ADDIS ABABA UNIVERSITY

FACULTY OF INFORMATICS

DEPARTMENT OF INFORMATION SCIENCE

SPEAKER DEPENDENT SPEECH RECOGNITION FOR WOLAYTTA LANGUAGE

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

By:

Habtamu Fanta Alambo

July, 2010

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY

FACULTY OF INFORMATICS

DEPARTMENT OF INFORMATION SCIENCE

SPEAKER DEPENDENT SPEECH RECOGNITION FOR WOLAYTTA LANGUAGE

By:

Habtamu Fanta Alambo

Signature of Board of Examiners for Approval

Name	**Signature**
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Acknowledgment

All my endeavours would have been in vain had I been without The Mighty God. Thank you Lord Jesus!

I would also like to acknowledge my Advisor Dr. Dereje Teferi for his critiques, guidance, and support throughout the thesis work; without whom this paper wouldn't have been lively.

My special thanks also goes to Hawassa College of Teachers' Education (HCTE) staff members and students for assisting me during the data collection phase of the research.

It gives me great pleasure and honour to thank my family (Mom, Dad, and Bro) for always being my role model and supporting me with their prayers since childhood, and my fiancé Tigist Dereju for being by my side in encouraging me till this finishing line.

Finally, I would like to thank my classmates for everything that we had throughout the two years spell, and everybody whose names I couldn't list down here, who had been so considerate and praying for me throughout my stay here.

Acronyms

AI – Artificial Intelligence

ANN – Artificial Neural Network

AP – Acoustic Parametre

ASR – Automatic Speech Recognition

CMU – Carnegie Mellon University

CV – Vowel preceded by a Consonant

DFT – Discrete Fourier Transform

EM – Expectation Maximization

FFT – Fast Fourier Transform

GB – Giga Byte

HMM – Hidden Markov Model

HTK – Hidden Markov model Toolkit

LPC – Linear Predictive Coding

MFCC – Mel-Frequency Cepstral Coefficients

PERL – Practical Extraction and Reporting Language

PLP – Perceptual Linear Prediction

RAM – Random Access Memory

VC – Consonant preceded by a Vowel

Table of Contents

List of Tables:

List of Figures:

Abstract

Speech recognition systems are built and researches are conducted world wide to meet the needs of different languages. In Ethiopia, speech recognition researches were carried out for languages like Amharic, Oromifa, and Tigrigna. These works have paved the way for researches on other languages like Wolaytta.

A speaker dependent approach to speech recognition was proposed and adopted for the Wolaytta language in this research. Word level context independent and word level context dependent models and recognizers were built based on an HMM based open source speech recognition tool known as Sphinx.

A speech corpus was constructed from utterances made by using 450 distinct words which were selected upon domain expert consultation, and were each recorded in a single audio file to make a total of 450 audio files. Two independent tests were carried out on this corpus. The first test involved randomly selecting 100 words from the training corpus (which consisted of 300 words) and testing the model using these set of words. The second test involved using 150 words that were not used in training the model and testing the model with these set of words. After the data preparation, preprocessing activities mandatory for model building were carried out such as preparing dictionary and phone set for Wolaytta, and extracting the feature vectors of the audio files used in training.

Tests were carried out on the test data for both models, and an accuracy of 53% was achieved for the context-dependent model, while accuracy of 41% was retrieved for the context-independent

model. This research work can further be extended and the results can be enhanced as per the recommendations made by the researcher.

Key terms: Context-dependent, Context-independent, HMM, Speaker dependent, Speech Recognition, Sphinx, Wolaytta

CHAPTER ONE

INTRODUCTION

1.1. Background

With the growth of Information Technology and its different features being observed in different arenas, one of its main incorporations (i.e. communication) has become the area mostly exploited in Information Communications Technology (ICT). People communicate through different media, and in different ways. One of these universal and ultimate ways of human communication is speech. It has also become the way through which human beings communicate with machines (or computers). Speech interface in user's own language is an ideal means of communication which is natural, flexible, efficient and convenient, and allows hands and eyes to be free (Nahm and Slater, 1997).

Zegaye (2003) states that there are at least two main areas of research in relation to man-machine communication through voice: speech synthesis (voice output) and speech recognition (voice input). Speech synthesis is a technology used to generate voice output from a computer. It converts a text to speech and read for the user (Zue and Cole, 1995). Speech recognition, on the other hand, refers to the ability to take speech as input and produce a transcript of that speech as output.

Technically, speech recognition is the conversion of acoustic signal to stream of text words (Rodman, 1999). It was one of the challenging fields that scientists had faced for long time. This is mainly due to the interdisciplinary nature of the field. It requires speech perception to be implied in the recognizer which in turn brings the use of intelligent systems to the scene (Waleed

and Nikola, 1999). However, over the past several years, speech technology in general and speech recognition in particular has gained considerable advancements (Dutoit 1995, and Holmes 2003). A variant of speech recognition, Automatic Speech Recognition (ASR), is the process by which a machine identifies speech, and its components are shown in figure 1.1. The identification may involve taking a human utterance as input and returning a string of words, phrases, or continuous speech in the form of text as output. The conventional methods of speech recognition insist in representing each word by its feature vector and pattern matching with the statistically available vectors using Hidden Markov Model (HMM) and Artificial Neural Network (ANN) (Maheswari et al., 2009).

Hidden Markov Models (HMMs) provide a simple and effective framework for modeling time-varying spectral vector sequences (Mark and Steve, 2008). A Hidden Markov Model provides a model of a system where data is generated according to a number of underlying processes. Which particular underlying process is active at a given time is not known, so the process responsible for generating a particular observation is not known.

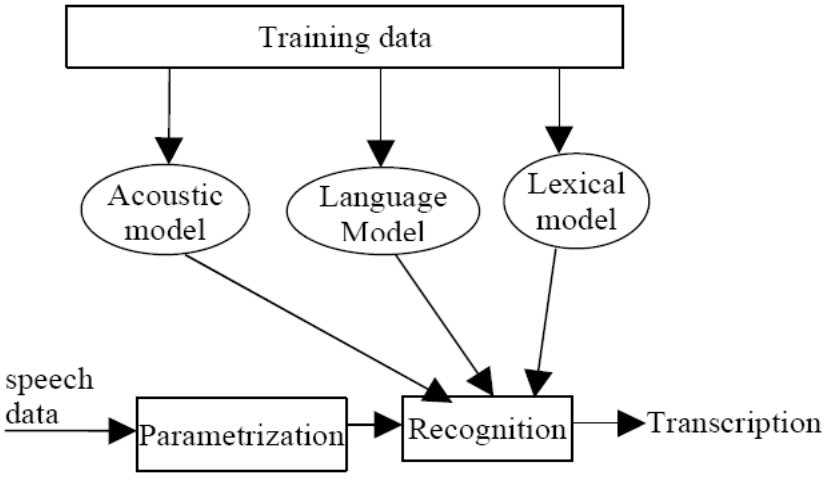

Figure 1.1: Components of an ASR System (Adapted from: Zue et al., 1996)

Rabiner and Juang (1993) stated that speech recognition is a rich field for both practical and intellectual reasons. Practically, it solves problems, improves productivity, and changes the way we run our lives. Intellectually, it holds considerable promise as well as challenges in the years to come for scientists and product developers alike.

Zue (1987) commented that, the long term goal of research in speech recognition is to develop a habitable environment where humans can communicate orally with machines.

Speech recognition systems can be categorized as follows based on their constraints:

Speaker Dependent vs. Speaker Independent Speech Recognition System:

A *speaker dependent system* is developed to operate for speaker/speakers used in training. It uses speech from the target speaker to learn the model parametres of the speakers' voice through training (Philip and Young, 1987).

Speaker independent systems, on the other hand, are capable of recognizing speech from a new speaker that may not have been included in the training. They are designed to be used by anyone, with no enrollment (Markowitz, 1996).

Speaker independent speech recognition is important for the successful development of speech recognizers in most real world applications. While speaker dependent speech recognizers have achieved close to 100% accuracy, speaker independent speech recognition systems have accuracy not exceeding 75% (Maheswari et al., 2009).

Discrete vs. Continuous Speech Recognition System:

Discrete or *isolated speech recognition systems* are systems that require the speaker to pause briefly between words (Zegaye, 2003). These systems process a single word at a time.

Continuous speech recognition systems operate on speech in which words are not separated by pauses. Speech is said to be continuous when it is uttered as a continuous flow of sounds with no inherent separations between them (Markowitz, 1996).

Studies in speech recognition can be conducted on Small, Medium or Large Vocabularies. Although there are variations among literature, generally, small vocabulary recognition systems are those which have a vocabulary size of 1 to 99 whereas medium and large vocabulary systems have a vocabulary size of 100 to 999, and 1000 or more words respectively (Solomon et al., 2008).

Several studies have been carried out on speech recognition systems and the roles they play in different areas. Some of the potential applications include command and control, dictation, transcription of recorded speech, searching audio documents and interactive spoken dialogues (Mark and Steve, 2008). Different models have been deployed to build such systems; the commonly used ones being Hidden Markov Model (HMM) and Artificial Neural Network (ANN). HMMs lie at the heart of virtually all modern speech recognition systems. Though the basic HMM framework has not changed significantly, the detailed modeling techniques developed have evolved to a state of considerable sophistication (Ibid.).

Reflecting the computational power, initial development in the 1980's focused on either discrete word speaker dependent large vocabulary systems (Jelinek, 1985) or whole word, small vocabulary, speaker independent applications (Rabiner et al., 1985).

Mohamad et al. (2008) implemented a hybrid approach to speech recognition that applies both Artificial Neural Networks (ANNs) and Hidden Markov Models (HMMs) for Arabic language.

This new, hybrid method gave comparable results to the already implemented HMM method for the recognition of words and it has overcome HMM in the recognition of sentences. The same approach was also shown to be successful in improving the performance of speech recognition systems (Waleed and Nikola, 1999). In Maheswari et al. (2009), a bottom-up approach to speech recognition was adopted in which a speech signal is resolved into a string of phonemes. Phoneme[1] recognition forms the basis for word recognition which in turn constitutes the full text output.

Research in automatic speech recognition (ASR) began in the late 1940s and early 1950s concurrently in Europe with Dreyfus-Graf who developed the first "Phonetographe" and in U.S.A with K-H Davis who built the first speaker dependent isolated digit recognizer. These systems were electronic devices. Research in computer-based ASR was started in the late 1950s and early 1960s. In general, the 1960s were the time for small vocabulary, isolated word recognition systems developed based on simple acoustic-phonetic properties of speech sound (Solomon et al., 2008). During these times, speech recognition systems tried to model the human articulatory channel.

The 1970s were the time for medium vocabulary recognition system built using simple template-based, pattern recognition methods. The first speech recognition commercial company, which developed VIP-100 system, was also founded during this time (Ibid.).

In 1980s researchers started to tackle large vocabulary speaker independent, continuous speech recognition problems based on statistical methods using technologies such as Hidden Markov

[1] A phoneme is the smallest, meaningful, and distinguishable element in a language's phonology (Maheswari et al., 2009).

Models (HMM). Speaker independence has been possible because of HMM's ability of capturing and modeling speech variability (Ibid.).

In 1990s it was possible to build large vocabulary systems with language models and task syntax for continuous speech recognition and understanding. As a result, researchers developed an increasing interest for speech processing under noisy or adverse conditions, as well as spontaneous speech recognition and dialog management (Ibid.).

In the first half of 2000s, the attention of researchers has been attracted towards spontaneous, robust speech recognition. Although accuracy of more than 95% can be achieved using read speech, the accuracy drastically decreased for spontaneous speech.

As it can be seen form the above discussion, speech recognition technology has gone through significant progress. The use of Hidden Markov Model, the development of large speech corpora for system development training and testing, establishment of standards for performance evaluation, and advances in computer technology are factors that fueled the progress (Ibid.).

At present, several very large vocabulary dictation systems are available for document generation; Dragon being an example (Ibid.). Although lots of work have been conducted in the area of speech recognition, there still remains a lot to be done for several languages.

1.2. Statement of the Problem

Ethiopia is a nation of over 80 million people where more than 80 languages are spoken. These languages fall in one of the six language families (Semetic, Cushitic, Omotic, Berber, Chadic, and Egyptian) (Motomichi, 2008). One of the languages in the Omotic family is the Wolaytta

language which is spoken in the Wolaita Zone and some parts of the Southern Nations, Nationalities, and People's Region of Ethiopia (Abebe, 2002). In this paper the name '*Wolaytta*' is used interchangeably with '*Wolaytta language*' or '*The Wolaytta language*' to address the name given to the target language.

The Wolaytta language consists of 29 consonant phonemes, including voiced glottalized consonants, which have been analyzed as consonant clusters, and 5 vowel phonemes, which can be combined to form long vowels and diphthongs (sounds formed by combining two vowels in a single syllable) (Motomichi, 2008).

The number of speakers of the Wolaytta language was estimated at 2,000,000 (1991, UBS). However, as stated by Lemma (2003), and according to the office of population and housing census commission of Ethiopia, this number grew to more than 3,300,000 in the year 1999. There are also conflicting claims about how widely Wolaytta is spoken. The Ethnologue identifies one smaller dialect region: Zala. Some hold that Melo, Oyda, and Gamo-Gofa-Dawro are also dialects (Abebe, 2002).

Speech recognition systems have been built for specific languages, and until recently limited themselves to the developed nations. In Ethiopia, attempts were made for developing speech recognition systems for local languages such Amharic (Zegaye, 2003), Oromifa (Ashenafi, 2009), and Tigrigna (Hafte, 2009). It is, therefore, necessary that similar efforts be made on other local languages.

As is the case with most of the Ethiopian regions (zones), the Wolaita Zone offers primary and secondary education in the Wolaytta language (Lemma, 2003). There is also a school for visually

impaired students in the zone (Tewodros, 2009). Therefore, developing a speech recognition system for the Wolaytta language can enable the visually impaired students in that school or elsewhere in the zone, and the language speakers, to communicate and learn as well as make use of computers.

However, much effort has not been put on the Wolaytta language with regard to the above issue and speech technology. To the best knowledge of the researcher, the work by Tewodros (2009) on speech synthesis is the only to be cited in the area for the same language.

The natural extension of this attempt and what had been suspended to be done in accomplishing these endeavours towards Wolaytta Automatic Speech Recognition is the development and realization of speaker dependent Wolaytta speech recognition.

Hence, this research aimed at experimenting the development of speaker dependent Wolaytta speech recognition system.

1.3. Objectives of the Study

1.3.1. General Objective

The general objective of this thesis work is to develop speaker dependent speech recognition system for Wolaytta language.

1.3.2. Specific Objectives

To achieve the aforementioned general objective, the following specific objectives are targeted:

- Understanding the general morphology and phonology of the Wolaytta language, and identifying features of the language through literature review;
- Reviewing literature on Automatic Speech Recognition Systems and ways of developing one for the Wolaytta language;
- Selecting technologies and tools that are suitable for designing speaker dependent speech recognition system for Wolaytta language;
- Developing a prototype for speaker dependent Wolaytta speech recognizer using Hidden Markov Model (HMM);
- Evaluating the performance of the developed prototype on a set of selected test dataset;
- Providing conclusions and forwarding recommendations for further research works;

1.4. Scope of the Study

This research mainly aimed to explore the prospect of developing a prototype for word level speaker dependent Wolaytta language speech recognizer. Word level context-dependent and word level context-independent models were built. The dataset used consisted of speech data recorded using 450 Wolaytta words.

1.5. Methodology

The research methodology defines what activities were carried out, how the research process proceeded with, and what measured the success of the research process. The following methods were deployed in the course of the research:

Literature Review

Related literatures were reviewed from books, journals, and the Internet in the course of the research process to explore the principles and theories of the various approaches, techniques, and tools employed. Furthermore, related literatures on the Wolaytta language including those describing the different features (phonology and morphology) of the language were also reviewed.

Data Collection

Appropriate data for both training and testing was collected and the dataset was prepared. Speech samples were taken from a speaker, which were then used to prepare the speech corpus. The research work focused on different words spoken (450 words to be precise), and these utterances were recorded for further processing activities.

Modelling

The recognition system in this work was developed using the Hidden Markov Model (HMM). HMM is a succinct representation of speech events and requires less storage than many other strategies (Lee, 1989), which is of great importance in modeling speech recognition systems. It is the most widely used and most successful modeling technique. Previous researchers; Martha, Kinfe, and Zegaye deployed this model for developing Amharic speech recognizer.

These HMM models were built using the SphinxTrain module and decoded using the Sphinx-4 decoder. Sphinx-4 is a state-of-the-art speech recognition system written entirely in Java and developed by Carnegie Mellon University. Sphinx-4 is becoming an excellent platform for speech research, and is currently the most complete archive of speech

recognition tool. It is also a very flexible system capable of performing many different types of recognition tasks, and produces better recognition results (Elshafei et al., 2008).

Testing and Evaluation

The testing parametre that was used in evaluating the performance of the speech recognition system is the accuracy of the recognition. The performance of the recognizer was tested using a set of test data. A test sample was taken from a speaker with different utterances (450 utterances), and the accuracy of recognition system was used to evaluate the system.

1.6. Significance of the Study

Speech recognition technology has a wide range of applications from which several advantages can be drawn. The raison d'etre for working on such research emanates from the benefits that can be gained from such systems in general and from developing the same for the Wolaytta language in particular, which include the following:

- Such systems may initiate software developers to develop application software in the target language that assist visually-impaired persons in using a computer;
- Such systems can encourage the development of dictation applications that can serve as a major teaching tool in educational institutions;
- Speech recognition systems have the potential of encouraging novices to use computers, and it can simplify difficult tasks that experienced users may face;
- Such systems can encourage constructing an application where physically disabled people can work in places like call centres using their voice, hence contributing a lot to the field of Computer Assisted Instruction (CAI). In addition, properly built CAI systems

can pave the way for anyone to acquire new skills or raise old ones in a lucrative and efficient manner;

- Such systems can enhance human interaction with computing machines by integrating advances in speech recognition using natural language technology, and hence contributing to the growth of the multidisciplinary field of Human Computer Interaction (HCI);

1.7. Limitations of the Study

The main limitation of this study was the absence of Wolaytta speech corpus to be used for speech recognition studies. This research had thus to be extended to preparing this speech corpus.

The time allotted for the completion of the research work has constrained it to the domain of developing word level speaker dependent systems for Wolaytta. Had there been more time, continuous speeches would have been studied that would model real world speech.

1.8. Organization of the Thesis

This thesis is organized in five chapters. Each of these chapters represents and discusses an aspect of the thesis work carried out in each phase.

The first chapter briefs the reader with the fundamental concepts of automatic speech recognition including definitions of basic terms and application arenas of speech recognition technology. Additionally, problem formulation from this research's perspective, objectives of the research, and the methodologies deployed in the research work are also briefly presented.

The second chapter intensively reviews different literatures written on human speech system, basics of speech recognition process, speech recognition methodologies and approaches, Hidden Markov Models, and highlights researches carried out internationally and locally on speech recognition. The chapter is concluded by discussing the Sphinx tool as applied to speech recognition problems.

Chapter three deals with the Wolaytta language; which include the phonology of Wolaytta, its writing system, and the phonetic features and the sound units of Wolaytta.

The fourth chapter deals with the detailed experimentation activities conducted throughout the research process, and discusses the results obtained through the experiment.

The last chapter discusses concluding remarks based on the results obtained in the previous chapter and puts forward recommendations on how the undertaken research can further be extended in the future.

CHAPTER TWO

LITERATURE SURVEY

Introduction

This chapter presents the human speech system, basics of speech recognition with different approaches pursued, Hidden Markov Models, automatic speech recognition system researches carried out on international and local languages, and the Sphinx package.

In recent years, research in speech recognition has resolved many constraints that were left out of the speech recognition cockpit because of their complexities. Developing speech recognition systems for large vocabulary, continuous, and speaker independent scenarios is becoming the area of research in speech recognition. The advent of faster CPUs and mass storage devices with decreasing costs has an invaluable contribution to the applicability and usability of speech recognition technology.

2.1 The Human Speech System

Speech is a natural way of human communication, and computers capable of understanding speech and “speak” with human beings are believed to have an immense contribution to the growth of natural man-machine interfaces (Holmes, 2003; and Daniel and James, 2000).

Speech is generated by the passage of air through various obstructions and routings of the human larynx, throat, mouth, tongue, lips, nose, and other involved organs. It is a series of pressure waves traveling through the air created by the vibrations of larynx, followed by openings or blockages en route to the outside.

Speech can be described as a result of the coordinated action of a number of muscles. The respiratory organs provide the energy needed to produce speech sounds, by forcing an air flow in the trachea and through the vocal cords (Nebiyou, 2005). These are actually composed of two contiguous membranes, whose tension is controlled by the neighbouring muscles. Figure 2.1 shows the human vocal tract organs in detail.

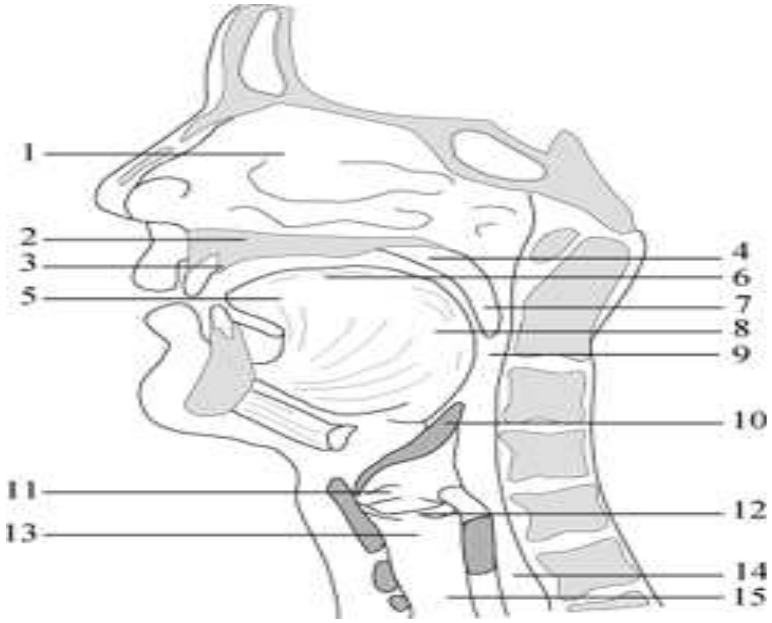

Figure 2.1 Schematic diagram of the human vocal tract (adapted from: Young, 1996)

Labels for the vocal organs: (1) Nasal cavity, (2) Hard palate, (3) Alveolar ridge, (4) Soft palate (Velum), (5) Tip of the tongue (Apex), (6) Dorsum, (7) Uvula, (8) Radix, (9) Pharynx, (10) glottis, (11) False vocal cords, (12) Vocal cord, (13) Larynx, (14) Esophagus, and (15) Trachea.

2.2 Basics of Speech Recognition

The basic principle behind any speech recognition system is that the waveform of a speech signal that comes out of a speaker's vocal organs is a realization of the concept that was stored in the form of symbols in his/her mind (Young et al., 2002). When a speaker generates an idea to produce speech, that idea of speech was already understood symbolically in the mind before

being put into speech signals or sound waves. A speech recognition system performs three primary activities: preprocessing, recognition, and communication (Markowitz, 1996), which are discussed below.

2.2.1 Preprocessing

Speech is a continuous flow of sound waves and silence in analog form. This form of speech cannot be processed directly by a computerized speech recognizer, and hence preprocessing activity is needed to convert the input spoken signal into a form that is understandable by the recognizer. It deals with digitizing analog speech signal and converting the speech wave to some form of parametric representation (Markowitz, 1996). This conversion to parametric representation is done using digital signal processing front ends. Although there are many ways for parametrically representing a speech signal, the most important one is the short time spectral envelope and hence treated as the base of the signal processing front end (Juang and Rabiner, 1993).

The two mostly applied methods of spectral analysis are filter bank and linear predictive coding (LPC) (Markowitz, 1996; Juang and Rabiner, 1993). These methods are shown to yield the best performance in most practical speech recognition systems.

Linear Predictive Coding bases its work on the principle that a speech sample can be approximated as a linear combination of past speech samples. It is one of the most powerful and dominant speech coding methods (Juang and Rabiner, 1993). It provides with reliable, precise, and robust method of estimating parametres that characterize a given speech signal. These merits of LPC have helped it to serve as the basis for coding like cepstral coefficients and vector quantization (Markowitz, 1996).

In contrast, the human ear is capable of resolving audio frequencies non-linearly across a spectrum. Empirical evidences also show that front ends that manipulate non-linear audio frequencies are able to improve recognition performance (Young et al., 2002). However, LPC does not take these issues into consideration as it only assumes linear combination of speech signals. This makes the filter bank a popular substitute to LPC as it entertains non-linearity in speech signals (Ibid.).

2.2.2 Recognition

The activity carried out in this phase is identifying the given speech sample. Three approaches to automatic speech recognition were indicated: acoustic-phonetic, pattern recognition, and artificial intelligence (Juang and Rabiner, 1993).

2.2.2.1 The Acoustic-Phonetic Approach

The acoustic-phonetic approach is based on the theory of acoustic phonetics. This theory states that there exist finite, distinctive phonetic units in spoken language that are broadly characterized by a set of properties manifested in a speech signal (Juang and Rabiner, 1993). It has two steps: segmentation and labeling, and word level recognition.

Segmentation and labeling involves dividing the speech signal into discrete sections and attaching one or more phonetic labels to each divided section on the basis of observed acoustic properties. This step lies at the heart of the acoustic-phonetic approach and is difficult to execute reliably. Because of this difficulty, control strategies are implemented to delimit the segmentation range and labeling possibilities. These control strategies will reduce search space and increase system performance (Ibid.).

However, this step suffers from uncertainty resulted from high degree of acoustic similarity among phonemes and phoneme variability (Markowitz, 1996). This uncertainty in turn results in a set of phoneme hypothesis often organized into a phoneme lattice. This phoneme lattice in turn is provided as an input to the second step of acoustic-phonetic approach, word level recognition, which attempts to find a valid word or string of words from the sequence of phonetic labels (Markowitz, 1996; Juang and Rabiner, 1993). Although the acoustic-phonetic approach is engrossing, it lacks success in practical speech recognition systems due to problems associated with it which include the following among others (Juang and Rabiner, 1993):

- The difficulty of decoding phonetic units into word strings (lexical access problem);
- The difficulty of getting reliable phoneme lattice for the lexical access stage;
- The need for in-depth knowledge of the acoustic properties of phonetic units;
- The choice of features is usually made based on irregular (or informal) considerations;
- The design of sound classifiers is not optimal;

2.2.2.2 The Pattern Recognition Approach

The pattern recognition approach is one in which the speech patterns are used directly without explicit feature determination and segmentation. This method has two steps: training of speech patterns, and recognition of the patterns through pattern comparison. The training phase brings speech knowledge into the system. The idea behind this is that if enough versions of a pattern to be recognized are included in the training set provided to the algorithm, the training algorithm should be able to sufficiently characterize the acoustic properties of the pattern. In the pattern recognition step, the speech to be recognized is compared with each possible pattern learned in the training phase and classified according to the goodness of match of the patterns (Juang and

Rabiner, 1993). If enough patterns are inputted to the speech recognition system in the training phase, it will perform better than the acoustic-phonetic approach. In general, the pattern recognition approach is used more often than the acoustic-phonetic approach because of its simplicity of use, invariance to different speech vocabularies, and accuracy (providing good performance) (Nebiyou, 2005).

The pattern recognition approach can further be categorized into different classes as template matching and stochastic based on factors such as the type of feature measurement, the choice of templates or models for reference patterns, and the process used to generate reference patterns and classify unknown test patterns (Juang and Rabiner, 1993). In stochastic pattern recognition approach, there is no direct matching between the stored model and the input, unlike the template matching approach where there is. It is rather based on complex statistical and probabilistic analysis techniques, which are best understood by studying the network like a structure in which those statistics are stored. The stochastic approach is the most commonly used approach to speech recognition (Markowitz, 1996).

The term stochastic refers to "the process of making a sequence of non-deterministic selections from a set of alternatives" (Markowitz, 1996). Stochastic modeling involves the creation and storage of models for each item that will be recognized. A famous and widely used stochastic model is the Hidden Markov Model (HMM).

The Hidden Markov Model

As discussed in Chapter 1, Automatic Speech Recognition has come of age in practical sense with several speech recognition systems currently in use in applications ranging from a voice dialer for telephone to a voice response system that quotes stock prices on verbal inquiry. What

makes these benefits happen is the recent technological advancement that enables speech recognition systems to respond reliably to non-specific talkers with a reasonably sized recognition vocabulary. One such major advancement is the adoption of statistical methods for modeling speech recognition systems, of which the Hidden Markov Model (HMM) is of particular interest (Juang and Rabiner, 1991).

A Hidden Markov Model also referred to as a Markov source or probabilistic function of Markov chains in communication literature, is defined as a doubly stochastic process based on a statistical model with an underlying stochastic (Markov) process that is not observable (it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observed symbols. The challenge is to determine the hidden parametres from the observable ones (Juang and Rabiner, 1986; Satori et al., 2007; and Ashenafi, 2009).

History of Hidden Markov Models

The basic theory of Hidden Markov Models has been hidden to mathematicians and engineers for almost 100 years, and it is only in the past three or four decades that it has been applied explicitly to speech recognition problems. Basic theories of HMMs were published in sequence of papers by Baum and his colleagues in late 1960s and early 1970s, and implemented for speech recognition and processing applications by Baker, and Jelinek and his colleagues in the 1970s (Juang and Rabiner, 1986; and Rabiner, 1989). There are several reasons why HMMs haven't been applied until recently for speech recognition applications. One of these reasons was the lack of a method for optimizing the parametres of the Markov model to match observed signal patterns. Such a method was proposed in the late 1960's and was applied to speech processing in several research institutions. Secondly, basic HMM theories were used to be published on

mathematical journals which were not often accessed and read by engineers working on speech recognition problems. The third reason was original applications of the HMM theory to speech recognition problems were not sufficiently provided in tutorial or other forms for readers and researchers in the area. Continual fine-tunings in the theory and implementation of Markov modeling techniques have greatly enhanced the method, leading to a wide range of applications of these models (Ibid.).

Basics of Hidden Markov Models

The use of Hidden Markov Models for speech recognition has become predominant in the last several years. The reasons for this predominance are the inherent statistical framework, the ease and availability of training algorithms for estimating the parametres of the models from finite training sets of speech data, the flexibility of the resulting recognition system whose size can easily be changed, type or architecture of the models to suit particular words or sounds, the ease of implementation of the overall recognition system, and its potential low cost and its capability of modeling various events (phonemes and syllables) with efficient parametric representations (Juang and Rabiner, 1991; and Juang et al., 1985). Despite these merits, HMM's assumption that observations corresponding to sequence of speech spectra produced by the underlying HMM process are state dependent only, has been treated as a demerit of HMMs (Dai, 1994).

The most important and successful HMM based speech recognition systems include the Dragon System at Carnegie Mellon University, the longstanding effort of IBM on a voice-detection system, the work at AT&T Bell Laboratories, MIT Lincoln Labs and Philips on whole-word recognition using HMMs, and the DARPA resource management task (Juang and Rabiner, 1991).

Consider a system which can be in one of N distinct states, S_1, S_2, …, S_N at any time. Assuming N=5, we have the following Markov chain with 5 states.

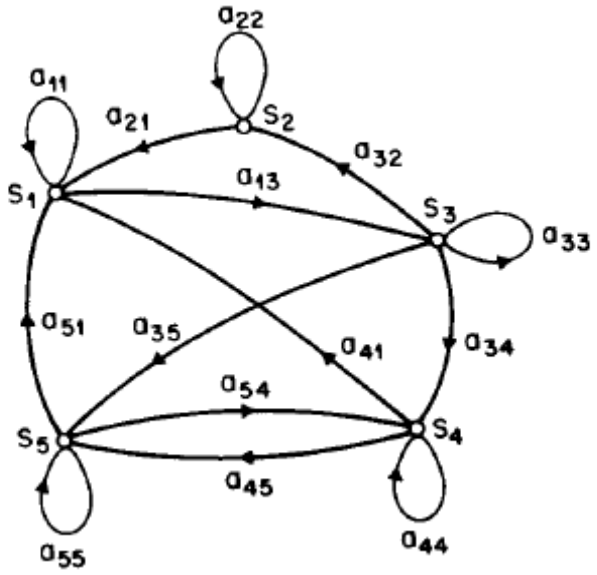

Fig. 2.2 A Markov chain with 5 states S_1 to S_5 with selected state transitions (adapted from: Rabiner, 1989)

At regularly spaced discrete instances, the above system goes through state change (possibly back to a same state) in accordance with a set of probabilities associated with the state. The time instances linked with the state changes are denoted as t = 1, 2…, and the actual state at time t is denoted as q_t.

An HMM is defined by the following elements (Rabiner, 1989):

- N, the number of states in the model. Although the states are not observable, for many practical scenarios there is some physical significance attached to the states. Individual states are labeled as $S = \{S_1, S_2, \ldots, S_N\}$ and the state at time t is denoted by q_t.

- M, the number of distinct observation symbols per state "i.e. the discrete alphabet size". Observation symbols match (map) the physical output of the system being modeled. Individual observation symbols are denoted by $V= \{v_1, v_2,\ldots,v_M\}$.
- The state transition probability distribution $A=\{a_{ij}\}$, where

  $a_{ij} = P(q_{t+1} = S_j / q_t = S_i)$, $1 \le i, j \le N$

  For the unique case where a state can be reached from any other state in a single step, $a_{ij} > 0$ for all i, j. For other HMM types, we would have $a_{ij} = 0$ for one or more (i, j) pairs.
- The observation symbol probability distribution in state j is given by, $B=\{b_j(k)\}$ in which

  $b_j(k) = P(V_k \text{ at } t / q_t = S_j)$, $1 \le k \le M$

  $1 \le j \le N$

  $b_j(k)$ defines the symbol distribution in state i, j=1,2,…,N
- The initial state distribution $\pi = \{\pi_i\}$ in which

  $\pi_i = P(q_1 = S_i)$, $1 \le i \le N$

Since a_{ij} and $b_j(k)$ are probabilities, they satisfy the following probability rules:

$a_{ij} = \ge 0, \forall$ i, j

bj (k) $\ge 0, \forall$ j, k

$\sum aij = 1, \forall$ i

$\sum bj(k) = 1, \forall$ j

If appropriate values are given for the parametres N, M, A, B, and π, the HMM can produce an observation sequence:

$$O = O_1\ O_2\ O_3 \ldots\ O_T$$

Where each observation O_t corresponds to one of the symbols in V, and T is the number of observations in the sequence. The observation sequence can be produced by the following steps (Rabiner, 1989):

a. Select an initial state $q_1 = S_i$ according to the initial distribution π.
b. Assign t to 1.
c. Select $O_t = v_k$ according to the symbol probability distribution in state S_i, i.e., $b_i(k)$.
d. Change to a new state $q_{t+1} = S_j$ according to the state transition probability distribution for state S_i, i.e. a_{ij}.
e. Assign t to t+1; go to (c) if $t < T$, else stop the procedure.

The above procedure set can be used both as a producer of observations, and as a model of how a given observation can be produced by an HMM.

The above discussions show that a complete HMM specification requires specification of two model parametres (N and M), observation symbols' specification, and the three probability measures' (i.e. A, B, and π) specification. A compact representation was used in Rabiner (1989) to show the complete parameter set of the model:

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

Basic Problems of HMMs

Given the HMM form stipulated previously, three basic problems of interest emanate that must be solved for the model to be successful in real world applications (Rabiner, 1989), which are briefly discussed below:

Problem One: Given the sequence of observations $O = O_1\ O_2\ O_3\ \ldots\ O_T$ and the model $\lambda = (A, B, \pi)$, how can we efficiently compute the probability observation sequence $P(O\backslash\ \lambda)$, given the model? This problem is known as the evaluation problem. The problem can also be viewed as one of scoring how well a given model matches a given sequence of observation. If we consider a case where we try to select between several competing models, Problem One's solution lets us choose a model that best matches the sequence of observations.

Problem Two: Given the sequence of observations $O = O_1\ O_2\ O_3\ \ldots\ O_T$ and the model λ, how can we choose a corresponding sequence of states $Q = q_1\ q_2\ q_3\ \ldots\ q_T$ which is optimal and best represents the observations? Here, an attempt is made to unveil the hidden part of the model and find "correct" state sequences, except for the degenerate models. For practical scenarios, an optimality criterion is usually used as best as possible to solve this problem. This optimality criterion depends on the intended use for the unveiled state sequence. Some of these uses include learning about the model structure, finding best state sequences for continuous speech recognition, or finding average statistics of individual states.

Problem Three: How can the model parametres $\lambda = (A, B, \pi)$ be fine-tuned to capitalize on $P(O\backslash\lambda)$? Here, an attempt is made to optimize the model parametres to describe how a sequence of observations is best produced. The observation sequence used to fine-tune the model parametres is known as the training sequence as it is used to "train" the HMM. The training problem is decisive for most HMM applications as it allows to best adapt the model parametres to the observed training data and produce best models for real-world scenarios.

Solutions to the Basic Problems of HMMs

This section presents the prescribed mathematical solutions to the basic problems of HMMs described in the aforementioned section.

- **Solution to Problem One**

Given the model λ, we calculate the probability of the observation sequence $O = O_1\ O_2\ \ldots\ O_T$, $P(O\backslash\lambda)$. The easiest way to do this is to enumerate every possible state sequence of length T (the number of observations). Consider the following fixed state sequence:

$$Q = q_1\ q_2\ \ldots\ q_T$$

where q_1 is the initial state. The probability of the observation sequence for the above state sequence is given by:

$$P(O\backslash Q, \lambda) = \prod_{t=1}^{T} P(O_t\backslash q_t, \lambda)$$

where statistical independence of observations is assumed. We then get

$$P(O\backslash Q, \lambda) = b_{q1}(O_1)\ .\ b_{q2}(O_2)\ \ldots\ b_{qT}(O_T)$$

The probability of the state sequence Q can be shown as:

$$P(O\backslash Q, \lambda) = \pi_{q1} a_{q1q2} a_{q2q3} \ldots a_{qT\text{-}1qT}$$

The probability that O and Q can occur at the same time (or the joint probability of O and Q) is the product of the above two terms, given by:

$$P(O, Q\backslash\lambda) = P(O\backslash Q, \lambda)\ \ P(O, \lambda)$$

The probability of O given λ can be obtained by summing the joint probability over all possible state sequences q yielding:

$$P(O\backslash\lambda) = \sum_{allQ} P(O\backslash Q, \lambda)\, P(Q\backslash\lambda)$$

$$= \sum_{q1,q2,\cdots,qT} \pi_{q1} b_{q1}(O_1)\, a_{q1q2} b_{q2}(O_2) \ldots a_{qT-1} b_{qT}(O_T)$$

Initially (at time t=1), we are in state q_1 with probability π_{q1}, and generate the symbol O_1 with probability $b_{q1}(O_1)$. With clock transition from time t to t+1 (t = 2), we make a transition from state q_1 to q_2 with probability a_{q1q2}, and generate symbol O_2 with probability $b_{q2}(O_2)$. This process continues until the last transition is made (at time T) from state q_{T-1} to q_T with probability $a_{qT-1}q_T$ and generate symbol O_T with probability $b_{qT}(O_T)$.

However, this calculation involves operations in the order of N^T. This is very large even when the length of the sequence T is moderate (Rabiner, 1989). Therefore, efficient algorithms are needed to solve Problem One. The forward-backward algorithm has been used to tackle this problem.

In the forward-backward algorithm, there is a forward variable $\alpha_t(i)$, which is defined by:

$$\alpha_t(i) = P(O_1\, O_2 \ldots O_t, q_t = S_i\backslash\, \lambda)$$

that is, the probability of the partial observation sequence, $O_1\, O_2 \ldots O_t$ up to time t, and state S_i at time t, given the model λ. $\alpha_t(i)$ can inductively be solved as follows:

1) Initialization:

$$\alpha_t(i) = \pi_i b_i(O_1), \qquad 1 \le i \le N$$

2) Induction:

$$\alpha_{t+1}(j) = [\sum_{j=1}^{N} \alpha_t(j) a_{ij}]\, b_j(O_{t+1}), \qquad 1 \le t \le T-1$$

$$1 \le j \le N$$

3) Termination:

$$P(O\backslash\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Step 1 initializes the forward probabilities as the joint probability of state S_i and initial observation O_1. The second step (the induction step) is at the nucleus of the forward calculation, and is illustrated in Figure 2.3

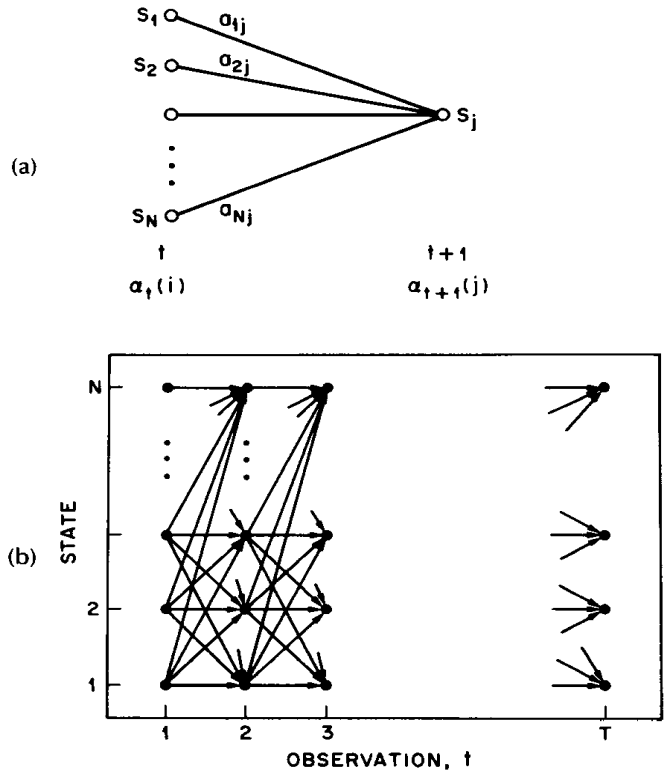

Fig. 2.3 (a) Illustration of the sequence of operations needed for the computation of the forward variable $\alpha_{t+1}(j)$. (b) Implementation of the computation of $\alpha_t(i)$ in terms of a lattice of observations t, and states i (adapted from (Rabiner, 1989)).

The above figure illustrates how state S_j can be reached at time t+1 from the N possible states, S_i, $1 \leq i \leq N$, at time t. Since $\alpha_t(i)$ is the probability of the joint event that $O_1\ O_2 \ldots O_t$ are observed,

and the state at time t is S_i, the product $\alpha_t(i)a_{ij}$ is the probability of the joint event that $O_1\ O_2\ \ldots\ O_t$ are observed, and state S_j is reached at time t+1 through state S_i at time t. Summing up this product over all N possible states S_i, $1 \leq i \leq N$ at time t results in the probability of S_j at time t+1 with all the supporting previous partial observations. Step 3 finally produces the desired calculation of $P(O\backslash\lambda)$ by summing up the terminal forward variables $\alpha_T(i)$. By definition:

$$\alpha_T(i) = P(O_1\ O_2\ \ldots\ O_T,\ q_T = S_i\backslash\ \lambda) \text{ and thus } P(O\backslash\lambda) \text{ is the sum of the } \alpha_T(i)\text{'s}$$

- **Solution to Problem Two**

Finding the "optimal" state sequence associated with a given observation is one of the several possible ways of solving this problem. But it is difficult to define the optimal state sequence as there are optimality criteria. One possible optimality criterion can be: choosing the states q_t that are individually most likely as this maximizes the expected number of correct individual states. Although following these optimality criteria can maximize the expected number of correct states (by selecting the most likely state for each t), some problems may arise from the resulting state sequence. One feasible solution to this problem is to adjust the optimality criterion to find the single best state sequence to maximize $P(Q\backslash O,\ \lambda)$ which is the same as maximizing $P(Q,\ O\backslash\lambda)$. A dynamic programming based technique for finding this single best state sequence is called the Viterbi algorithm (Rabiner, 1989). To find the single best state sequence, $Q = (q_1\ q_2\ \ldots\ q_T)$, for a given observation sequence $O = (O_1\ O_2\ \ldots\ O_T)$, we define:

$$\delta_t(i) = \max_{q1,q2,\cdots,qt-1} P[q_1\ q_2\ \ldots\ q_t = i,\ O_1\ O_2\ \ldots O_t\backslash\ \lambda]$$

where $\delta_t(i)$ is the best score (maximum probability) along a single path, at time t, which accounts for the first t observations and ends in state S_i. By induction we have:

$$\delta_{t+1}(j) = [\max_i \delta_t(i)\ a_{ij}]\ .\ b_j(O_{t+1})$$

To get the state sequence, we need to follow the argument which maximized the previous formula, for each t and j. This is done with an array $\psi_t(j)$. Below is the complete set of steps for finding the best state sequence:

1) Initialization:

$$\delta_1(i) = \pi_i b_i(O_1) \qquad 1 \le i \le N$$

$$\psi_1(i) = 0$$

2) Recursion:

$$\delta_t(j) = \max_{1 \le i \le N} [\, \delta_{t-1}(i)\, a_{ij}]\, b_j(O_t), \qquad 2 \le t \le T$$

$$1 \le j \le N$$

$$\psi_1(j) = \arg\max_{1 \le j \le N} [\, \delta_{t-1}(i)\, a_{ij}], \qquad 2 \le t \le T$$

$$1 \le j \le N$$

3) Termination:

$$P^* = \max_{1 \le i \le N} [\delta_T(i)]$$

$$q^*_T = \arg\max_{1 \le i \le N} [\, \delta_T(i)]$$

4) Path (state sequence) backtracking

$$q_t^* = \psi_{t+1}(q^*_{t+1}),\ t = T-1, T-2, \ldots, 1$$

As it can be seen, the Viterbi algorithm is similar in implementation to the forward calculation except for the 4th step.

- **Solution to Problem Three**

No known way exists to analytically solve for the model parameters that maximizes the probability of the observation sequence in a closed form. Still, it is possible to choose $\lambda = (A, B,$

π) so that its likelihood, P(O\λ) is iteratively locally maximized using the Baum-Welch method (or the expectation-modification (EM)) or gradient techniques.

The Baum-Welch algorithm, also known as the forward-backward algorithm takes the following procedures.

1. Assumes a set of initial transition probabilities.
2. While (the transition probabilities are improving) {
    a. Expectation: Use transition P to estimate states $P(S^t \backslash W^{1:T})$.
    b. Maximization: Estimate new transition probabilities by counting expected number of transitions, given state estimates.}

The "Improvement" condition for looping is measured by comparing the cross-entropy after each iteration using the expression, $w^{1:T}$

$$-\frac{1}{n}\sum_{w} 1:T \;\; P_{M-1}(w^{1:T}) \log_2 P_M(w^{1:T})$$

and terminates when change in cross-entropy is less than some ε value.

Types of HMMs

Hidden Markov Models can be classified into different groups based on different criteria. On the basis of the structure of the transition matrix, HMMs can be classified as Ergodic/fully connected and left-to-right/Bakis (Rabiner, 1989). In Ergodic HMM, every state of the model can be reached from every other state in a finite but aperiodic number of steps. In left-to-right HMM, the state index proportionally increases (or stays the same) as time increases, and backward transition is not possible (Martha, 2003).

According to the nature of the elements of the matrix for the output distribution functions, HMMs can fall into one of the following categories:

- **Discrete HMMs**

In Discrete HMMs, distributions are defined on finite spaces. In this case, observations are vectors of symbols in a finite alphabet of N different elements. For each one of the Q vector components, a discrete density $\{w(k) \mid k = 1,..., N\}$ is defined, and the distribution is obtained by multiplying the probabilities of each component. This definition assumes that the different components are independent (Ashenafi, 2009).

- **Continuous HMMs**

Another possibility is to define distributions as probability densities on continuous observation spaces. In this case, strong restrictions have to be imposed on the functional form of the distributions, in order to have a manageable number of statistical parameters to estimate. The most popular approach is to characterize the model transitions with mixtures of base densities g of a family G having a simple parametric form. The base densities $g \in G$ are usually Gaussian or Laplacian, and can be parameterized by the mean vector and the covariance matrix. HMMs with these kinds of distributions are usually referred to as continuous HMMs (Ashenafi, 2009).

- **Semi Continuous HMMs**

In order to model complex distributions using continuous HMMs, rather large number of base densities has to be used in every mixture. This may require a very large training corpus of data for the estimation of the distribution parameters. Problems arising when the available corpus is not large enough can be alleviated by sharing distributions among transitions of different models.

In semi continuous HMMs all mixtures are expressed in terms of a common set of base densities. Different mixtures are characterized only by different weights. A common generalization of semi continuous modeling consists of interpreting the input vector y as composed of several components $y[1],..., y[Q]$, each of which is associated with a different set of base distributions. The components are assumed to be statistically independent (Ashenafi, 2009), and thus the distributions associated with model transitions are products of the component density functions.

2.2.2.3 The Artificial Intelligence Approach

The third approach to speech recognition is the Artificial Intelligence (AI) approach which hybridizes the acoustic-phonetic and pattern recognition approaches as it exploits the ideas and concepts of both methods. This approach attempts to simulate the way a person applies his/her intelligence in visualizing, analyzing and making a decision on measured acoustic features (Juang and Rabiner, 1993). To do this, the approach requires several knowledge. In fact, the basic idea of AI approach is to "compile and incorporate knowledge from a variety of knowledge sources and to bring it to bear on the problem at hand" (Ibid.). The knowledge needed by AI approach is acoustic, lexical, syntactic, semantic and pragmatic. Some of the techniques used in this approach are the use of expert systems for segmentation and labeling, learning and adapting over time, and the use of artificial neural network (ANN) for learning the relationship between phonetic events and all known inputs as well as for discrimination between similar sound classes (Ibid.).

The use of ANNs can be treated on its own as a separate structural approach or can be regarded as an implementation architecture that may be integrated in acoustic-phonetic, pattern recognition and AI approaches (Juang and Rabiner, 1993).

Neural networks have been shown to yield good performance for small vocabulary scenarios of speech recognition. Sometimes they are better than HMMs for short, isolated speech units. But it remains a challenge for neural networks to demonstrate that they can be as effective as HMMs for dealing with large vocabulary speech recognition (Nebiyou, 2005).

2.2.3 Units of Recognition

Reference models are built at the training phase of a speech recognizer. These models can be based on various units of speech such as phones, word dependent phones, context-dependent phones, multi-phone units, and Words (Martha, 2003). Phrases can also be considered as a possible unit of recognition. The unit of speech used in speech recognition should be trainable, well defined and relatively insensitive to context.

Phones, word-dependent phones, context dependent phones and multi-phone units are called sub word units (Ibid.). As every language has few phones, phone is a trainable unit. However, phones are highly sensitive to context than other units like words and multi-phone units (such as syllables). Since they do not model the co-articulation effect, phone models are also considered as inadequate models for speech recognition, i.e. phone models assume that a phone in any context is equivalent to the same phone in any other context, which may not always hold true in real world circumstances (Ibid.).

Modeling multiple phone units such as syllables is one way of overcoming the aforementioned weaknesses of phone models, and enabling modeling co-articulation effects. However, the large number (although smaller in number than words) of these units affects its trainability (Lee, 1990). Therefore, other recognition units that take context into consideration are proposed, of which word-dependent phones and context-dependent phones or triphones are the ones to look

for. Word-dependent phones try to compromise amongst word modeling and phone modeling units. With word-dependent phones, phone models are considered but they are word-dependent showing that a phone model in one word has dissimilar parameters from the phone model of the same phone in another word. Though word-dependent models are shown to model context, they need large training data and storage (Ibid.).

Triphone or context-dependent phones exhibit some similar features with word-dependent phones. Triphone models are phone models that consider the left and right neighboring phones for modeling. It is also indicated that two phones having the same identity but different left to right context are considered as different (Lee, 1990). Although triphone modeling is a powerful modeling scheme considering its capability of modeling co-articulatory effects and insensitivity to context than phone modeling, it is not trainable as there are many triphones. Furthermore, triphone modeling has two drawbacks. One is memory wastage, and the other is it assumes that every triphone context is different, but in reality many phones have similar effect on their neighbours (Ibid.). It is thus a good practice to consider all these sub word units to achieve a large vocabulary speech recognition system.

As human beings want to recognize words and communicate through them, words are treated as the most natural unit of speech (Lee, 1990). Models built based on words yield best performance results given sufficient training data. In word models, training data can not be shared among the different words; hence each word has to be trained individually. This means that large examples of each word are needed for adequate training (Martha, 2003). Although this produces problems in large vocabulary recognition, it has no problem in small vocabulary systems, which shows how word models suit and succeed with small vocabulary recognition (Martha, 2003; and Lee,

1990). Several words spoken together (or phrases) can also be taken as units of recognition. When phrase are modeled, they are considered as one long word (Martha, 2003).

2.2.3.1 Isolated word recognition

Depending on the flow of the speech signal, speech recognition systems can be catalogued into discrete or isolated and continuous speech recognition systems. In the latter case, speakers are expected to speak in a natural way. However, it is difficult to model and build than discrete or isolated word recognition. In isolated word recognition, speakers are expected to insert artificial pauses while talking. Although this is not a natural way of communication, isolated word recognition is easier to develop since word boundaries are clearly identified and inter-word co-articulation effects are minimized (Markowitz, 1996). The pattern-matching process could reliably be performed without being concerned about uncertainties in the endpoints of the patterns being compared (Juang and Rabiner, 1993). Many task-oriented isolated word recognition systems have been developed and performed quite well. The performance of isolated word recognizers increases substantially when designed and used for a particular task (Martha, 2003). The appropriateness of isolated word recognition systems for command and control applications in which the user is required to speak the command words one at a time (i.e., with distinct pauses between command words) works well (Juang and Rabiner, 1993).

2.2.4 Communication

After the recognition task is concluded, the recognized input signal should be sent to the hardware/software system that the target user uses for communication purposes. This communication aspect of speech recognition is the one that provides the human-computer interface. In order to have a better speech recognition system interface, synchronization should

be maintained between the recognizer and the hardware/software system that uses it (Markowitz, 1996).

2.3 The Speech Recognition Process

When a speaker's words that are uttered come into a speech recognition system, they are recorded and digitized. The speech is then acoustically analyzed with Fast Fourier Transformation (FFT) resulting in a sequence of feature vectors shown in figure 2.4. This diminishes the speech input to the only information needed to recognize it. The feature vectors consist of energy and frequency components and they are the input to the speech recognizer. The recognition process itself is actually a search process. The system tries to find the most likely sequence of words that matches the input sequence of vectors. The feature vectors are assigned to the basic acoustic units (a word or a phoneme), which is done with help of an acoustic model (Johnson and Garmark, 2000).

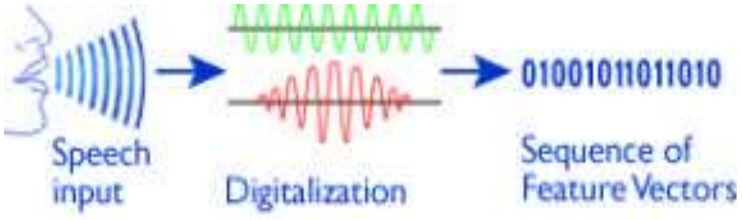

Figure 2.4 Feature extraction process (adapted from: Johnson and Garmark, 2000).

The acoustic model describes the pattern of how a certain person or groups of people speak as shown in figure 2.5. During the recognition, the incoming signal is compared to the patterns known from training. The pattern that has the least difference compared to the incoming signal is then the most likely sequence of phonemes. The acoustic unit is based on Hidden Markov Model (HMM) theory. An HMM describes the speaking rate and the acoustic unit, for example a phoneme or a word. The speaking rate is important to model since the system should manage to

understand both slow and fast speakers. The Hidden Markov Model is a flexible model, but it can be trained by efficient algorithms to recognise speech (Johnson and Garmark, 2000).

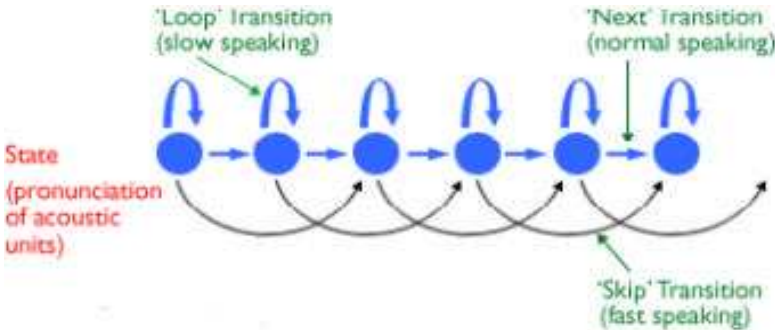

Figure 2.5 an acoustic model (adapted from: Johnson and Garmark, 2000).

When a sequence of phonemes is found, the system must find a word that matches its acoustic pattern. A lexicon that has the sequences of phonemes for all words, a phonetic transcription, is used to find a match. To increase the accuracy rate a language model is also used. It has statistical data or grammar rules that describe in what order words are most likely to come. The speech recognition technique used for dictation purposes takes great advantage of the language model (Johnson and Garmark, 2000). This drastically reduces the number of likely words that can follow a word.

2.3.1 Speech Understanding

What words should the speech recognition system pass on to the understanding unit? If only the most likely word sequence is passed on, there is a risk of losing information if one of the words would be wrongly interpreted (Johnson and Garmark, 2000). To keep information loss at bay, the system has to pass on several possible word sequences, which would be highly redundant; and a solution is to use a word graph, analogous to the one shown in figure 2.6.

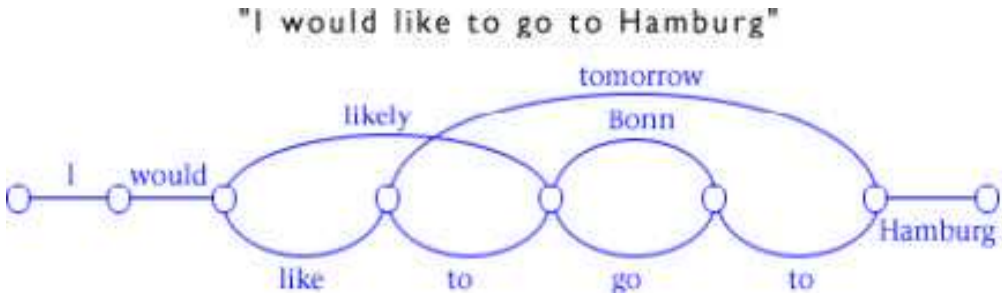

Figure 2.6 an example of a word graph describing the phrase "I would like to go to Hamburg" (adapted from: Johnson and Garmark, 2000).

In a word graph, sentence hypothesis can be mined by following a complete path through the graph. A statistical grammar is used to find the optimal path in the graph and hence the most likely sentence. For speaker dependent recognition systems, the word graph is based on language grammar; while for speaker independent recognition systems, it is based on expected words (Johnson and Garmark, 2000).

2.4 Related Works

In this section related literatures were reviewed that describe some of the works carried out on speech recognition for international and local languages. Attention was given to works that focused on speaker independence and those that used Hidden Markov Models as their modeling approach.

2.4.1 ASR Researches on International Languages

Deshmukh et al., (1999) outlined the use of acoustic parametres to cope with inter-speaker variability (differences in the vocal tract characteristics of speakers) for English language. In this work, it was shown that Acoustic Parametres (APs), which are based on phonetic features, target the linguistic information in the speech signal, and hence are capable of outperforming the

traditional MFCCs (Mel-Frequency Cepstral Coefficients) for speaker independent speech recognition systems. Better results were found for models trained with boys' speech and tested with girls' speech and vice versa, and trained with adults' speech and tested with children's speech and vice versa.

Nogueiras et al., (2001) presented an HMM based approach to emotion recognition in Spanish. An accuracy of more than 80% was achieved on the INTERFACE project whose main goal was the construction of a real-time multi-lingual speaker-independent emotion recognizer.

Gauvain and Lamel (1993) described speaker independent phone recognition for French based on BREF database. BREF is a large read-speech corpus, containing over 100 hours of speech material, from 120 speakers. An HMM based recognizer was trained with hand-verified data from 43 speakers. Using 35 context independent phone models, a baseline phone accuracy of 60% was achieved on an independent test of 7635 phone segments from 19 new speakers.

2.4.2 ASR Researches on Local Languages

Speech recognition researches have been conducted and speech recognition systems have been built for local languages to take advantage of the whole host of its applications. Some of these researches are briefly outlined below.

Hidden Markov Models were used to develop a large vocabulary, speaker independent, continuous speech recognition for Amharic language (Zegaye, 2003). A database comprised of 8000 utterances used for training and more than 500 sentences for development and evaluation was used. Performance tests were conducted at various stages and a 79% word level correctness, 76.18% word accuracy, and 30.01% sentence level correctness was obtained. The researcher

recommended that other approaches like hybrid HMM and Artificial Neural Networks (ANNs) be deployed to develop Amharic speech recognizers.

The same approach and model with that of Amharic was implemented for Tigrigna language (Hafte, 2009). A database consisting of 250 utterances used for training and 50 sentences for testing and evaluation was deployed. Performance tests showed that a 60.20% word level correctness, 58.97% word accuracy, and 20.06% sentence level correctness was achieved. The researcher recommended that approaches other than HMMs be made with Neural Networks or Hybrid types to develop Tigrigna speech recognizers.

An isolated word speaker independent speech recognizer based on Hidden Markov Model was developed for Oromifa language (Ashenafi, 2009). Context independent word based models and context dependent phoneme based models were developed. A database of 1000 utterances uttered by 20 different people was used, of which 2/3 was used for training and the rest 1/3 used for evaluation. Performance tests showed that 82.83% and 81.081% word level accuracy was obtained for context dependent phone based models and context independent word based models, respectively. The researcher recommended that further endeavours be made with continuous and large vocabulary scenarios. He also recommended the use of larger sub-word units like syllables to improve system performance, and the development of speaker adaptation systems for the same language.

2.5 The Sphinx-4 System

2.5.1 Introduction

Researchers working on speech recognition systems are usually faced with the challenge of developing an entire system from scratch even though they want to address a particular aspect of the field. Open source speech recognition systems exist such as HTK (Hidden Markov Model Toolkit) and Sphinx which can typically suit a particular approach to speech recognition systems design (Walker et al., 2004), but creating barriers for future researches that deviate from the targeted purpose of the system.

Sphinx-4 is an open source state of the art methodology that fosters new innovation in speech recognition systems (Lamere et al., 2003). It is a Java based speech recognition tool made available to a wide spectrum of development environments. It is "a modular and pluggable framework that incorporates design patterns from existing systems, with sufficient flexibility to support emerging areas of research interest" (Walker et al., 2004). It is modular in that it consists of discrete modules bestowed to particular tasks, and is pluggable in that the modules can easily be substituted at runtime.

2.5.2 The Sphinx-4 Framework

Sphinx-4 has been developed with high level of flexibility and modularity. Figure 2.7 shows the overall architecture of the Sphinx-4 system, where the labeled elements show the modules that can easily be replaced enabling researchers to work with various module implementations without modifying other portions of the system (Walker et al., 2004).

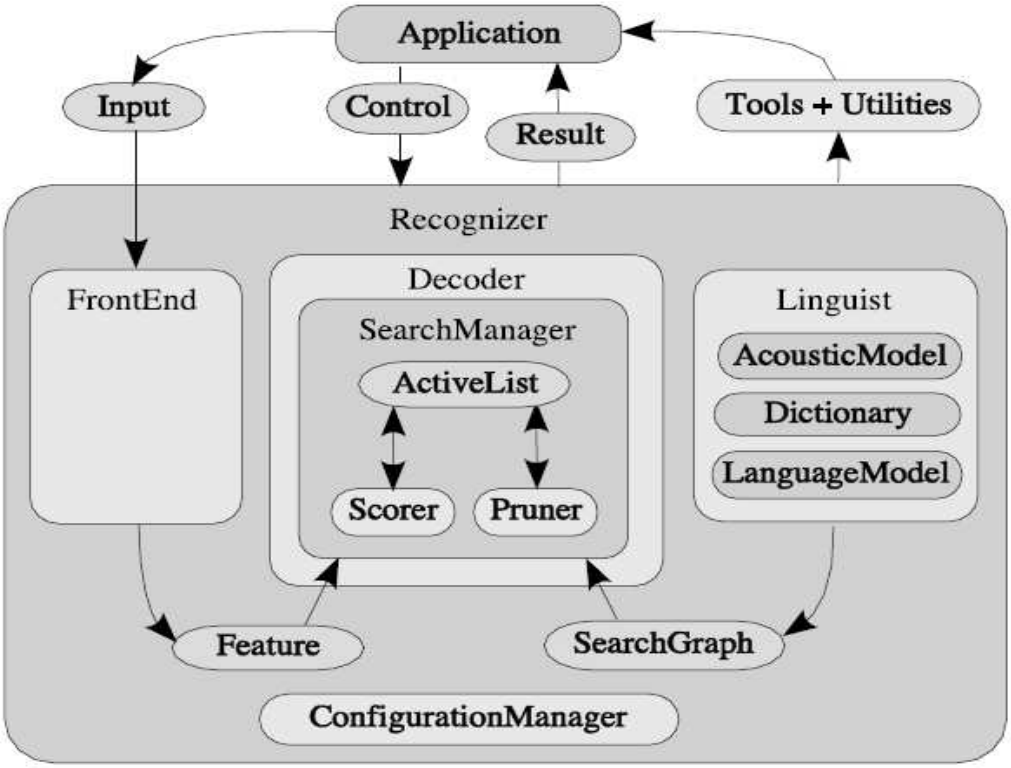

Fig. 2.7 Sphinx-4 Decoder Framework. The main blocks are the Front End, the Decoder, and the Linguist (adapted from: Walker et al., 2004).

The Sphinx-4 framework has three principal modules: the FrontEnd, the Decoder, and the Linguist. The FrontEnd accepts one or more signals as input and parametrizes them into a sequence of Features. The Linguist transforms any kind of standard language model together with pronounciation details from the Dictionary and structural details from one or more sets of acoustic models, into a search graph. The search manager in the Decoder uses the Features of the FrontEnd and the search graph of the Linguist to carry out the actual decoding, which produces Results. Prior to or at the time of recognition, Controls can be issued to each of the modules making them partners to the recognition process (Ibid.).

Like most other speech recognition systems, Sphinx-4 has many configurable parametres, such as the search beam size, which are used for adjusting system performance, and the

ConfiguratonManager which is responsible for handling such tasks. On the other hand, the ConfigurationManager makes its presence felt by allowing Sphinx-4 to exhibit a unique feature that allows it to dynamically load and configure modules at runtime thereby making it flexible and pluggable. For instance, Sphinx-4 is typically configured with a FrontEnd that generates Mel-Frequency Cepstral Coefficients (MFCCs) (Davis and Mermelstein, 1980). However, the ConfigurationManager can be used to reconfigure the Sphinx-4 system to build a different FrontEnd that generates Perceptual Linear Prediction (PLP) coefficients without modifying the source codes or recompiling the system (Hermansky, 1990).

Sphinx-4 comes with a number of Tools that provide applications and developers the ability to track decoder statistics like word error rate, runtime speed, and memory usage. These Tools are highly configurable for a variety of systems analysis tasks, and provide an interactive runtime environment that allows for rapid experimentation with a number of parametre settings (Walker et al., 2004).

Moreover, Sphinx-4 provides Utilities that support application level processing of recognition outputs, such as obtaining result lattices, confidence scores, and natural language understanding (Ibid.).

a) The FrontEnd

The FrontEnd is used to parametrize an input signal (e.g. audio) into a series of output Features. It consists of one or more multiple chains of replaceable communicating signal processing modules called DataProcessors, shown in Figure 2.8. These multiple chains allow the FrontEnd to simultaneously compute different types of parametres from the same or different input signals,

hence enabling simultaneous decoding using MFCCs and PLP, and also parametres derived from non-speech signals like video (Walker et al., 2004).

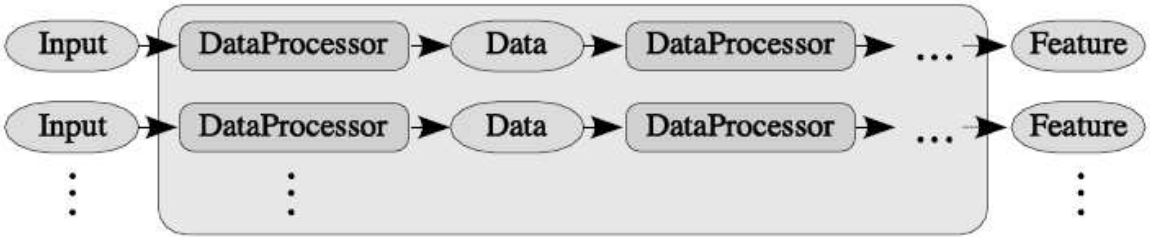

Fig. 2.8 The Sphinx-4 FrontEnd consisting one or more parallel chains of communicating DataProcessors.

Each DataProcessor provides an input and an output that can be linked to another DataProcessor, allowing arbitrary long series of chains. Each DataProcessor's inputs and outputs are generic Data objects that encapsulate processed input data in addition to the markers that show data classification events like end-point detection. The final DataProcessor in each chain generates a Data object composed of parametrized signals, called Features, which will then be used by the Decoder (Ibid.).

Like other systems, Sphinx-4 has the ability to generate parallel series of features, but is unique in that it allows an arbitrary number of parallel streams.

The communication between the blocks follows a pull design, where a DataProcessor asks a previous module for input when needed, in contrary to the push design, where a module forwards its output to the upcoming module as soon as it is produced. This pull design allows the processors to carry out buffering tasks enabling consumers to look forwards or backwards in time, and hence allowing the Decoder to perform Viterbi searches, and other searches like depth-first and A* (Ibid.).

Within the generic FrontEnd structure, Sphinx-4 provides a set of DataProcessors that implement common signal processing techniques. These implementations include support for reading from different input formats for batch mode operation, reading from an input audio device for live mode operation, mel-frequency filtering, and linear predictive coding (LPC) among others (Ibid.).

b) The Decoder

The Decoder in Sphinx-4 is responsible for the actual recognition task (Elshafei et al., 2008). It takes the Features generated by the FrontEnd and the SearchGraph provided by the Linguist to recognize the speech and generate Result (Walker et al., 2004, and Elshafei et al., 2008). The Decoder consists of the SearchManager, which is the most interesting, and other supporting codes that make the decoding process easier.

The Decoder is merely responsible for telling the SearchManager to identify a set of Feature frames. While executing, the SearchManager creates a Result object that contains all the paths that have arrived at a final non-emitting state. Sphinx-4 processes the result by providing utilities which can produce lattice and confidence scores from the Result (Walker et al., 2004).

Implementation of the SeachManagers is based on a token passing algorithm. A token in Sphinx-4 is "an object that is associated with a SearchState and contains the overall acoustic and language scores of the path at a given point, a reference to the SearchState, a reference to an input Feature frame, and other relevant information." The SearchState reference enables the SearchManager to link a token to its output distribution, pronounication, context-dependent phonetic unit, word, and grammar state. The SearchManagers are supported by a sub-framework consisting of an ActiveList, a Pruner, and a Scorer (Ibid.).

ActiveLists are generated from currently active tokens in the search lattice by pruning with a pluggable pruner. The Pruner can be implemented to carry out both relative and absolute beam pruning, which is highly simplified by the garbage collector of the Java platform. The Scorer is a pluggable state probability estimation module which offers state output density values when needed. Upon request from the SearchManager for a state at a given time, the Scorer evaluates the feature vector for that time and computes the score with mathematical operations. For parallel decoding using parallel acoustic models, the Scorer matches the acoustic model set to be used with the feature vector (Ibid.).

c) The Linguist

The Linguist contains the details of the recognized language. It generates the SearchGraph which the Decoder uses during searching, while encapsulating the difficulties involved in producing this graph (Walker et al., 2004, and Elshafei et al., 2008).

A Linguist develops the SearchGraph using the language structure depicted by a given LanguageModel and the topological structure of the AcousticModel (HMMs for fundamental sound units used). The Linguist may also use a Pronounciation Dictionary to make word mapping from the LanguageModel to the AcousticModel. While producing the SearchGraph, the Linguist may also include sub-word units with contents of arbitrary length (Walker et al., 2004). The Linguist consists of three modules: the LanguageModel, the Dictionary, and the AcousticModel (Walker et al., 2004, and Elshafei et al., 2008), which are discussed below.

A. The LanguageModel

This module provides the grammar and word-level language structure used by the system, which can be represented by any number of pluggable implementations (Walker et al., 2004, and

Elshafei et al., 2008). These implementations can either be graph-driven grammars or stochastic N-gram models. The former represents a directed word graph, each node representing a single word and each arc depicting the probability of an occurring word transition (Walker et al., 2004).

B. The Dictionary

This module provides the pronounciations of the words in the LanguageModel, acting as an interface between the AcousticModel and the LanguageModel. It contains the available words of the language and their pronounciation equivalent in terms of the phonemes in the AcousticModel (Walker et al., 2004, and Elshafei et al., 2008). Sphinx-4 provides Dictionary interface implementations that support the CMU Pronounciation Dictionary. Based on the size of the active vocabulary, these implementations provide optimal usage patterns (Walker et al., 2004).

C. The AcousticModel

This module helps the Hidden Markov Models to be deployed to recognize the speech, and provides the mapping between a speech unit and an HMM that can be scored alongside arriving features provided by the FrontEnd (Walker et al., 2004, and Elshafei et al., 2008). The mapping process may consider contextual and word position information. For instance, in triphones, the context shows the single phonemes to the left and right of the given phoneme, and the word position shows whether the triphone is at the start, middle, or end of a word (is a word itself) (Walker et al., 2004).

The Linguist splits each word in the active vocabulary into a series of context-dependent sub-word units. It then transfers the units together with their contexts to the AcousticModel, retrieving the HMM graphs related to those units. These HMM graphs are then used with the LanguageModel to build the SearchGraph (Ibid.).

Sphinx-4 represents the HMM graphs as a mere directed graph of objects, in contrary to other systems which represent as a fixed structure in memory. Each node in the HMM graph corresponds to an HMM state while the arcs show the probability of transforming from one state to another in the HMM. Sphinx-4 also enables variation in the number of states of an HMM from one unit to another within the same AcousticModel (Ibid.).

Each of the states in the HMM are capable of generating a score from an observed feature. The actual code for calculating the score is performed by the HMM state itself, encapsulating its implementation from the rest of the system. The AcousticModel also paves the way for various modules to be shared at all levels; the modules that make up a particular HMM state such as Gaussian mixtures, transition matrices, and mixture weights can be shared by any of the HMM states to an excellent level (Ibid.).

CHAPTER THREE

THE WOLAYTTA LANGUAGE

3.1 Background

This chapter presents the Wolaytta language phonology and writing system. Wolaytta is the language spoken by the Wolaytta people mainly inhabiting the Ethiopian provinces of Gamu Gofa and Sidamo (the now Wolaytta zone and some parts of the Southern Nations, Nationalities, and People's Region of Ethiopia) (Lamberti and Sottile, 1997; and Abebe, 2002). It is one of the main languages of the Ometo group of the Omotic (or 'west cushitic' in Lamberti's terms) family, which belongs to the Afro-asiatic language phylum. The Wolaytta people are surrounded in the west and in the south by populations (such as the Dawro, the K'uc'a, the Borodda and the Gamu) who speak another Ometo dialect and thus an idiom very similar to Wolaytta, and in the east and in the north by people speaking a Burji-Sidamo language (such as the T'ambaro, the Kambata and the Sidamo, from the last of which are separated by the River Bilatte) (Lamberti and Sottile, 1997). The native people call their language "Wolaytta" (*wolaittáttuwa* in their language). The language is also referred to as *woláítta dóónaa* (literally mouth of Wolaytta) or *woláítta Káálaa* (literally word of Wolaytta) (Motomichi, 2008). The term "Wolaytta" is also used as the name by which the people refer to themselves and their region (Hirut, 1999; Motomichi, 2008). The Wolaytta language officially uses Latin alphabets for writing purpose (including educational purposes) even though Ethiopic characters are also found to be sometimes used. Even though the Wolaytta language is believed to have a wide span of services like other Ethiopian languages, it is not well studied and a suitable platform has not been laid until recently for its growth (Alemaayehu and Tereezzaa, 1991 E.C.). Stressing this, Bruce Adams (1983) said

"when the languages in Ethiopia are measured in terms of the social services they give, Wolaytta is in fourth widely spoken language next to Oromifa, Tigrigna, and Amharic, but it is fascinating that not much research has been conducted on the language until recently".

3.2 Phonology

Phonology is a branch of Linguistics that studies the physical sounds of human speech and is concerned with the physical properties of phonemes, and the processes of their physiological production (Zue et al., 1996). As is the case with all languages, the speech sounds of Wolaytta fall in one of the two classes, consonants and vowels (Motomichi, 2008).

3.2.1 Consonants

Two sounds are considered to be separate phonemes if they bring change of meaning in a pair of words (Hirut, 1999). The Wolaytta language has 29 consonant phonemes, including voiced glottalized consonants, which have been analyzed as consonant clusters (Motomichi, 2008). The consonantal phonemes used in Wolaytta are the following:

Voiceles	Stop: *p, t, k, 7 (?)*
	Fricative: *s, sh, h, nh*
	Affricate: *ch*
Voiced	Stop: *b, d, g*
	Fricative: *z, zh*
	Affricate: *j*
	Nasal: *m, n*
	Liquid: *r, l*
	Approximant: *w, y*
Glottalized	Voiceless: *P (ph), T (x), K (Q), C (c)*
	Voiced: *D (dh), L M, N*

Table 3.1: Wolaytta Consonantal Phoneme Inventory

The following are example words that contain the phoneme ‘t’.

tir-áa ‘chest’

tookk-íis ‘he carried’

me***t***-úwa ‘problem’

misa***t***-íis ‘he seemed’

kawóte***tt***-aa ‘kingdom’

wo***tt***-íis ‘he put’

gis***t***-íya ‘wheat’

men***t***-íis ‘he broke’

The following are examples of words that contain the phoneme ‘k’.

keett-áa ‘house’

ko***kk***or-íis ‘he trembled’

ka***K***-áa ‘cliff’

7a***k***ee***k***-íis ‘he became careful’

lí***kk***-e ‘correct’

mo***kk***-íis ‘it grew’

3.2.2 Vowels

Vowels are speech sounds generated without an obstruction of airflow in the oral cavity. Five vowels are established as vowel phonemes of the Wolaytta language: /i/, /e/, /a/, /o/, and /u/.

There are five short vowel phonemes in Wolaytta (Motomichi, 2008), which are briefly discussed below.

/i/

This is usually realized as a *high front* vowel (Motomichi. 2008).

The following are examples of words that contain the phoneme /i/.

7***i***sh-áa ‘brother’

7***i***TT-***íi***s ‘he hated’

7***i***dimm-***íi***s ‘he embraced’

g-***íí***d***í*** ‘he having said’

/e/

This is realized as a *mid front* vowel (Motomichi, 2008).

The following are examples of words that contain the phoneme /e/.

7***e***tá ‘them’

7***e***C***e***r-íya ‘mouse’

d***e***r***e***KK-íis ‘he opened his eyes wide’

l***éé***7-***e*** ‘thin’

/a/

This is realized as a *low* vowel (Motomichi, 2008).

The following are examples of words that contain the phoneme /a/.

7***a***cc-***áa*** ‘tooth’

7***á***tt-iis ‘he was saved’

t***á***mm-***á*** ‘ten’

7***a***gg-***á*** ‘cease!’

/o/

This is realized as a *mid back* round vowel (Motomichi, 2008).

The following are examples of words that contain the phoneme /o/.

7***o***g-íya ‘way‘

g***o***r***o***m***óó***t-iya ‘evil eye‘

deeT-***ó*** ‘heavy’

7anj-***ó*** ‘may he bless’

/u/

This is realized as a *high back* round vowel (Motomichi, 2008).

The following are examples of words that contain the phoneme /u/.

7***u***bb-áa ‘all’

7***u***s***ú***pp***u***n-a ‘six‘

daap***u***r-íis ‘he became tired‘

b***úú***CC-***ú*** ‘let her mow‘

Corresponding to the five short vowels discussed previously, there are five long vowels in the Wolaytta language: /ii/, /ee/, /aa/, /oo/, and /uu/ (Motomichi, 2008). The following are examples of words that contain these long vowels with their English equivalent.

7***íí***t-a ‘bad’

lo77-***íi*** ‘is it good?’

7***ee***s-úwa ‘speed’

shar***éé***cc-uwa ‘witch doctor’

7***aa***tt-***íi***s ‘he transferred’

bullácc-***aa*** ‘wedding’

7***oo***rátt-a ‘new’

gorom***óó***t-iya ‘evil eye’

7***úú***z-iya ‘stingy one’

KunC***úú***t-iya ‘thread’

3.2.3 Diphthongs

Diphthongs are sounds formed by the combination of two vowels in a single syllable. Wolaytta has four diphthongs: /ai/, /au/, /oi/, and /ui/ (Motomichi, 2008). The following are examples of words that contain these diphthongs with their phoneme equivalent.

7***ai***ll-íya ‘7 ai ll i y a’

7up***áí***tt-iis ‘7 u p ai tt ii s’

7***au***-g-áá ‘7 au g aa’

dad-***áu*** ‘d a d au’

7***oi***cc-íis ‘7 oi cc ii s’

k***óí***r-o ‘k oi r o’

s***úí***K-iis ‘s ui K ii s’

s***ui***ll-íya ‘s u ill i y a‘

3.3 Syllables

The possible syllables of Wolaytta can be formulated as CV (V) (C), where C stands for a consonant and V for a vowel, parenthesized segments being optional. This formulation applies to

any positions of a word (Motomichi, 2008). These syllables include V, VV, VC, VCC, CV, CVV, CVC, CVVC, and CVCC.

The following are examples of words that contain these syllables.

Syllable structure	**Wolaytta Word**	**English Equivalent**
V	Aguntta	'thorn'
VV	Aawa	'father'
VC	int'ar'sa	'tongue'
VCC	Indde	'neck'
CVV	Maataa	'grass'
CVC	Zokuwa	'back'
CVVC	Keetta	'house'
CVCC	Galbba	'hide'

Table 3.2: Example of Words with Syllables (adapted from: Tewodros, 2009)

3.4 Tones and Stress

The prosodic system of the Wolaytta language is quite complicated and requires further research. Wolaytta does not seem to be a tone language. The language constitutes few 'homonyms'[2] which might theoretically constitute some minimal pairs differing from each other only in their tone structure (Lamberti and Sottile, 1997). However, Lamberti and Sottile (1997) believe that they are really homonyms i.e. thereby the tone has no distinctive or phonological relevance. Examples of these homonyms include:

bita (witchcraft) : bita (bewitch!)

t'uma (darkness) : t'uma (become dark!)

t'aanta (udder) : t'aanta (suckle!)

[2] Homonyms are words that sound the same but have different meaning and/or spellings.

k’aant’a (short)	:	k’aant’a (cut!)		
ola (war)	:	ola (throw!)		
c’ucca (louse)	:	c’ucca (saliva)	:	c’ucca (spit!)
gusa (sperm)	:	gusa (have diarrhoea!)		
t’iink’o (stench)	:	t’iink’o (let him stink!)		
na?a (boy)	:	na?a (two)		
sa?a (land)	:	sa?a (bite!)		

CHAPTER FOUR

EXPERIMENTATION AND PERFORMANCE EVALUATION

4.1 Introduction

This chapter presents the design, development, and performance evaluation of the Wolaytta speech recognizer which is capable of understanding Wolaytta speech. The main target was to build and test the appropriateness of a new context dependent and context independent language model for Wolaytta speech recognizer.

The experimentation activity was conducted on a 1 GB RAM, 120 GB size hard disk Toshiba Satellite Series Laptop, on a 32-bit Windows Vista Home Edition platform using the Sphinx tool. Two types of recognizers were built for the Wolaytta language: context dependent isolated word recognizer and context independent isolated word recognizer.

4.2 The Recognizers' System Design

Systems design refers to how a particular system is built, and how its different components (sub systems) communicate and interact within themselves and other outside components in an imaginary way. In this section, the Wolaytta speech recognizers' system design is briefly discussed which is shown in Figure 4.1.

As can be seen from Figure 4.1 below, the training corpus consists of Wolaytta audio files and their text transcriptions, which are converted to feature vectors upon the control of a control file. These extracted feature vectors are then used together with the dictionaries and the phone set to produce the acoustic model, which contains model definition files and other files that contain statistical information. The outputs of the acoustic model are then coupled with the language

model, the lexicon and test audio files from the test corpus, and given as inputs to the Sphinx-4 decoder, which with the aid of the Accuracy tracker component tracks the accuracy of the recognition system and outputs recognition results.

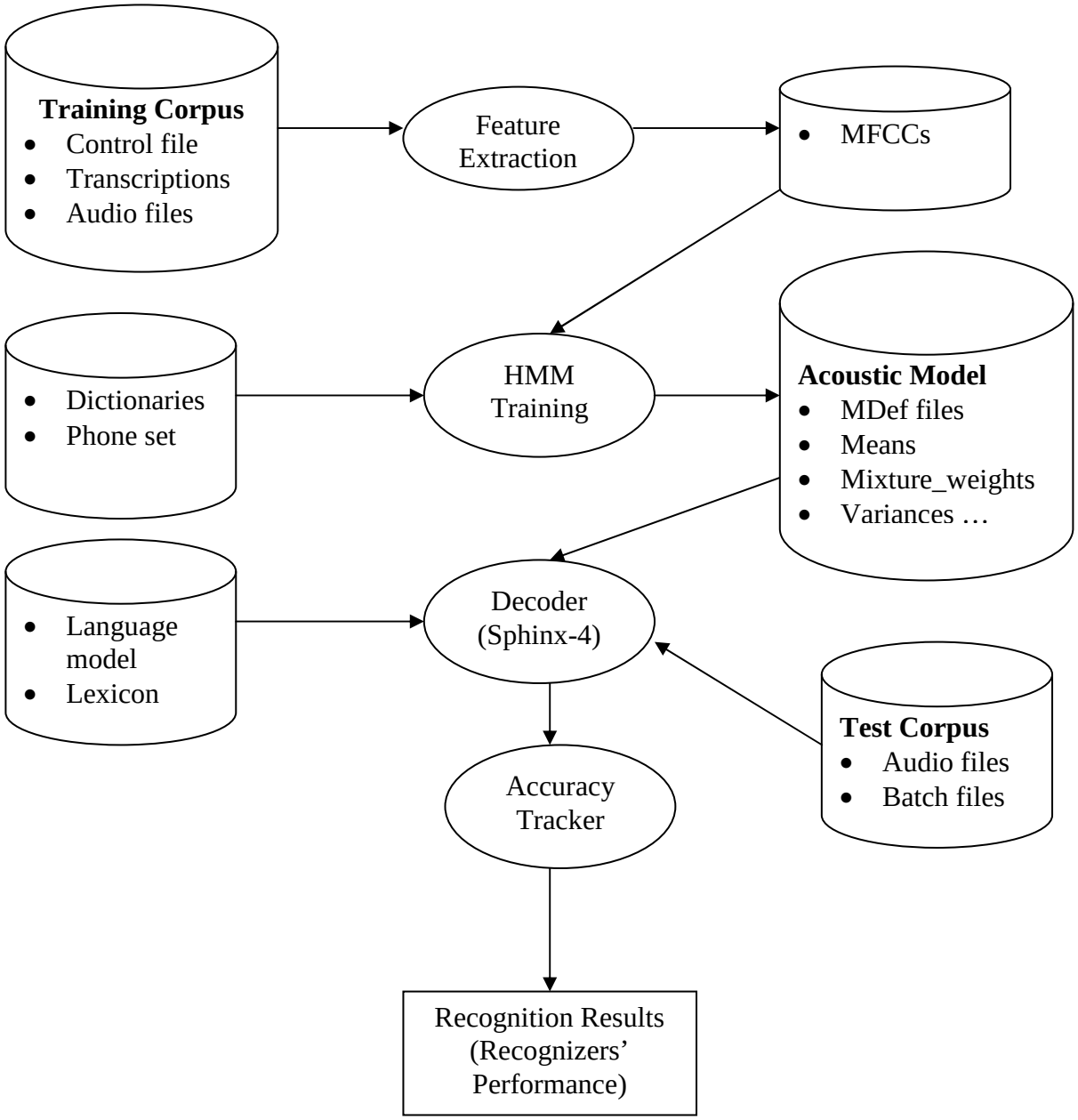

Fig. 4.1 Wolaytta speech recognizer design

4.3 Data Preparation and Preprocessing

Data preparation and preprocessing was the first task carried out as far as the experimentation activity is concerned. Data preparation involved selecting the words to be uttered for experimentation purpose upon the consultation of a domain expert, and making these words uttered by a speaker. Hence, 450 Wolaytta words were selected, uttered, and then recorded. The recording was conducted in a quiet room to enhance the quality of the recorded speech used for the experimentation. The speech database was composed of these recorded speech utterances. The speech was sampled and recorded at 16 KHz using the Praat tool. Praat is a speech analysis and synthesis programme capable of providing quality speech data (Martha, 2003).

Once all the speech was recorded, 2/3 of the recorded corpus was used for training the model while the remaining third was used for testing the system based on the number of words used for building the corpus. The following table shows the number of words used for training and testing to develop the models.

Total words used	450
Words used for training the models	300
Test words used deployed in the training set	100
Test words used not deployed in the training set	150

Table 4.1 Speech Corpus

The Dictionary

Preparing the dictionaries was one of the preprocessing activities carried out to make the recorded data ready for training and testing. Two types of dictionaries were prepared for this research's experimentation task: a pronounciation dictionary and a filler dictionary. A

pronounciation dictionary is a text file which contains in a sorted order (alphabetically) the words that are used and recognized by a speech recognition system, and the way they are pronounced in terms of sub-word units or phonemes. It describes the series of Hidden Markov Models (HMMs) that constitute each word (Kinfe, 2002). The pronounciation dictionary used in this research consisted of phoneme-based models, which consisted of words mapped to their string phoneme equivalent. The phoneme-based model dictionary was originally generated by an on-line tool[3], and then manually fine tuned to comply with the syllabic nature of the Wolaytta language. For the purpose of this research work, each word was assumed to have one pronounciation, which is a sequence of phones. This sequence is similar to what one might find in dictionaries such as Oxford or Cambridge. The phone-based model dictionary used a set of 49 phones. Examples of the entries of these two sets of dictionaries are shown below:

Phoneme based model dictionary	
MAAZHIIS	M AA ZH II S
DARO	D A R O
TAASSI	T AA SS I

Table 4.2 Example entries in the dictionaries

In addition to the main dictionary discussed above, the Wolaytta recognizer built in this research made use of a filler dictionary which shows a string representation of the pauses between words and phonemes, and silences at the beginning and end of utterances. The following table shows contents of the filler dictionary:

[3] Sphinx Knowledge Base Tool – found at http://www.speech.cs.cmu.edu/tools/lmtool-adv.html

Representation	Description
<s>	Silence at the beginning of a word or phoneme
<sil>	Silence (pause) between words or phonemes
</s>	Silence at the conclusion of a word or phoneme

Table 4.3 Contents against description of the contents of the filler dictionary (wolaytta.filler)

The Phonemes

The set of phonemes used for the experimentation process were manually extracted from the main pronounciation dictionary generated above, and then manually selected to avoid redundancy. For example, the above main pronounication dictionary produced the following pronounciation equivalent for the word CARKUWA:

```
C A R K U W A
```

As can be seen, there are 7 phones for the word CARKUWA, and similar phone extraction processes were carried out on all the words found in the dictionary (including the filler dictionary which has the "SIL" string represented as it is) and put together in one file. Finally, the redundant phones were removed from this set to only have one representative of each phone in the phone set producing the ultimate phoneme set used for the experimentation.

4.4 Training the Hidden Markov Model

Automatic Speech Recognition systems need to learn (be trained with) the sounds (speech signals) that need to be decoded. In this case, training the model refers to estimating the Hidden Markov Model parametres, which include the means, mixture weights, transition matrices, and

variances (Lee et al., 1990). A total of 57 phonemes were used for the Wolaytta phoneme-based model dictionary, which were each modeled with HMM. Creation of these HMMs was carried out using the SphinxTrain package that comes with the Sphinx tool, putting it in the same directory as the Wolaytta Project, and running the following script from the command line, which readies the *wolaytta* task components for the training and model creation phase:

*C:\Research\wolaytta>**perl ../SphinxTrain/scripts_pl/setup_SphinxTrain.pl –task wolaytta***

The first activity that was carried out as far as the training phase is concerned was representing the speech signal in a way that its phonetic properties are emphasized. In Sphinx, this is done by the SphinxTrain module which makes use of the *make_feats* and *wave2feat* perl scripts to compute a 13-dimensional Mel Frequency Cepstral Coefficients (MFCCs) of the given speech signals. This is the perl script that actually creates these feature vectors:

*C:\Research\Wolaytta>**perl ./scripts_pl/make_feats.pl –ctl etc/wolaytta_train.fileids***

This command makes use of the control file *wolaytta_train.fileids*, which contains the location of the wave audio files used to train the model, and the configuration file *sphinx_train.cfg*, which contains configuration parametres of the training model. The command generated feature vectors of these audio files used for training the model, and put them inside the *feat* directory inside the *wolaytta* task folder.

Mel-Frequency Cepstral Coefficients (MFCCs) Feature Extraction

Mel-Frequency Cepstral Coefficients are well known features used to describe speech signal. They are based on the evidence that the information carried by low frequency components of speech signal are phonetically more important for human beings than those carried by the high

frequency components (Karpov, 2003). Computing MFCCs is based on short-term analysis, and hence from each frame an MFCC vector is computed. The feature extraction process in MFCC is depicted in figure 4.2. First, short term spectral analysis is calculated for the input speech frame (signals) by filtering and overlapping triangular band pass filters. Only the log-energy is calculated for each sub band, which is later subjected to Discrete Cosine Transform (DCT).

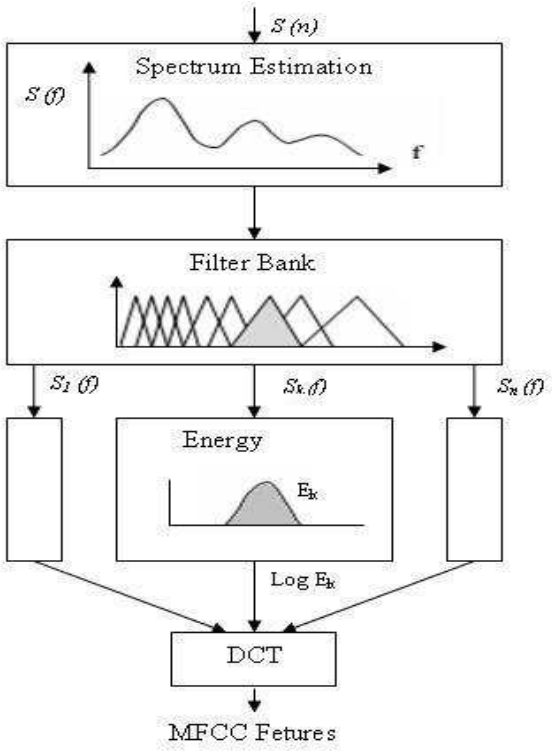

Figure 4.2 MFCC feature extraction (adapted from: Natarajan and Thangarajan, 2009)

After the feature files were obtained through feature extraction, they were used to train the acoustic models. SphinxTrain uses HMMs to characterize all the triphones that appear in the training corpus. Things would have been difficult to handle had different HMMs been assigned to every triphone. But this problem was solved by applying state sharing, where similar states from different HMMs were grouped into a set called a senone. The training process was then set

up using the *RunAll.pl* script (shown below), which produced the model definition files, and statistical and probability parametres of the HMM states.

*C:\Research\wolaytta>**perl ./scripts_pl/RunAll.pl***

The trained probabilities were then directly used in the recognition process.

In order to build the model for Wolaytta, the SphinxTrain package used the following set of files:

- Phone list: which specifies the phones used in training the Wolaytta speech recognition model with each phone on a separate line (wolaytta.phone);
- Dictionary: which specifies how each word used in the system is pronounced, and lists the phones used to make each word (wolaytta.dic);
- Filler dictionary: which specifies special words used in the system such as silence and pause (wolaytta.filler);
- Transcript: which specifies the word equivalent (transcript) of each audio file in the corpus in accordance with the dictionary (wolaytta_train.transcription);
- Speech files: which specify the audio equivalent (in wave format) of the words listed in the transcription file;

The SphinxTrain module produced the acoustic model, which together with the language model and the dictionary were used by the decoder and the testing module to recognize a set of speech data in accordance with the already built model. The acoustic model was put in a JAR file in order for the decoder (Sphinx-4 in this case) to easily manipulate it. The merit of having the acoustic model in a JAR file is that the JAR file can simply be incorporated in the classpath and then referenced in the configuration file for use with Sphinx-4.

Configuration and Control Files

A configuration file contains programme-defined, configurable configuration parametres and values that control the different modules incorporated in the Sphinx package. For example, training the HMM models required a configuration file *sphinx_train.cfg*, which was created while setting up the task for Wolaytta using the SphinxTrain package, and it packaged different variables and their respective values used for the successful accomplishment and control of the training phase

Control files on the other hand contain the paths (location) of audio files that were converted to feature vectors. The *wolaytta_train.fileids* is a control file which consisted of relative paths of the audio files that were converted to Mel-Frequency Cepstral Coefficients for training purpose, and ultimately building the acoustic model.

Vector Quantization

One of the processes carried out while building the model with the *RunAll* script was vector quantization, which is a data reduction technique that maps a real vector onto a discrete symbol. A vector quantizer is described by a codebook, or reproduction vector, where each prototype vector has the same dimension as the input vector. While doing the mapping, the input vector is matched against each prototype vector in the reproduction vector using some distortion measure. The *slave.VQ.pl* script is the command that carries out the quantization activity, and is integrated into the *RunAll* script (command).

Context-Independent HMM Training

Context-independent phone models are models which do not consider contexts (i.e. left and right neighbouring phones) when modeling. Sphinx is based on phonetic, 3-state Hidden Markov

Models (Lee et al., 1990). Fifty seven phonemes were identified for Wolaytta, and a hidden markov model was trained and built for each phoneme. Each phonetic HMM contains three discrete output distributions of the vector quantization symbols. Each distribution is a cumulative density of the three states' probability density functions, which are regarded as independent. The training procedure was initialized with a word-level database whereby forward-backward algorithm was then used to train the parametres of the 57 phonetic HMMs. The training database consisted of 300 unique words, and for each word, word HMMs were built by concatenating phone HMMs. These word HMMs were then concatenated to form sentence HMMs (if necessary), and then trained on the respective speech. Five iterations of each Gaussian (1, 2, 4, and 8) Baum Welch algorithm were conducted, and 57 context-independent phone models were generated.

Context-Dependent HMM Training

As the realization of a phone crucially depends on context, simple phone models are usually treated as inadequate. So, triphone (context-dependent) models were proposed to model most prominent contextual effects (Lee et al., 1990).

Context-dependent phone models are analogous to word-dependent phone models, except that instead of modeling phones in a word, they model phones in context. Context in this case refers to the immediate left and/or right neighbouring phone. A left context dependent phone is dependent on the left context, where as a right context dependent phone is dependent on the right context. A triphone model, which is one variant of context-dependent phone model, considers both the left and right neighbouring phones (Ibid.).

Different triphone models were used for both left and right context. As triphone models consider neighbouring phonetic contexts while modeling, they are powerful in modeling important co-articulatory effects, and are more reliable and thorough than phone modeling. Thus, they have brought good results. Even though triphone models have these merits, they are very large in number, and hence can only be sparsely trained. Otherwise, they cause memory wastage. In addition, they do not take into account the similarities among phones and their effect on one another (Ibid.).

The Acoustic Model

The first and basic principle that makes a speech recognition system crucial and robust is the acoustic model. Acoustic modeling is a vital process as it directly affects the search speed and accuracy of a recognition system. The design factors of acoustic modeling consist of a number of models that are appropriate for covering the target language and the size of the speech training corpus. The size of the training corpus has a direct effect on the system performance, and the number of acoustic models corresponds to linguistic knowledge of the target language (Kasuriya et al., 2004).

The acoustic model consists of a set of files which contain statistical representation of each distinct sound that make up a spoken word in the target language. It contains the sounds for each word used in the language grammar. These words give the speech recognition system the series of sounds that it must recognize. The system then listens for the sequence of sounds that constitute a particular word, and when it finds one, it returns the textual representation of that particular word (Ibid.).

The fundamental concept behind any acoustic model lies in the capabilities of the feature vectors to capture distinctive properties of the speech being modeled. An acoustic model is considered to be good if takes into account speaker variations, pronounciation variations, environmental variations, and context dependent phonetic coarticulation variations (Çomez, 2003). Hence, the acoustic training database has to somehow be large to get a robust acoustic model.

In Sphinx, the acoustic model is made of a set of left-to-right Hidden Markov Models (HMMs) with one HMM per unit. The units depict the phones in a triphone context. The acoustic model for this research was built using the SphinTrain module that comes with the Sphinx tool, and this module actually produced a set of model definition files (files with *.mdef* extensions) and statistical parametres (*means, mixture weights, transition matrices, and variances*) that were later used by the decoder module.

The Language Model

The language model is the other component that is used by recognizers to understand and decode speech signals. The purpose of a language model is to select (choose) the correct words from competing words with acoustic similarity in the decoder, and aids the acoustic model in selecting the correct sequence of words. In speech recognition and automatic translation systems, a language model is used to improve performances of these systems, and reduce the number of possible candidates for an output text. A language model tells the system the probability of a certain string of words being uttered (Zegaye, 2003). It typically determines the probability of words in given contexts; usually, of a given word being followed by another word or set of words. The language model is used by the recognizer to utilize context dependent monophone models, and model consistencies in the natural language. There are two language modeling

ways; Finite-state Grammar, and N-gram model. The latter approach was adopted in this research, where a corpus of representative text for the task was prepared, and then processed by an on-line tool[4] to create an N-gram language model. The produced N-gram language model (unigram in this research's case) was then used by Sphinx-4 for decoding and testing the performance of the recognizer. A unigram language model is adopted for this research as the research focused on independent, word-level models. The N-gram language model attempts to catch the syntactic and semantic constraints by calculating (guessing) the probability of a recognition unit (word, sub-word, or phoneme) in a sentence provided the previous (n – 1) recognition units (Ashenafi, 2009). An extract of the unigram language model deployed in this research is shown in Appendix 3.

4.5 The Recognition Process

The recognition process involved decoding the already built acoustic model using the appropriate decoder, and testing the performance of recognizer (decoder) against some set of test data. Two different tests were carried out. The first test involved randomly selecting 100 words from those words used for training the model and testing the model using these set of test words. The second test involved testing the model with 150 words not used for training the model, and testing the model using these set of independent test words.

Decoding refers to the process of searching for an uttered word or word sequence among words in a dictionary. As stipulated above, the acoustic model yielded from the training phase was integrated with the dictionary (lexicon) and the language model to carry out the decoding and testing activity with the Sphinx-4 decoder. Using the models from SphinxTrain involved defining the dictionary, the language model, the acoustic model, and the model loader.

[4] Sphinx Knowledge Base Tool – found at http://www.speech.cs.cmu.edu/tools/lmtool-adv.html

Sphinx-4 accepts and processes audio data only in the form of *raw* format (i.e. audio files with "*.raw"* extension) for decoding purpose. Thus, the already recorded *wave* files (with "*.wav*" extension) which were clustered to be used for testing were converted to *raw* format with the help of the *switch sound file converter* tool. A batch file was then created (*wolaytta.batch*) which contained the location (path) of these raw audio files together with their transcriptions.

After defining and integrating these components together and putting them in their respective correct locations, the *ant* command was run from the *Wolaytta* test directory to display the reference[5] and hypothesis[6] texts used in matching the acoustic model with the test set, and the recognition performance of the respective recognizers with their detailed information (which are discussed in section 4.6).

4.6 Recognition Results

As discussed in the previous sections, two types of models and recognizers were built for Wolyatta based on context-dependent (triphone-based) and context-independent (word-based) levels. To track and analyse the performance of the recognizer, the accuracy tracker component of Sphinx-4 was integrated and its features defined into the recognizer design, which accepts test speech utterances together with their manual transcriptions to test how well the recognizer has recognized the test utterances.

The following set of tables show the performance of the recognizer for Wolaytta based on the first test (i.e. test on 100 words which were also in the training set).

[5] Reference: is the reference or transcript that should be recognized.
[6] Hypothesis: is the result that is generated by the recognizer, and this is what was recognized.

Accuracy: 53.000%	Errors: 47	(Sub:47 Ins:0 Del:0)
Words: 100	Matches: 53	WER: 47.000%
Sentences: 100	Matches: 53	SentenceAcc: 53.000%
This Time Audio: 1.73s	Proc: 0.19s	Speed: 0.11 X real time
Total Time Audio: 185.79s	Proc: 27.53s	Speed: 0.15 X real time
Mem Total: 126.69 Mb		Free: 108.32 Mb
Used: This: 18.37 Mb	Avg: 17.01 Mb	Max: 22.81 Mb

Table 4.4 Performance of the context-dependent recognizer

As shown in table 4.4, 100 words were used as test set for the recognizer of which 53 were correctly recognized while the rest 47 were wrongly classified. No insertion and deletion errors were reported by the recognizer.

Accuracy: 41.000%	Errors: 76	(Sub:59 Ins:17 Del:0)
Words: 100	Matches: 41	WER: 76.000%
Sentences: 100	Matches: 34	SentenceAcc: 34.000%
This Time Audio: 1.73s	Proc: 0.72s	Speed: 0.41 X real time
Total Time Audio: 185.79s	Proc: 82.29s	Speed: 0.44 X real time
Mem Total: 126.69 Mb		Free: 53.88 Mb
Used: This: 72.81 Mb	Avg: 46.73 Mb	Max: 78.41 Mb

Table 4.5 Performance of the context-independent recognizer

As shown in table 4.5, 100 words were used for recognition of which 41 were correctly recognized while the rest 59 words were wrongly recognized. In addition to the incorrectly recognized words, 17 words were inserted wrongly.

The following set of tables on the other hand depict the performance of the recognizer based on the second test where tests were carried out on an independent set of 150 words not deployed in training the model.

Accuracy: 51.333%	Errors: 73 (Sub:73 Ins:0 Del: 0)	
Words: 150	Matches: 77	WER: 48.667%
Sentences: 150	Matches: 77	SentenceAcc: 51.333%
This Time Audio: 4.48s	Proc: 3.74s	Speed: 0.84 X real time
Total Time Audio: 337.60s	Proc: 310.72s	Speed: 0.92 X real time
Mem Total: 126.69 Mb		Free: 66.79 Mb
Used: This: 59.90 Mb	Avg: 81.09 Mb	Max: 118.53 Mb

Table 4.6 Performance of the context-dependent recognizer

As shown in table 4.6, 150 unique words were used by the recognizer for testing of which 77 were correctly classified, whereas the remaining 73 words were wrongly classified. This recognizer has not made insertion and deletion errors.

Accuracy: 46.667%	Errors: 99 (Sub:80 Ins:19 Del:0)	
Words: 150	Matches: 70	WER: 66.000%
Sentences: 150	Matches: 62	SentenceAcc: 41.333%
This Time Audio: 4.48s	Proc: 4.86s	Speed: 1.08 X real time
Total Time Audio: 337.60s	Proc: 364.07s	Speed: 1.08 X real time
Mem Total: 126.69 Mb		Free: 44.30 Mb
Used: This: 82.39 Mb	Avg: 84.06 Mb	Max: 124.26 Mb

Table 4.7 Performance of the context-independent recognizer

As can be seen from table 4.7, 70 of the 150 test words were correctly recognized by the context-independent recognizer while the remaining 80 words were wrongly recognized. In addition to these incorrect classifications, 19 wrong insertions were also reported.

The above results show that the lion's share of the errors committed by the respective recognizers were substitution errors, which are errors caused by transcribing a word (an utterance) incorrectly. For instance, the word *'daro'* was transcribed as *'deree'*, and the word *'wottaa'* was transcribed as *'boottaa'* while decoding, which arose due to similarities in some of the phonemes in the words.

The results also show that deletion errors were also committed by the recognizers. Deletion errors are errors which are caused by wrongly inserting a word (words) into an incorrect position while decoding. These types of errors come into the scene by the presence of some noise in the test data set.

As can be noted from the above tables, there were few deletion errors committed by the recognizers, which implies that very few words were not matched by the recognizers with any of the patterns in the acoustic model.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1 Introduction

This study aimed at experimenting the development of speaker dependent Wolaytta automatic speech recognition system from two perspectives: context-independent word level and context-dependent word level. Doing this, it also tried to compare these two models based on the recognition performance of the respective recognizers built.

This chapter discusses the conclusions made based on the experiments conducted and discussed in the other chapter, and propagates recommendations pertinent to further researches for the same language in particular and speech recognition field in general.

5.2 Conclusions

Various speech recognition technologies are currently in use and speech recognition systems are being built to integrate speech technology in different equipment (such as computers and cell phones).

In this research, an attempt was made to develop Hidden Markov Model based automatic speaker dependent speech recognition for Wolaytta. Context-dependent triphone based and context-independent monophone based recognizers were developed based on medium size vocabulary. Performance evaluation showed that promising results were achieved for both context-dependent and context-independent models.

The results showed that the context-dependent (triphone-based) model has out performed the context-independent (monophone-based) model achieving an accuracy of 53% while the latter model achieved an accuracy of 41%. In addition to the recognition performance, the recognition speed of the triphone-based recognizer was better than that of the monophone-based recognizer. This shows that modeling with triphones (context-dependent phonemes) produces better speech recognizers for Wolaytta than those recognizers modeled with monophones (context-independent phonemes). Albeit considering monophone-based and triphone-based modeling with the Sphinx tool, comparisons would have been sound had there been studies conducted on analyzing the appropriate feature vector extraction schemes for Wolaytta. Hence, it can be concluded that it is possible to build an HMM based speech recognizer using the Sphinx-4 tool for languages which share similar features with Wolaytta.

5.3 Recommendations

This research has attempted to see the prospect of developing a speaker dependent word based and triphone based speech recognition for Wolaytta. In order to improve the performance of the Wolaytta recognizer developed in this study and provide additional facilities and facets, the following recommendations were made for further research:

- A speech database made up of 450 words was used to build the recognizers in this work (which is medium size vocabulary). Further researches can extend this number to engulf large size vocabularies, and speaker independent scenarios where speakers from different gender, age group, and dialect can be considered for model building and decoding;

- This research was based on HMMs and used an HMM based tool SphinxTrain for training, and Sphinx-4 for decoding. Further researches can consider the deployment of other approaches such as Artificial Neural Networks (ANNs) or hybrid HMM-ANN;
- This thesis work considered only developing monophone and triphone based recognizers for Wolaytta. The extension of this endeavour can be working on other recognition units such as words and sub-word units such as syllables to enhance recognition performance;
- This research considered developing a recognizer for word level (isolated word) automatic speech recognition systems. This idea can further be researched by considering continuous speech utterances to entertain co-articulation effects amongst successive words;

References:

Abebe, A. (2002). Sociolinguistic survey report on the Ometo dialect of Ethiopia, part II. SIL Electronic Survey Reports 2002-012.

Adams, B. (1983). A tagmemic analysis of the WolaittaLanguage. Unpublished Doctoral Dissertation, The University of London.

Alemaayehu Doogamo and Tereezzaa Hayile Messqqalo (1991 E.C.) *Wolayttatto qaalatu Amaaratto birshshettaa* [Wolaytta-Amharic dictionary]. Addisaaba: *Tophphiyaa Doonatu Xinaatiyaanne Pilggettaa Ooso Keettaa, Addisaaba Yunivers*ttiyaa.

Ashenafi Demisse, (2009). A Speech Recognition System for Afaan Oromo. Master's Thesis, Addis Ababa University, Addis Ababa.

Çomez M. A., (2003). Large Vocabulary Continuous Speech Recognition for Turkish using HTK. Master's Thesis, The Middle East Technical University, Turkey.

Dai, J., (1994). Hybrid Approach to Speech Recognition using Hidden Markov Models and Markov Chains. IEE Proc.-Vis. Image Signal Process, Vol. 141, No. 5.

Daniel, J., James, H., (2000). Speech and Language Processing. Prentice Hall Publishing.

Davis, S. B., Mermelstein, P., (1980). Comparison of parametric representations for monosyllable word recognition in continuously spoken sentences. In IEEE Transactions on Acoustic, Speech and Signal Processing, vol. 28, no. 4.

Deshmukh, O., Espy-Wilson C. Y., Juneja A., (1999). Acoustic-Phonetic Speech Parameters for Speaker Independent Speech Recognition. Department of Electrical & Computer Engineering, University of Maryland.

Dutoit, T., (1995). A Short Introduction to Text-to-Speech. Kluwer Academic Publishers, Dordrecht Boston, London.

Elshafei, M., Al-Muhtaseb, H., Al-Ghamdi M., (2008). Speaker-Independent Natural Arabic Speech Recognition System. King Abdulaziz City of Science and technology.

Evgeny Karpov, (2003). Real-Time Speaker Identification. Master's Thesis, University of Joensuu, Finland.

Hafte Abera, (2009). Hidden Markov Model Based Large Vocabulary, Speaker Independent, Continuous Tigrigna Speech Recognition. Master's Thesis, Addis Ababa University, Addis Ababa.

Hermansky, H., (1990). Perceptual linear predictive (PLP) analysis of speech. Journal of the Acoustical Society of America, vol. 87, no. 4, pp. 1738–1752.

Hirut Woldemariam, (1999). Linguistic Description of the Wolaytta Language. Addis Ababa University.

Holmes, J., Holmes, W., (2003). Speech Synthesis and Recognition. Taylor and Francies New Fetter Lane, London ECAP 4EE.

Gauvain, J. L., Lamel, L. F., (1993). Speaker-Independent Phone Recognition Using BREF. LIMSI-CNRS, BP 133. 91403, Orsay cedex, FRANCE

Johnson, A., Garmark, S. (2000). Speech Recognition – Possibility and Usability for People with Disabilities. Lund Institute of Technology. Lund University, Sweden.

Juang, B. H., Rabiner, L. R., Levinson, S. E., Sondhi, M. M., (1985). Recent Developments in Applications of Hidden Markov Models to Speaker-Independent Isolated Word Recognition. AT&T Bell Laboratories, Murray Hill, New Jersey.

Juang, B. H., Rabiner L. R., (1993). Fundamentals of Speech Recognition. Englewood Cliffs, New Jersey: Prentice Hall, Inc.

Juang, B. H., Rabiner, L. R., (1986). An Introduction to Hidden Markov Models. IEEE ASSP Magazine.

Juang, B. H., Rabiner, L. R., (1991). Hidden Markov Models for Speech Recognition. Technometrics, Vol. 33, No. 3, pp. 251-272. AT&T Laboratories, Murray Hill, New Jersey.

Kasuriya, S. Kanokphara, S., Thatphithakkul, N., Cotsomrong, P., Sunpethniyom, T., (2004) Context-independent Acoustic Models for Thai Speech Recognition, ISCIT 2004, Sapporo, Japan, October 26 – 29.

Kinfe Tadesse (2002). Sub-word based Amharic Word Recognition: an Experiment using Hidden Markov Model (HMM). Master's Thesis, Addis Ababa University, Addis Ababa.

Lamberti M, and Sottile R., (1997). The Wolaytta language (Studia linguarum Africae Orientalis, 6.) Ko1n: Rudiger Koppe Verlag.

Lamere, P., Kwok, P., Walker, W., Gouvea, E., Singh, R., Raj, B., Wolf, P., (2003). Design of the CMU Sphinx-4 decoder. In Proceedings of the 8th European Conference on Speech Communication and Technology, Geneve, Switzerland, pp. 1181–1184.

Lee K. F., Hon, H. W., Hwang, M. Y., (1990). Recent Progress in the Sphinx Speech Recognition System. Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213.

Lee, K. (1990). Context-Dependent Phonetic Hidden Markov Models for Speaker independent Continuous Speech Recognition. In Reading in Speech Recognition edited by Alex Waibel and Kai-Fu Lee. pp. 347-365. California: Morgan Kuafman.

Lemma Lessa, (2003). Development of Stemming Algorithm for Wolaytta Language. Masters Thesis, Addis Ababa University, Addis Ababa.

Maheswari, N. U., Kabilan, A. P., Venkatesh, R., (2009). Speaker Independent Speech Recognition System Using Neural Networks. India.

Maheswari, N. U., Kabilan, A. P., Venkatesh, R., (2009). Speech Recognition System Based on Phonemes Using Neural Networks. International Journal of Computer Science and Network Security, Vol. 9, No. 7, Dindigul-624622, India.

Mark, G., Steve Y., (2008). The Application of Hidden Markov Models in Speech Recognition. Cambridge University Engineering Department, Vol. 1, No. 3, pp. 195-304, United Kingdom.

Markowitz, J. A., (1996). Using Speech Recognition. Upper Saddle River, New Jersey: Prentice Hall, Inc.

Martha Yifru, (2003). Application of Amharic Speech Recognition System to Command and Control Computer: an Experiment with Microsoft Word. Master's Thesis, Addis Ababa University, Addis Ababa.

Mohamad, A. A., Lina, A. K., Jimmy, A., Elias, Y., (2008). Speech Recognition Using Artificial Neural Networks and Hidden Markov Models, IMCL2008 Conference, Beirut, Lebanon.

Motomichi, W., (2008). A Descriptive Study of the Modern Wolaytta Language. Doctoral Dissertation, University of Tokyo.

Nahm, E., Slater, D., (1997). Speech Recognition. The Ultimate Multimedia Handbook. Ed. Keyes, Jessica, New York: McGraw-Hill.

Natarajan, A. M., Thangarajan, R., (2009). A Robust Front-End Processor combining Mel Frequency Cepstral Coefficient and Sub-band Spectral Centroid Histogram methods for Automatic Speech Recognition. International Journal of Signal Processing, Image Processing and Pattern Recognition, Vol. 2, No. 2.

Nebiyou, Tsegaye, (2005). Speech to Text Conversion Using Amharic Characters. Master's Thesis, Addis Ababa University, Addis Ababa.

Nogueiras, A., Moreno, A., Bonafonte, A., Marino, J. B., (2001). Speech Emotion Recognition Using Hidden Markov Models. Research Centre TALP, Universitat Politècnica de Catalunya. Spain.

Philip, G., Young, E. S., (1987). Man-machine Interaction by voice: Developments in Speech Technology. Journal of Information Science, vol. 13. pp. 3-14.

Rabiner, L. R., (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, Vol. 77, No. 2.

Rabiner, L.R., Juang, B. H., (1993). Fundamentals of Speech Recognition. Englewood Cliffs, New Jersey: Prentice Hall, Inc.

Rodman, R. D., (1999). Computer Speech Technology. Norwood: Artech House, Inc.

Satori, H., Hiyassat, H., Harti, M., Chenfour, N., (2007). Investigation Arabic Speech Recognition Using CMU Sphinx System. Arab Academy for Banking and Financial Sciences, Amman, Jordan.

Solomon, T., Martha, Y., Menzel, W., (2008). Amharic Speech Recognition: Past, Present, and Future. University of Hamburg, Germany.

Tewodros Abebe, (2009). "Text-to-Speech Synthesizer for Wolaytta Language" Master Thesis, Addis Ababa University, Addis Ababa.

Walker, W., Lamere, P., Kwok, P., Raj, B., Singh, R., Gouvea, E., Wolf, P., Woelfel, J., (2004). Sphinx-4: A Flexible Open Source Framework for Speech Recognition. Sun Microsystems Inc.

Waleed, H. A., Nikola, K. K., (1999). The Concepts of Hidden Markov Model in Speech Recognition. Knowledge Engineering Lab, University of Otago, New Zealand.

Young, S., (1996). Large Vocabulary Continuous Speech Recognition: a Review. Technical Report. Cambridge University Engineering Department.

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P., (2002). The HTK Book. Microsoft Corporation

Zegaye Seifu, (2003). Hidden Markov Model Based Large Vocabulary, Speaker Independent, Continuous Amharic Speech Recognition. Master's Thesis, Addis Ababa University, Addis Ababa.

Zue, V., Cole, R., (1996). Speech Recognition. In Survey of the State of the art in Human Language Technology. Edited by Ronald A. Cole, Joseph Mariani, hans Uszkoreit, Annie Zaenen and Victor Zue.

Zue, V, Cole, R., (1995) Spoken Language Input: Overview. Survey of the State of the Art in Human Language Technology. Eds. Cole, R.A et al., Oregon Graduate Institute.

Appendix 1 – Wolaytta Character Set

Latiine			Latiine		
Qeeraa	**Woggaa**	**Saaba**	**Qeera**	**Woggaa**	**Saaba**
a	A	ዐ	r	R	ሬ
b	B	በ	s	S	ስ
c	C	ጭ	t	T	ት
d	D	ድ	u	U	ኡ
e	E	ኤ	v	V	ቭ
f	F	ፍ	w	W	ው
g	G	ግ	x	X	ጥ
h	H	ሀ	y	Y	ይ
i	I	እ	z	Z	ዝ
j	J	ጅ	ch	CH	ች
k	K	ክ	dh	DH	ደ
l	L	ል	ny	NY	ኝ
m	M	ም	ph	PH	ጵ
n	N	ን	sh	SH	ሽ
o	O	ኦ	ts	TS	ጽ
p	P	ፕ	zh	ZH	ዠ
q	Q	ቅ		7	ኧ

Appendix 2 – Sample excerpt of the recorded words to build the recognizer

7AADHDHIIS	7ECERIYA	7EKKAABE7IKKII	7IXXIIS
7UPAITTAAS	AGGADII	AIMALA	AISSI
7ANJIIS	BIITTAA	CARKUWA	DABBO
DARO	DE7II	DEREQQIIS	GAADA
GALATAIS	GAMM7ADII	GIIGISSADA	HACHCHI
KAWOTETTAA	KEEHIN	KEETTAA	LO77OO
LO77OTETTAA	MAAZHIIS	MOORETTIN	OOSOI
SARO	SAROTETTAA	SHAAKKIIS	SHEESHSHAA
SIIQO	SIQUWANNE	SUIQIIS	TAASSI
TAMMA	WURSIKKO	XOOSSAI	YAANA

7 – is a voiceless stop whose Amharic (Saaba) equivalent is represented in Appendix 1.

Appendix 3 – An excerpt of the unigram language model used for testing the recognizers (wolaytta.unigram.lm)

```
Language model created by QuickLM on Fri Apr 30 03:57:29 EDT 2010
Carnegie Mellon University (c) 1996

This model based on a corpus of 10 sentences and 90 words
The (fixed) discount mass is 0.5
This file is in the ARPA-standard format introduced by Doug Paul.

p(wd3|wd1,wd2)= if(trigram exists)           p_3(wd1,wd2,wd3)
                else if(bigram w1,w2 exists) bo_wt_2(w1,w2)*p(wd3|wd2)
                else                         p(wd3|w2)

p(wd2|wd1)= if(bigram exists) p_2(wd1,wd2)
            else              bo_wt_1(wd1)*p_1(wd2)

All probs and back-off weights (bo_wt) are given in log10 form.

Data formats:

Beginning of data mark: \data\
ngram 1=nr            # number of 1-grams
ngram 2=nr            # number of 2-grams

\1-grams:
p_1     wd_1 bo_wt_1
\2-grams:
p_2     wd_1 wd_2

end of data mark: \end\

\data\
ngram 1=102

\1-grams:
-3.2553 @AADHDHIIS -0.2218
-3.2553 @AAPPUN -0.2218
-3.2553 @ACUWA -0.2218
-3.2553 @ADUSSA -0.2218
-3.2553 @AKEEKADA -0.2218
-3.2553 @AKEEKIIS -0.2218
-3.2553 @ANJIIS -0.2218
-3.2553 @APALAA -0.2218
-3.2553 @ASHUWA -0.2218
-3.2553 @AZNAA -0.2218
.....................
-3.2553 @EKKITE -0.2218
-3.2553 @EPUWA -0.2218
-3.2553 @ERIBE@ENNA -0.2218
```

```
-3.2553 @IMATTA -0.2218
-3.2553 @IMMO -0.2218
-3.2553 @IRXA -0.2218
-3.2553 @ISHAA -0.2218
-3.2553 @ISHATAMMU -0.2218
-3.2553 @UPAITTAAS -0.2218
-3.2553 @URQAA -0.2218
-3.2553 @USHACHCHA -0.2218
-3.2553 @UTTAAS -0.2218
-0.7782 </s> -0.3010
-0.7782 <s> -0.2218
-3.2553 AGGADII -0.2218
-3.2553 AIMALA -0.2218
-3.2553 AISSI -0.2218
-3.2553 ASAI -0.2218
-3.2553 BAAWA -0.2218
-3.2553 BADALAA -0.2218
-3.2553 BADALAI -0.2218
.....................
-3.2553 BE@AAS -0.2218
-3.2553 BEETTENNA -0.2218
-3.2553 BIDINTA -0.2218
-3.2553 BIITTAA -0.2218
-3.2553 BITANEE -0.2218
-3.2553 BOCETTAA -0.2218
-3.2553 BOLLAA -0.2218
-3.2553 BONEE -0.2218
-3.2553 BOORAA -0.2218
-3.2553 BOOTTAA -0.2218
-3.2553 BULLUKKUWA -0.2218
-3.2553 CARKUWA -0.2218
-3.2553 CASHSHA -0.2218
-3.2553 CEEGA -0.2218
-3.2553 CIMMAI -0.2218
-3.2553 CORA -0.2218
-3.2553 DAAPURIDI -0.2218
-3.2553 DABBO -0.2218
-3.2553 DABBUWA -0.2218
-3.2553 DANDAYAKKA -0.2218
-3.2553 DANDAYEES -0.2218
-3.2553 DANDAYENNA -0.2218
-3.2553 DARO -0.2218
-3.2553 DE@II -0.2218
-3.2553 DE@OOSONA -0.2218
-3.2553 DEEXO -0.2218
-3.2553 DEREE -0.2218
-3.2553 DEREQQIIS -0.2218
-3.2553 DEWUZZA -0.2218
-3.2553 DICCAAS -0.2218
-3.2553 DICCIIS -0.2218
-3.2553 DOGOPPA -0.2218
-3.2553 DONOI -0.2218
-3.2553 DORSA -0.2218
-3.2553 DOSAIS -0.2218
-3.2553 DOSIKKE -0.2218
-3.2553 DOZHZHUGIIS -0.2218
-3.2553 DUGE -0.2218
```

```
-3.2553 DUMMA -0.2218
-3.2553 DUREE -0.2218
-3.2553 DURSAA -0.2218
-3.2553 GAADA -0.2218
-3.2553 GACINUWA -0.2218
-3.2553 GAITTANAU -0.2218
-3.2553 GAKKANAU -0.2218
-3.2553 GALATAIS -0.2218
-3.2553 GALLA -0.2218
-3.2553 GALLASSA -0.2218
-3.2553 GAMM@ADII -0.2218
-3.2553 GANJEE -0.2218
-3.2553 GARAWAA -0.2218
-3.2553 GAXAA -0.2218
-3.2553 GELIDA -0.2218
-3.2553 GELIDI -0.2218
-3.2553 GIDANA -0.2218
-3.2553 GIIGISSADA -0.2218
-3.2553 GIYAA -0.2218
.....................
-3.2553 HAASAYA -0.2218
-3.2553 HAATTA -0.2218
-3.2553 HAATTAA -0.2218
-3.2553 HAAYA -0.2218
-3.2553 HACHCHI -0.2218
-3.2553 HADDIRSA -0.2218
-3.2553 HAGAAPPE -0.2218
-3.2553 HAIQUWA -0.2218
-3.2553 HAMMA -0.2218
-3.2553 HARBAINNO -0.2218
-3.2553 HARGIYA -0.2218
-3.2553 HASTAMA -0.2218
-3.2553 HEEZZANTA -0.2218
-3.2553 HEMETTAIDDA -0.2218
-3.2553 HEXXISHIIS -0.2218
-3.2553 HIRAAGA -0.2218
-3.2553 HIRAISIIS -0.2218
-3.2553 HUUPHIYA -0.2218
-3.2553 KAALLANA -0.2218
-3.2553 KAISO -0.2218
-3.2553 KALTAA -0.2218
-3.2553 KANAA -0.2218
-3.2553 KANTAAS -0.2218
.....................
-3.2553 KIPILIYA -0.2218
-3.2553 KOIRO -0.2218
-3.2553 KUIXAARUWA -0.2218
-3.2553 MIICCIIS -0.2218
-3.2553 ZARPHIYA -0.2218

\end\
```

- For the purpose of preparing the dictionary and the language model for this research, the voiceless stop 7 was substituted by the special character @.