the world, the person will be accepted, and otherwise will be rejected.

4.2.8 Discussion of the results

To take care of the comparison and produce the parameters live, a small program is written in C++, and stored by name INTERFAC.EXE. The code of the program is attached in *Appendix C.*

The program is written based on the flow depicted by fig.4.3.

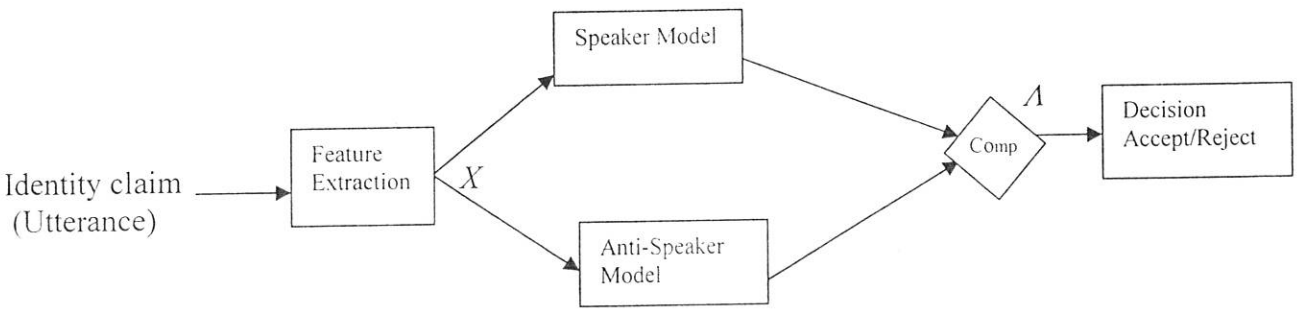

Fig.4. 3 Verification phase (Testing) Likelihood Ratio Test (LRT)

X is a set of MFCC feature vectors extracted from the utterance.

The probability that X is from the claimed speaker is given by P (λc | X) and,

X is not from the claimed speaker is given by P ($\lambda c'$| X)

Upon execution, the program first changes directory to the root directory, and load the HTK batch file to the root directory (C:\>), which will help to execute the HTK tools that will be needed latter.

The program also reads in wave files, writes the *hcopy –C config* together with the source and target file names to the command line, where the input is in wave format and the output is in an .mfc format, and writes the output to the current directory. The name of the newly created mfc file will be written to a script file. This will latter be submitted to the HVite for recognition purpose. The recognition results of the observation by the model of the person and the world model will be written to two separate files named 'model.mlf' and 'world.mlf'.

The major part of the program reads in the log likelihoods of each recognition as mlp and wlp for model and world recognition outputs respectively. These two values are compared for decision. If ***mlp*** is greater than ***wlp***, the person is accepted, and otherwise rejected.

4.2.8.1 Analysis of the Results

The system is tested by using test data both from the client and non-client sets. Each individual is tested against his/her own ten test data, and nine test utterances from individuals away from the training set. Two cross-testing was also made for each client

APPENDICES

Appendix A: Wave to MFCC Script file

abraham_0.wav abraham_0.mfc

abraham_1.wav abraham_1.mfc

abraham_2.wav abraham_2.mfc

abraham_3.wav abraham_3.mfc

abraham_4.wav abraham_4.mfc

abraham_5.wav abraham_5.mfc

fantahun_0.wav fantahun_0.mfc

fantahun_1.wav fantahun_1.mfc

fantahun_2.wav fantahun_2.mfc

fantahun_3.wav fantahun_3.mfc

fantahun_4.wav fantahun_4.mfc

fantahun_5.wav fantahun_5.mfc

fekadu_0.wav fekadu_0.mfc

fekadu_1.wav fekadu_1.mfc

fekadu_2.wav fekadu_2.mfc

fekadu_3.wav fekadu_3.mfc

fekadu_4.wav fekadu_4.mfc

fekadu_5.wav fekadu_5.mfc

tesfaye_0.wav tesfaye_0.mfc

tesfaye_1.wav tesfaye_1.mfc

tesfaye_2.wav tesfaye_2.mfc

tesfaye_3.wav tesfaye_3.mfc

tesfaye_4.wav tesfaye_4.mfc

tesfaye_5.wav tesfaye_5.mfc

wondisho_0.wav wondisho_0.mfc

wondisho_1.wav wondisho_1.mfc

wondisho_2.wav wondisho_2.mfc

wondisho_3.wav wondisho_3.mfc

wondisho_4.wav wondisho_4.mfc

wondisho_5.wav wondisho_5.mfc

yehenew_0.wav yehenew_0.mfc

yehenew_1.wav yehenew_1.mfc

yehenew_2.wav yehenew_2.mfc

yehenew_3.wav yehenew_3.mfc

yehenew_4.wav yehenew_4.mfc

yehenew_5.wav yehenew_5.mfc

Appendix B: Prototype definition for the phones

~h"proto"
<BeginHMM>
<VecSize> 39 <MFCC_0_D_A>
<NumStates> 5
<State> 2
<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 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 1.0
<State> 3
<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 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 1.0
<State> 4
<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 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 1.0
<Transp> 5
0.0 1.0 0.0 0.0 0.0
0.0 0.5 0.3 0.2 0.0
0.0 0.0 0.5 0.3 0.2
0.0 0.0 0.0 0.6 0.4
0.0 0.0 0.0 0.0 0.0
<ENDHMM>

Appendix C: C++ Code for the Interfac.exe

```
#include<iostream.h>
#include<process.h>
#include<string.h>
#include<stdlib.h>
#include<fstream.h>
#include<stdio.h>
int main()
{
        char name[30];
        char inputWave[50];
        char command[200];
        cout<<"Enter the name of the expected person ===>";
        cin>>name;

        cout<<"Enter the file name of the wave file ===>";
        cin>>inputWave;

        strcpy(command, "cd\\");
       system(command);

        cout<<command<<endl;

        strcpy(command, "htk");
        system(command);

        cout<<command<<endl;

        strcpy(command, "hcopy -C config ");
        strcat (command, inputWave);
        strcat(command, " c:\\wave.mfc");
        system(command);
        cout<<command<<endl;

        ofstream outf;
        outf.open("c:\\mfcfilelist.scp");
        if(outf.fail())
        {
                cout<<"error in creating the file mfcfilelist.scp"<<endl;
                exit(1);
        }
       outf<<"c:\\wave.mfc"<<endl;
       outf.close();
```

```
        char modelName[100];
        strcpy(modelName, name);
        strcat(modelName, "hmmdefs");

        strcpy(command, "hvite -H hmm3/worldhmmdefs -S mfcfilelist.scp -i
        world.mlf -w wdnet dict monophones");
        system(command);

        strcpy(command, "hvite -H hmm3/");
        strcat(command, modelName);
        strcat(command, " -S mfcfilelist.scp -i model.mlf -w wdnet dict
        monophones");
        system(command);

        ifstream inpf1, inpf2;

        inpf1.open("world.mlf");
        inpf2.open("model.mlf");

        char token[40];
        double wlp, mlp;
        inpf1.getline(token, 40);
        inpf1.getline(token, 40);
        inpf1>>token;
        inpf1>>token;
        inpf1>>token;
        inpf1>>wlp;

        inpf2.getline(token, 40);
        inpf2.getline(token, 40);
        inpf2>>token;
        inpf2>>token;
        inpf2>>token;
        inpf2>>mlp;

        if (wlp <= mlp)
                cout<<"person accepted"<<endl;
        else
                cout<<"not accepted"<<endl;
return 0;
}
```

21. Young, S. et al. (2003), The HTK Book. Cambridge University Engineering Department.

22. Thian, N. et al: Spectral Sub-band Centroids as Complementary Features for Speaker Authentication. http://www.idiap.ch/~bengio/cv/publications/pdf/poh_2004_icba.pdf (pp 5)

23. Li, S. et al: Learning to Boost GMM Based Speaker Verification. http://research.microsoft.com/~szli/papers/SpeakerID.pdf (pp 2)

24. Orteg, J. et al: Facing severe channel variability in forensic Speaker verification conditions. http://www.atvs.diac.upm.es/publicaciones/docs/Ort99b.pdf (pp 2)

25. Speaker Verification Technology in the CAVE project: final version. www.ptt-telecom.nl/cave/download/P_D42.pdf (pp 7)

Declaration

I hereby declare that this thesis is of my own composition, and that it contains no material previously submitted for the award of any other degree at any University. The work reported in this thesis has been executed by myself, except where due acknowledgement is made in the text.

Fantahun Aberra

June 2004

The thesis has been submitted for examination with my approval as university advisor.

Dr. Eniyew Adugna (PhD)

June 2004