ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
FACULTY OF TECHNOLOGY

SPEECH TO TEXT CONVERSION USING AMHARIC CHARACTERS

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT
FOR
THE DEGREE OF MASTERS OF SCIENCE IN COMMUNICATION ENGINEERING

BY

NEBIYOU TSEGAYE

DECEMBER 2005

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
FACULTY OF TECHNOLOGY
DEPARTMENT OF COMMUNICATION ENGINEERING

SPEECH TO TEXT CONVERSION USING AMHARIC CHARACTERS

BY

NEBIYOU TSEGAYE

Signature of board of examiners for approval

Advisor: Dr. Enyew Adugna _____

Internal Examiner: Dr. Ing Hailu Ayele _____

External Examiner: Ato Molalgn _____

Dean: Dr. Getachew _____

ACKNOWLEDGMENTS

I would like to thank Dr. Enyew Adugna for his support and motivation through the course of my thesis and for his proof reading of this material.

I would also like to thank my mother Yalemtsehai Shifferaw who encouraged me in a number of ways.

I am also thankful for my colleagues and friends in the communication Engineering postgraduate program.

My appreciation to General Manager Mesay Zegeye of "Concepts Data Systems" for allowing me to learn while working in his company.

Last but not least, I would like to thank all the staff of Electrical and Computer engineering department especially Ato Sintayehu and Ato kiros for their cooperation through out my thesis work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVATIONS

DFT	**Discrete Fourier Transform**
HMM	**Hidden Markov Model**
LPC	**Linear Prediction Coefficient**
MATLAB	**Matrix Laboratory**
MFCC	**Mel-Frequency Cepstral Coefficient**
STFT	**Short-Term Fourier Transform**
STT	**Speech To Text (conversion)**
TTS	**Text To Speech (conversion)**

ABSTRACT

Spoken language is the primary method of human to human communication. This communication by spoken language is now extended by use of technologies such as telephony, radio, etc. These technological advancements reflect that spoken communication is the preferred method in human psychology.

Spoken language is also a preferred method of human-machine interaction. A spoken language system needs to have both speech recognition and speech synthesis capabilities. But this thesis is about building only the speech recognition (Speech to Text) system, specifically for Amharic language. Amharic language has more than 200 characters but the standard keyboard is made for English alphabet. This limited number of keys has imposed the need of 2 – 4 key strokes to write a single Amharic letter.

The practical project of this thesis is to develop functional software with speech to text capabilities for Amharic language. But this software by no means covers all Ethiopic characters. The algorithms and models developed will be experimented on small part of the Ethiopic characters with minimal error rate as possible.

There are different approaches to speech recognition. But the statistical approach to speech recognition seems to be industries current favorite, as it delivers better performance. It is also easier to implement. So the statistical approach is used in the development of the software. This approach requires acoustic models and language models to be built. Acoustic model refer to representation of knowledge about acoustics, phonetics, etc whereas Language

model refers to system knowledge of what constitutes a possible word, what words likely to co-occur and in what sequence.

This thesis is an attempt to build STT conversion for Amharic language using the statistical approach. So the inventory of speech files is made by recording and from these data appropriate models are built. The purpose is to test the performance based on the models built and prove that statistical models are suited to modeling speech signals.

Chapter One

Introduction

1.1 Introduction to speech recognition

1.1.1 Background to the study

With the advances of technology, a lot of people may think that integrating the ability of understanding human speech in a computer system is easy. However, scientists disagree. Since the early nineteen fifties, scientists have tried to implement the perfect automatic speech recognition system, but they failed. They were successful in making the computer recognize a large number of words, but till now, a computer that understands everything without meeting any preconditions does not exist. Due to the enormous applications, a lot of money and time is spent in improving speech recognition systems.

1.1.2 Speech Recognition: Principles and Applications

Nowadays, computer systems play a major role in our lives. They are used everywhere beginning with homes, offices, restaurants, gas stations, and so on. Nonetheless, for some, computers still represent the machine they will never know how to use. Communicating with a computer is done using a keyboard or a mouse, devices many people are not comfortable using. Speech recognition solves this problem and destroys the boundaries between humans and computers. Using a computer will be as easy as talking with your friend. Unfortunately, scientists have discovered that implementing a perfect speech recognition system is no easy task. This chapter will present the principles and the major approaches to speech recognition systems along with some of their applications.

1.2 Overview of the Characteristics of Speech Recognition Systems

The above sections described the general goals of the speech recognition task and generally suggested the major problems involved. In this section, we wish to more formally discuss various problems in speech recognition. We address the question of what factors influence the success or

failure of a speech recognition system and dictate the degree or sophistication necessary in the design of the system. These factors are enumerated as answers to the following questions:

1. Is the system required to recognize a specific individual or multiple speakers? (Including, perhaps, all speakers)?
2. What is the size of the vocabulary?
3. Is the speech to be entered in discrete units (usually words) with distinct pauses among them (discrete utterance recognition), or as a continuous utterance (continuous speech recognition).
4. Is the system to be operated in a quiet or noisy environment, and what is the nature of the environmental noise if it exists?
5. What are the linguistic constraints placed upon the speech, and what linguistic knowledge is built into the recognizer?

We consider each of these questions sequentially in the following subsections.

1.2.1 Speaker-Dependent versus Speaker-Independent Recognition

Most speech recognition algorithms, in principle, can be used in either a "speaker-dependent" or "speaker-independent" mode, and the designation for a particular system depends upon the mode of training.

A *speaker-dependent* recognizer uses the utterances of a single speaker to learn the parameters (or models) that characterize the system's internal model of the speech process. The system is then used specifically for recognizing the speech of its trainer. Accordingly, the recognizer will yield relatively high recognition results compared with a *speaker-independent* recognizer, which is trained by multiple speakers and used to recognize many speakers (who may be outside of the training population). Although more accurate, the apparent disadvantage of a speaker-dependent system is the need to retrain the system each time it is to be used with a new speaker.

Beyond the accuracy/convenience trade-off is the issue of necessity. A telephone system that must respond to inquiries from the public must necessarily be speaker-independent. Both types of systems

(speaker dependent and independent), are used in practice, and both have been studied and tested extensively in the Literature and laboratory.

Before continuing, let us note that some authors distinguish between speaker-independent systems for which the training populations are the same as the users, and those for which the training populations are different from the users. In the former case, the term "*multiple speaker"* is used while the term "*speaker independent*" is reserved for the latter. It is important to take note of this issue in comparing the performance of various systems.

1.2.2 Vocabulary Size

Clearly, we would expect performance and speed of a particular recognizer to degrade with increasing vocabulary size. As a rule of thumb, some speech researchers estimate that the difficulty of the recognition problem increases logarithmically with the size of the vocabulary. Memory requirements also increase with increasing vocabulary size.

Speech recognition systems or algorithms are generally classified as small, medium, or large vocabulary. There is some variation in the literature on the quantification of these terms, but as a rule of thumb, *small vocabulary* systems are those which have vocabulary sizes in the range of 1-99 words; *medium*, 100-999 words; and *large*, 1000 words or more.

Small-vocabulary (as defined here) systems have become routinely available and have been used in tasks such as credit card or telephone number recognition. The focus of the medium-sized vocabulary systems has been experimental laboratory systems for continuous-speech recognition research (driven in part by the availability of standardized databases). Large-vocabulary systems have been used for commercial products currently aimed at such applications as dictation. These systems have been of the isolated-word type in which the speaker must utter each word discretely from the others. It is important to keep in mind that a given-size vocabulary can require far more effort for a speaker-independent system than a speaker-dependent one. Continuous-speech recognition is also much more difficult than discrete utterance recognition; thus, vocabulary size is only one measure of difficulty.

For small vocabularies and relatively constrained tasks (e.g., recognizing numerical strings), simple discrete word utterance recognition strategies can often be employed. In these cases, models for each word in the vocabulary are resident in the system and the list can be exhaustively searched

for each word to be recognized. As vocabularies become larger and recognition tasks more complicated, training and storing models for each word is generally impossible and models for subword units (e.g., syllables, phonemes) are employed. Simple exhaustive search of all possible messages (built from these subword units) also becomes unmanageable and much more sophisticated search algorithms that decreases the number of items searched must be designed. Significant to these algorithms are "linguistic constraints" on the search that eliminate unmeaningful and grammatically incorrect constructions. Also complicating the recognition task as vocabularies become larger is the potential for an increased number of confusable items in the vocabulary.

1.2.3 Isolated-Word versus Continuous-Speech Recognition

From now on in this discussion, we will use the term sentence to mean any string of words to be recognized that is presumably taken from the vocabulary under consideration. The "sentence" can be what we would ordinarily think of as a grammatically correct sentence as in day to day communication between people or, a simple string of digits, or even, in the "degenerate case," a single word.

1.2.3.1 Isolated-Word Recognition.

Discrete-utterance recognizers are trained with discrete renditions of speech units. Since the discrete utterances are usually words, this form of speech recognition is usually called *isolated word recognition* (IWR). In the recognition phase, it is assumed that the speaker deliberately utters sentences with sufficiently long pauses between words (typically, a minimum of 200 msec is required) so that silences are not confused with weak fricatives (like /s/ sound) and gaps in plosives (like /d/ sounds). Single-word sentences are special cases with "infinite" pauses. The fact that boundaries between words can be located significantly simplifies the speech recognition task. These boundaries are located in various technical ways, including the use of an endpoint detection algorithm to mark the beginning and end of a word. This is the simplest form of recognition strategy, and it requires a cooperative speaker. It is nevertheless very suitable for certain applications, particularly those in which single-word commands from a small vocabulary are issued to a machine at "lengthy" intervals. A good example of an application with such intervals arises in the *sorting machine* application, in which the operator utters a destination as each package presents itself on a. conveyor.

When the vocabulary size is large, isolated-word recognizers need to be specially constructed and trained using subword models. Further, if sentences composed of isolated words are to be recognized, the performance can be enhanced by exploiting probabilistic (or simply ordering) relationships among words ("syntactic" knowledge) in the sentences.

1.2.3.2 Continuous-Speech Recognition.

The most complex recognition systems are those that perform *continuous-speech recognition* (CSR), in which the user utters the message in a relatively (or completely) unconstrained manner. First, the recognizer must be capable of somehow dealing with unknown word boundaries in the acoustic signal. Second, the recognizer must be capable of performing well in the presence of all the coarticulatory effects that accompany flowing speech. Whereas the CSR problem does not in the extreme case require any cooperation from the speaker, it must compensate for this fact by employing algorithms that are robust to the coarticulation effects of flowing speech. CSR systems are the most natural from the user's point of view.

In large-vocabulary CSR speech systems, the same two considerations as in the IWR case apply. Words must be trained as subword units, and interword relationships (syntax) must be exploited for good performance.

1.2.4 Linguistic Constraints

Another problem involved in speech recognition is endowing the recognizer with the appropriate "language constraints." Whether we view phones, syllables, or words as the basic unit of speech, *language* (or *linguistic) constraints* are generally concerned with how these fundamental units may be concatenated, in what order, in what context, and with what intended meaning. This problem is more involved than simply programming the correct grammatical rules for the language. Clearly, the more constrained the rules of language in the recognizer, the less freedom of expression the user has in constructing spoken messages. The challenge of language modeling is to balance the need for maximally constraining the "pathways" that messages may take in the recognizer, while minimizing the degree to which the speaker's freedom of expression is diminished.

Let us begin the consideration of language constraints by posing an abstract model of for the language: which are symbolic models, grammatical models, and semantic models of the language.

The *Symbols* of a language are defined to be the most fundamental units from which all messages are ultimately composed. In the spoken form of a language, for example, the symbols might be words or phonemes, whereas in the written form, the alphabet of the language might serve as the symbols.

The *Grammar* of the language is concerned with how symbols are related to one another to form ultimate message units. If we consider the sentence to be the ultimate message unit, and we choose phonemes as symbols, then how words are formed from phonemes is properly considered as part of grammar, as well as the manner in which words form sentences. How phonemes form words is governed by *lexical* constraints, and how words form sentences by *syntactic* constraints. Lexical and syntactic constraints are both components of the grammar.

The *Semantics* of the language is concerned with the way in which symbols are combined to form meaningful sentence.

Let us view the following "sentences" to compare with our definition of grammar. Next to each sentence is a description of its conformity to the linguistic concepts discussed above.

1. Colorless paper packages crackle loudly. [Grammatically correct]
2. Colorless yellow ideas sleep furiously. [Grammatically correct, semantically incorrect]
3. Sleep roses dangerously young colorless. [Grammatically (syntactically) incorrect]
4. Ben burada ne yaptigimi bilmiyorum.1o [grammatically (lexically) incorrect]

1.2.5 Block diagram of general speech recognizer

A typical speech recognition system consists of basic components shown in fig. 1.1. Acoustic knowledge includes representation of knowledge about acoustics, phonetics, and environment variability. Language Model refer to a system knowledge of what constitutes a possible word, what words are likely to co-occur, and in what sequence.

The speech signal is processed in the signal processing module that extracts salient feature vectors for the decoder. The decoder uses both acoustic and language models to generate the word sequence that has the maximum posterior probability for the input feature vectors.

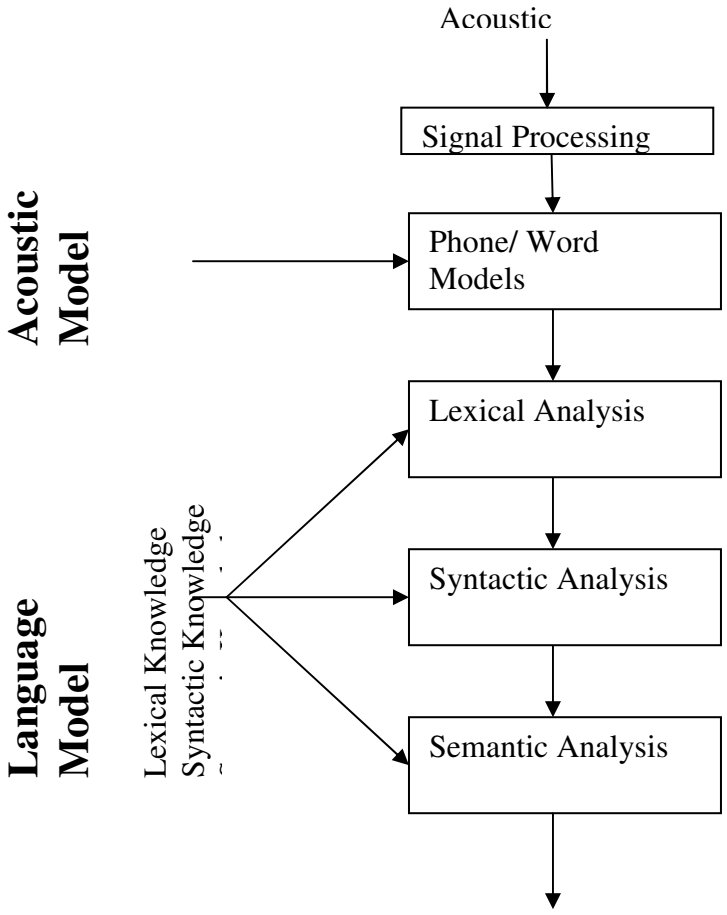


Figure 1.1 represents a general flow diagram of speech recognizers.

1.3 Different Approaches to speech recognition

- Acoustic-Phonetic Approach

The theory behind acoustic-phonetic approach is acoustic phonetics (to be described in chapter two). This theory assumes that spoken language is divided into phonetic units that are finite and particular. For example, the speech units in Amharic language are ሀ, ለ, ሐ. . . . etc. These phonetic units are distinguished by properties that are apparent in the speech signal. The process by which speech is recognized is described briefly in what follows:

Initially, speech is divided into segments according to some segmentation algorithm. Then depending on the acoustic properties of these segments, an appropriate phonetic unit is attached to it. The obtained sequence of units is used to formulate a valid word.

The acoustic phonetic approach is an earlier attempt to speech recognition. This approach does not take context dependency into consideration. Context dependency results from Coarticulation effects (to be described in chapter two).

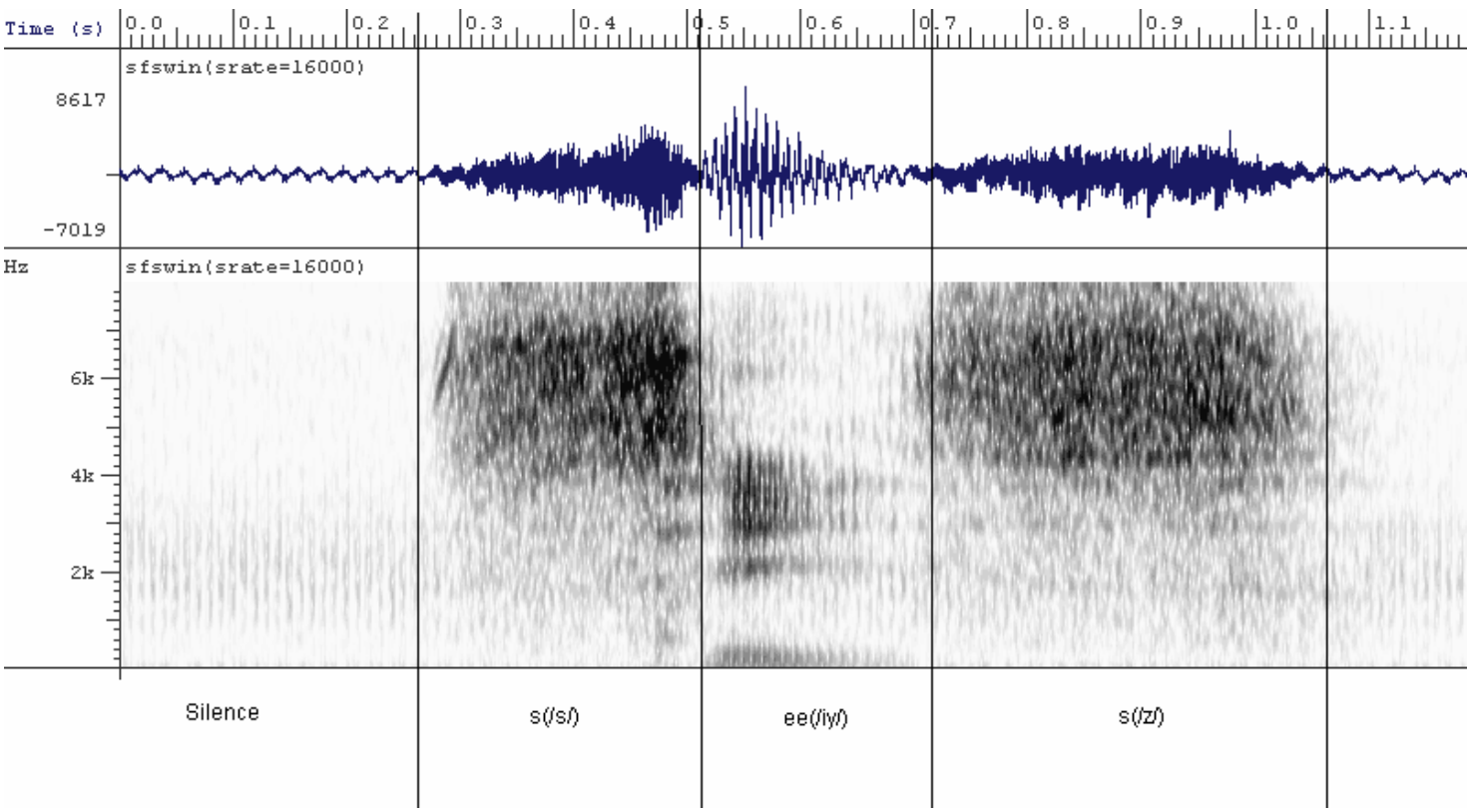


Figure 1.2: Speech waveform and spectrogram representation of the word “sees”.

In this approach, acoustic features (e.g. spectral features) of a phone or syllable are found manually studying the characteristics of each phone.

- Statistical Pattern Recognition Approach

In statistical pattern recognition, the speech patterns are directly inputted into the system and compared with the patterns inputted in the system during training. Unlike the acoustic-phonetic

approach, the speech is not segmented nor checked for its properties. If enough patterns are inputted to the speech recognition system during training, it will perform better than the acoustic-phonetic approach. In general, statistical pattern recognition approach is used more than acoustic-phonetic approach because it is simpler to use, invariant to different speech vocabularies, and more accurate (higher performance).

- Neural Network Approach

Neural Networks have been shown to yield good performance for small vocabulary 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.

1.4 Applications of Speech Recognition Systems

With all the time and money spent on researches on speech recognition systems, someone may wonder about the applications of speech recognition. This part will present some of the currently available applications along with some future applications of automatic speech recognition systems.

- Automated Call-Type Recognition

An interesting and relatively simple application of speech recognition systems is automated call-type recognition. In pay phones, operators are needed to determine the call-type of the caller. Speech recognition may be used instead of operators. Five types of calls are available: '*collect*', '*calling card'*, '*operator*' for operator assisted calls, '*third number*' for third party billing calls, '*person*' for person-to-person calls. For this application, the speech recognition system must be speaker independent and capable of recognizing and spotting the five key words mentioned above in a speech sample. The problem in this application is the high amount of background noise since pay phones are usually available in public places; however, this problem can be solved using appropriate speech recognition systems.

- Data Entry

Entering data using speech recognition is very practical when performing a manual task. A speech recognition system for this application is highly complex and structured since it should contain a large vocabulary. For data entry, speaker-dependent or speaker-independent speech recognition systems are available even though speaker-dependent systems perform better than speaker-independent systems. They are also available for discrete or continuous speech. Data entry applications are still limited since the performance of speech recognition systems in this field is still limited.

- Future applications

Using automatic speech recognition systems with the increasing performance of automatic speech recognition systems, companies are more interested in integrating speech recognition systems in their products. Car manufacturers are interested in replacing all the levers, knobs, and buttons by a speech recognition system capable of doing everything, from raising temperature to locking doors and turning on the radio. In this way, the electronic content of the car is increased whereas the mechanical is reduced. This makes the car easier to design and build, therefore costing less.

Others think of applying speech recognition systems in kitchen appliances such as dishwashers, ovens, refrigerators. Air-conditioners might some day be voice controlled.

The gradual but inevitable development of speech recognition systems will surely lead to a system that will one day compare to the perfect speech recognition device, the human being. New methods and algorithms are researched every day to improve the performance of speech recognition systems. We might reach a stage where keyboards, buttons, and all input devices become obsolete.

1.5 Objective of the study

The objectives of the study are the following. The main objective of this research is to develop a prototype speech to text conversion for Amharic Language using statistical approach to the speech recognition problem. At the core of the statistical approach to speech recognition is the Hidden

Markov Model (HMM). This model must be built and trained and at last tested to see the performance. The specific objectives of the study are the following.

The specific objectives of the study are to:

- Select a set of characters from all the Ge'ez letters so that the project would be successful (give useful result) in time. The selected characters are shown in the figure 1.3 below.
- Create a word list that constitutes the above characters.
- Collect (record) a list of speech data to the selected characters or words.
- Develop algorithms/model for building a speech to text (STT) for Amharic language.
- Test the system on how it performs for different numbers of words in the model.
- Draw useful conclusion and forward recommendation for further study.

.

The enclosed characters and words constituting this characters are selected for

Figure 1.3. The 28 selected characters from the Ge'ez alphabet

1.6 Methods

The following methods will be employed while developing the speech recognizer.

-Review of related literature

A number of resources such as books, research reports, articles in journals and other published and unpublished documents have been used

- to understand speech, speech production.
- to understand phonology and linguistics.
- to examine and select types of speech, and acoustic units to be modeled.
- to examine appropriate model for speech production system(vocal tract).
- to compare and contrast speech signal representations.
- to identify and study statistical speech recognition models.
- to identify appropriate tools required to develop a prototype.

-Development tools and techniques

A statistical approach using HMM is used following its success in speech recognition systems.

Microsoft Visual C++ 6 programming language is used because of its simplicity and my awareness to the language. With Visual C++, various tools, libraries, software components can be easily integrated into a code.

Speech processing C++ codes are available on the internet, which can be easily used in C++ projects.

Speech Filing system (SFS) software is used for visual spectrogram plotting and segmentation of speech waveforms into phones. With this software one can do the following Acquisition and replay, Waveform processing, Filtering, Signal editing, Spectrographic analysis, Resampling and speed/pitch changing etc…

-Testing technique

It is critical to evaluate the performance of speech recognition systems. The word recognition error rate is widely used as one of the most important measures. When you compare different acoustic modeling algorithms, it is important to compare their relative error rate reduction.

Empirically, it is required to have abundant test data set to reliably estimate the recognition error rate. A test data set should be a new data of recorded utterance that is never used in training.

There are three types of word recognition errors in speech recognition.

1. Substitution: an incorrect word have been substituted for an incorrect word
2. Deletion: a correct word was emitted in a recognized sentence
3. Insertion: an extra word was added in the recognized sentence

Example

Correct: - Did you see the Copeland café on the nineteen eighty one street.

Recognized: - Did you see ** **Copy** land café in the nineteen **east** one street.

Where substitutions are **bold,** insertions are underlined, and the deletions are denoted as **.

$$\text{Word error rate} = \frac{\text{Subs} + \text{Dels} + \text{Ins}}{\text{No. of words in the correct sentence}}$$

For isolated word recognition only Substitution errors are considered.

1.7 Scope and Limitation of the study

Statistical approach to speech recognition requires a large amount of recorded speech data (Corpus Data). Building the corpus data is time consuming task. So, only a limited number of speech database is built for the selected part of Ge'ez characters. This would limit the number of words and limit the application to be isolated word recognition instead of continuous speech recognition. The utterance of words is assumed to be separated by silence. Chapter four deals with the practical application developed in detail.

Chapter Two

Speech and Language

2.1 Introduction

Language is the ability to express one's thoughts by means of a set of signs, whether graphical (as with the Roman, Ethiopic, or Arabic alphabets, or the Chinese ideograms), gestural (like the sign language for the deaf-mute), or acoustic (as with speech; one often refers to acoustic gestures). It is a distinctive feature of human beings, who are the only creatures to use such a structured system. Speech is one of its main components. It is by far the oldest means of communication between people and it is also the most widely used. People have extensively studied it and often tried to build machines to handle it in an automatic way.

2.2 Speech Production

2.2.1 Vocal fold vibration: physiology and acoustics

Indeed, speech can be described as the 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* (or *folds).* These are actually composed of two contiguous membranes, the tension of which is controlled by neighboring muscles (Fig. 2.1). They provide an aperture in the larynx, called the *glottis.* The air flows unimpeded through it during breathing,

whispered speech, and when producing *voiceless sounds*.

Voiced sounds on the contrary originate in the total obstruction of the larynx, which increases the air pressure up-stream and forces the vocal cords to open so as to release the air. The Bernoulli effect then causes a pressure drop, which suffices to close the glottis abruptly, and a cycle is completed, the duration of which depends on the tension of the vocal cords. It ensues that a glottal waveform composed of a sequence of pulses is fed through the vocal cavities-namely, the pharyngeal and oral cavities for most sounds. When the *uvula,* which is the tip of a soft tissue structure called the *velum,* is lowered, the nasal cavity is shunted with the oral one.

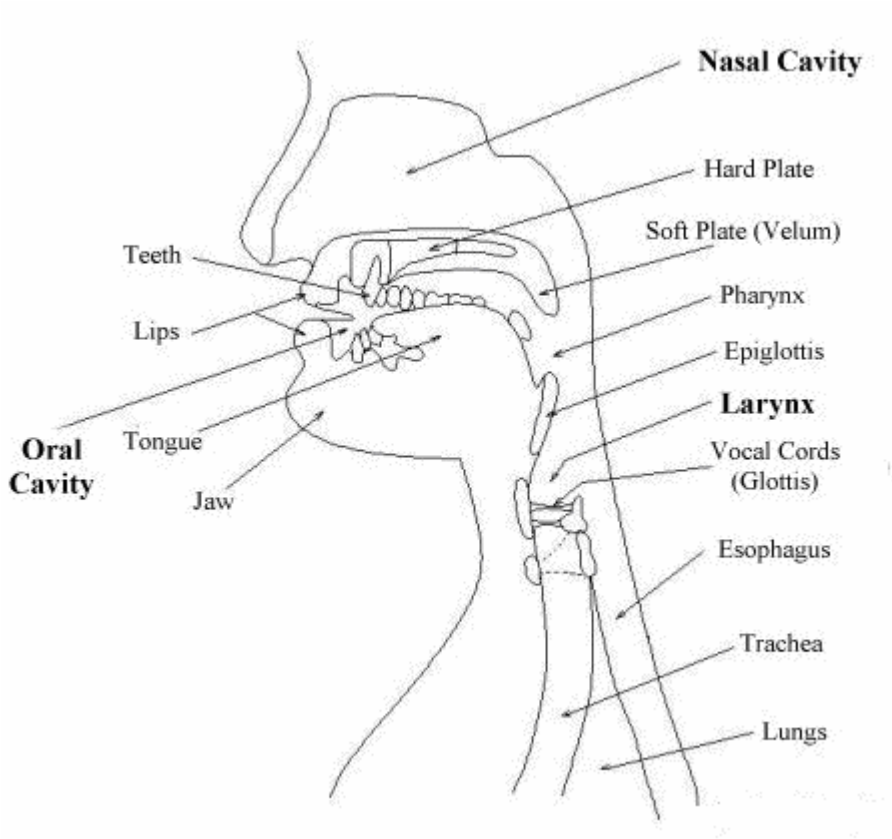


Figure 2.1: Schematic diagram of the human the vocal tract.

The acoustic traits (such as spectrum, energy etc) of a speech sound are naturally related to its production. Its intensity is related to the air pressure above the larynx. Its pitch, which is simply the frequency of the opening/closure cycle of the vocal folds, is additionally determined by the tension of the glottal muscles. Its timbre originates in the dynamic filtering of the glottal pulses through successive acoustic cavities.

2.2.2 What is speech made of?

The information conveyed by speech can be analyzed in many ways. Speech scientists generally distinguish several nonmutually exclusive levels of description-namely, the *phonetic, acoustic, phonological, morphological, syntactic,* and *semantic.*

2.2.2.1 The phonetic level

Phoneticians have a continued interest in studying the way speech signals are produced by the articulatory apparatus, presented in Fig. 2.1 and the way in which speech signals are represented acoustically.

There are two branches of phonetics. These are

1. Articulatory phonetics
2. Acoustic phonetics

2.2.2.1.1 Articulatory phonetics

It is convenient to group speech sounds into *broad phonetic classes,* related to their *manner* of articulation. In English, one generally distinguishes seven groups-namely, *vowels, fricatives, stops, nasals, affricates etc.*

Vowels differ from consonants in the degree of aperture of the vocal tract (and not, as one might think at first, in the level of activity of the vocal folds, already mentioned above and termed as voicing). If the vocal tract is open enough for the air pulsed by the lungs to flow without meeting any obstacle, a vowel is produced. The distinct vowel timbres are created by using the tongue and lips to shape the main oral resonance cavity in different ways. If, on the other hand, the path ever narrows or even temporarily closes, the airflow gives birth to a noise: a *consonant* is produced. The mouth then fully

becomes a speech production organ.

In the case of *nasals* [m, n, l], the uvula is lowered, and the nasal tract provides the main transmission channel.

Fricatives [s, z, v] originate in a constriction of the vocal tract, either at the glottis, at the hard palate, at the teeth, or at the lips. Voiceless fricatives are produced by a turbulent noise from the glottis; voiced ones combine periodic and aperiodic components: the vocal folds vibrate but they never really close.

Stops or *plosives* [b, d] are the most dynamic sounds. They are articulated in three steps: the vocal tract first closes at some point, which results in a build-up of pressure, the release of which produces a transient burst. Voiced plosives are distinguished from voiceless ones according to the presence or absence of vibration of the focal folds during the build-up.

It is also possible to group sounds as a function of their place of articulation: labial [m], dental [t], alveolar [s], palatal, velar [k], pharyngeal [h].

2.2.2.1.2 Acoustic Phonetics

Speech physically appears as a variation of the air pressure, caused and emitted by the articulatory system. Acoustic phoneticians study our speech by transforming it into an electrical signal with the adequate transducer: a microphone. In modern recording systems, the resulting electrical signal is then digitized: it is successively low-pass filtered, sampled, and quantized**.** It may then be submitted to various digital signal-processing operations, so as to highlight its acoustic traits: *fundamental frequency* (often denoted as F0), *intensity,* and *spectral energy distribution.* Each acoustic trait is itself related to a perceptual quantity: *pitch, loudness,* and *timbre.*

A simple computation of the *short-term Fourier transform* (STFT) uncovers its spectral contents. It is typically implemented as the Discrete Fourier Transform (DFT) of the digitized signal *x(n)* weighted with a finite length window *w(n)* (for example, 30 ms long), over which one can reasonably assume that the signal remains stationary given the inertia of the articulatory system).

Inside voiced portions; the fundamental frequency evolves slowly with time. It most often ranges from 70 to 200 Hz for men, from 150 to 400 Hz for women, and from 200 to 600 Hz for children. In contrast, the intensity of the signal may vary abruptly, even within voiced portions.

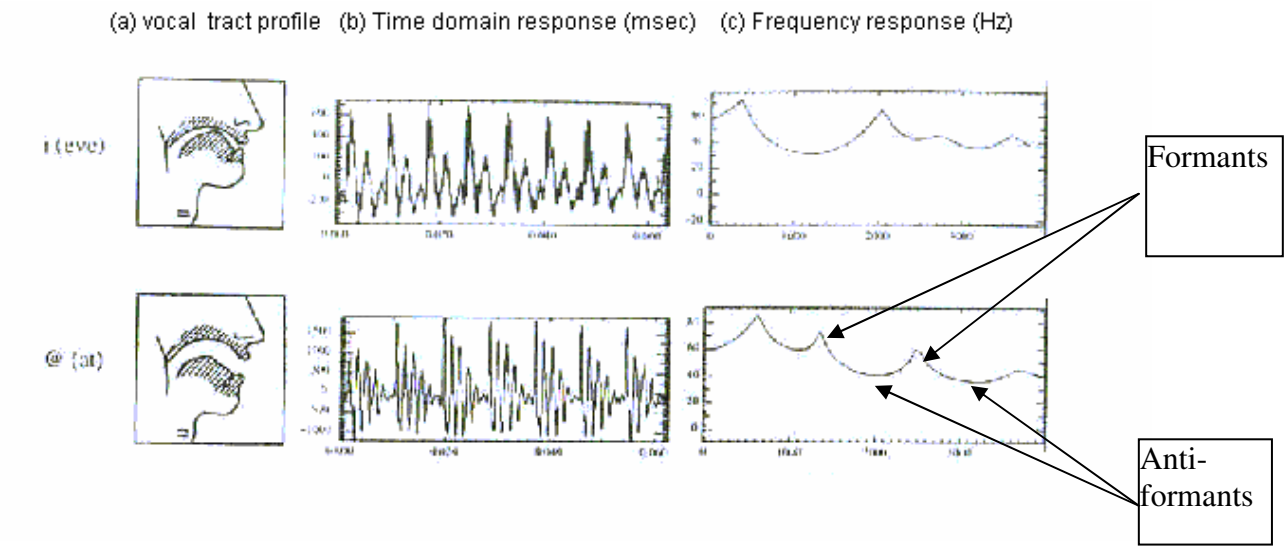


Figure 2.2 Collections of features for vowels. Column (a) schematic vocal tract profile, (b) typical acoustic waveforms, and (c) the corresponding vocal tract spectrum for each vowel.

Voiced parts appear as narrow spectral peaks, (See fig. 2.2) while the very nature of unvoiced spectra is stochastic. The overall spectral shape of both types of sounds, however, called the *spectral envelope,* exhibits broad peaks and valleys, called *formants* and *anti formants* (See fig. 2.2). The time variation of their center frequencies and bandwidths determine the *timbre* of the related sound. It turns out that most voiced sounds have a low-pass spectrum, with about one formant per kilohertz of bandwidth, of which only the first three or four have a real phonetic value. In contrast, unvoiced sounds generally appear as high-pass spectra (see fig. 2.3).

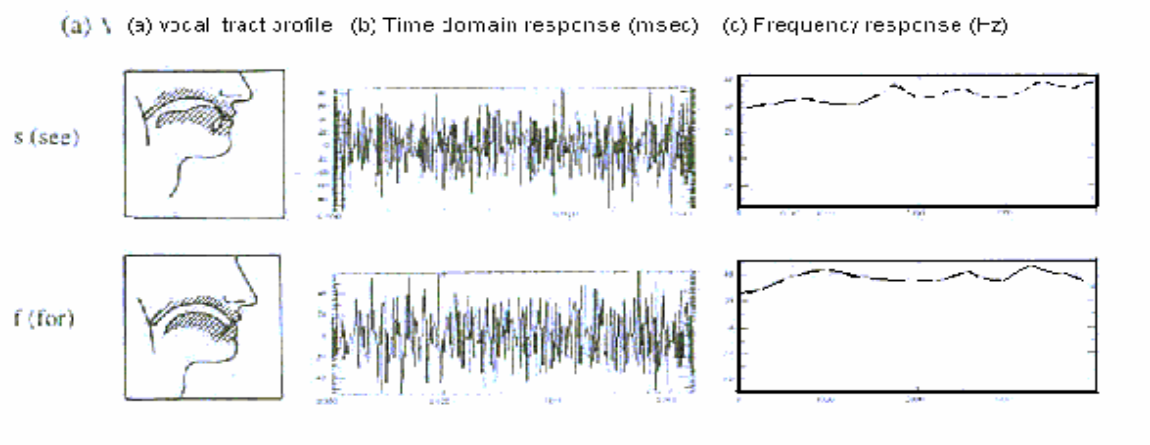


Figure 2.3 Collections of features for consonants. Column (a) schematic vocal tract profile, (b) typical acoustic waveforms, and (c) the corresponding vocal tract spectrum for each vowel.

The *spectrogram* gives a time-frequency representation of the signal by showing the time variation of

its short-term spectral amplitude as gray levels in a two-dimensional plot. Spectrograms are termed as narrow- or wide-band, depending the duration of the weighting window *w(n).* Wide-band spectrograms are obtained with short windows (< 10 ms) thus have a good time resolution at the expense of lower frequency resolution. Hence, the corresponding filters have wideband width (> 200 Hz). Narrow Band spectrograms use relatively long windows (> 20 ms) which lead to filters with narrow band width (< 100 Hz). On the other hand, time resolution is lower than wide-band spectrogram. Spectrograms can aid in determining formant frequencies and fundamental frequency, as well as voiced and unvoiced regions. Experts can even *read* spectrograms-that is, recover the utterance from its time-frequency representation (see fig 2.4).

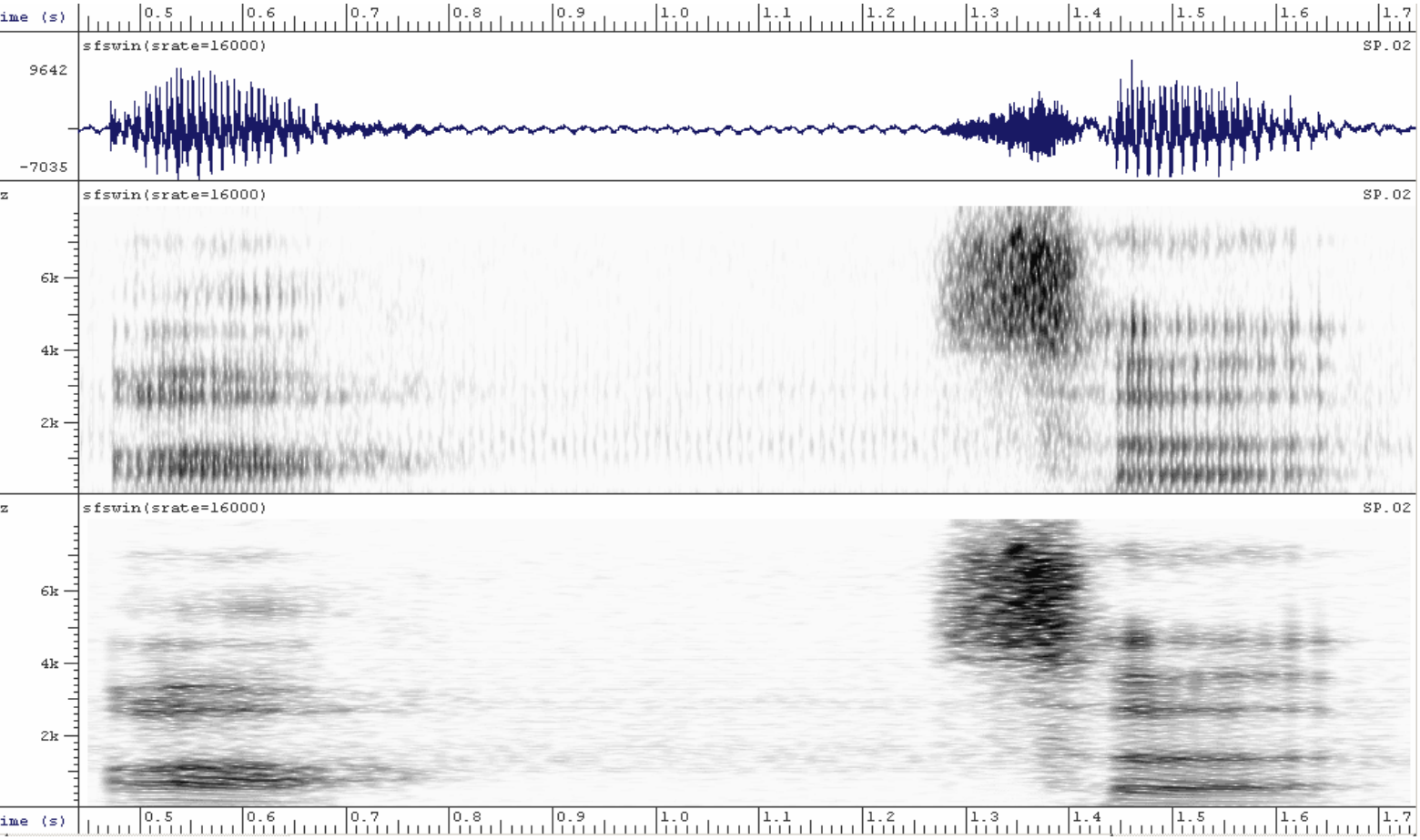


Figure 2.4 spectrogram for the utterance of Ge'ez characters › (ah) left and g (sh) right. Notice the low pass characteristic of the vowel › and the character g is consonant-vowel combination where the consonant is high pass and the vowel is low pass. The middle graph is wide band spectrogram while the bottom graph is narrow band spectrogram.

2.2.2.2 Phonological level

Phonology (sometimes called *functional phonetics)* is the necessary interface between phonetics and higher-level linguistics.

In the previous sections, we have described speech on an acoustic or physiological point of view as if it were not to convey any meaning. Speech sounds have therefore been presented independently. Phonology introduces abstract *linguistic* units called *phonemes.* A phoneme is the smallest meaningful contrastive unit in a language. It is thus neither defined on an acoustic nor physiological basis but on a *functional* one. Phonemes have no independent existence: they constitute a structured set of units in which each element is intentionally different from all the other ones. Besides, a list of the phonemes for languages is set up on the basis of the study of so-called *minimal pairs*, composed of pairs of words sharing all but one sound, which suffices to change their meaning (as for *put/pet).*

Phonetics could thus be better defined as the *study of the articulation of phonemes.* Phonemes can indeed appear in practical speech through a multitude of articulatory forms, called *phones.*

To clearly distinguish between phonemes and phones. We can think of a phoneme as an ideal sound unit with a corresponding set of articulatory gestures. Due to many factors including, for example, accents, gender, and most importantly coarticulatory effects, a given phoneme will have a variety of acoustic manifestations in the course of continuous flowing speech. Therefore, any acoustic utterance that is "clearly supposed to be" that ideal phoneme, would be labeled as that phoneme. Therefore, we see that from acoustical point of view, the phoneme clearly represents a *class* of sounds that convey the same meaning. The phonemes of a language, therefore, comprise a minimal theoretical set of units that are sufficient to convey all meanings in a language.

The actual sounds that are produced in speaking are called phones. The study of phonemes and phones and their relationship in language is called *Phonology.*

If a talker is asked to 'speak a phoneme' in isolation, the phoneme will be clearly identifiable in the acoustic waveform. However when spoken in context (in a word) a phoneme boundaries become increasingly difficult to label. This is due to the physical properties of speech articulators. Since the vocal tract articulators consist of human tissue, their positioning from one phoneme to another is not executed by hard mechanical switches, but by the movement of muscles that control articulator movement. Accordingly, there is a period of transition between phonemes, which under certain conditions can slightly modify the manner in which a phoneme is produced. These effects are known as *Coarticulation.* Therefore, associated with each phoneme is a collection of phones that represent

slight acoustic variations of the basic unit. Since no semantic information is conveyed such variations, do not give birth to new phonemes. There are schemes different ways for labeling phonemes. The International Phonetic Alphabet is one of standard of phonemes or symbols. The International Phonetic Alphabet (IPA) associates phonetic symbols to sounds, so that pronunciations can be written in a compact and universal way.

2.2.2.3 Morphological level

When studying a language, it is striking to notice the words it is made of, although very numerous, often share some of their spelling, as if they were formed from other smaller words or parts thereof (for example, *image. images. imagine. imagination. imagery. image-maker.* and so on). *Morphology* is the part of linguistics that describes word forms as a function of a reduced set of meaningful units, called *morpheme,* and subsequently separated into *stems* and *affixes* (themselves separated into *prefixes. infixes.* and *suffixes).*

One generally distinguishes *inflectional* morphology, which accounts for morphological features such as gender, number, mode, tense, or person *(image. images), derivational* morphology, which studies the construction of words from various syntactic categories from a common stem *(image. imagine. imagination. imagery)* and *compounding* morphology, whose task is to explain how two or more stems can be combined to form a new one *(image+make=image-maker).*

The pronunciation**,** the part of speech category, and the meaning of words, can be explained in terms of these morphemic components. Hence the importance of morphology in the context of natural language processing.

2.2.2.4 Syntactic level

Not all sequences of words from the lexicon of a language result in a correct sentence. Indeed, the list of permissible sentences, although infinite in natural languages, is restricted by their *syntax.* Most words lose their individuality when dealt with by grammatical rules: only their *part of speech category (nouns, pronouns, adjectives etc) is* examined. These are in turn defined as lists of words that are interchangeable for a given grammar.

2.2.2.5 Semantic level

Although syntax drastically restricts the set of well-formed sentences, it does not constitute an exhaustive criterion for acceptability. Many more sentences should be ruled out, simply because they have no meaning at all (as for *the yellow politeness cries bread).* This basically originates in the confusion that is intentionally made between words belonging to the same part of speech category. The study of word meanings, how they are related to one another, and the bases for lexical choice is the subject of *lexical semantics.*

2.3 Acoustic Models of Speech Production

The acoustic model analyzing the physics of the propagation of sound waves through the vocal tract should consider:

- three dimensional wave propagation
- variation of the vocal tract shape with time
- viscous friction at the walls,
- softness of the tract walls,
- radiation of sound at the lips,
- nasal coupling ,
- excitation of sound

A Model that considers all of the above is not yet available, but some models provide good approximation in practice and good understanding of physics involved. Examples are:

– Lossless Tube Concatenation

– Source Filter Models

2.3.1 Lossless Tube Concatenation Model

• Idea: The vocal tract can be represented as a concatenation of 'p' lossless tubes

• It consists of a series of cylinders (Helmholtz-Resonator) of equal length 'l'

• The cross-sections A_i approximate the area function A(x) of the vocal tract

• If 'p' is large, and 'l' is short, the frequency response is expected to be close to those of tubes with continuously varying area functions

• Further assume: No loss due to viscosity or thermal conduction, and area A remains constant over time

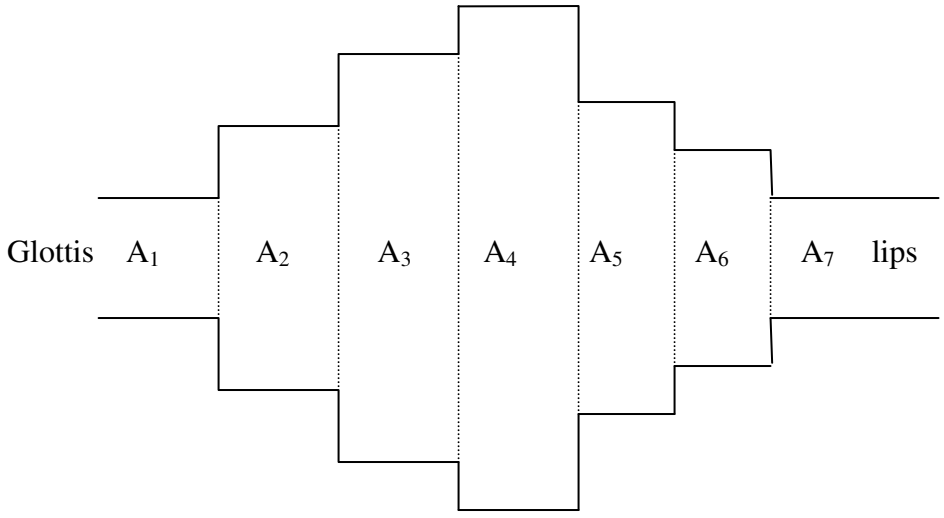


Figure 2.5 Lossless Tube Concatenation Model

The cross-sectional areas A_1 , A_2 , A_3 …. are chosen to approximate the time varying area function of the vocal tract. If appropriate number of tubes is chosen, the resonant frequencies of these

concatenated model of tubes can be made to approximate those of the vocal tract. It must be noticed that a stationery signal is to modeled so that time dependence of the area can be dropped.

2.3.2 The Source-Filter Model

• Sounds are produced by either vibrating the vocal cords (voiced sounds) or random noise resulting from friction of the airflow (unvoiced sounds). Voiced fricatives need a mixed excitation model.

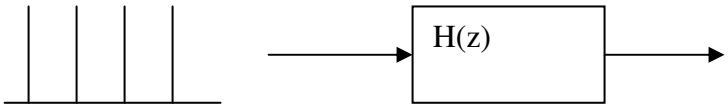


Figure 2.6 model of excitation for voiced sounds.

For voiced sounds we can model as impulse train convolved with filter transfer function.

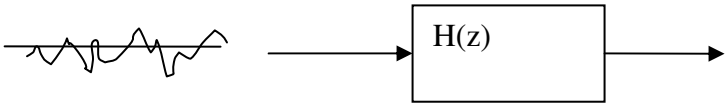


Figure 2.7 model of excitation for unvoiced sounds.

For unvoiced sounds we can model random noise convolved with filter transfer function.

2.4 Speech Signal Representation

In statistically based automatic speech recognition, the speech waveform is sampled at a rate between 8.0 kHz and 20 kHz and processed to produce a new representation as a sequence of vectors containing values of what are generally called *parameter vectors*. The vectors typically comprise between 10 and 20 parameters, and are usually computed every 10 to 30 msec. These parameter values are then used in succeeding stages in the estimation of the probability that the portion of

waveform just analyzed corresponds to a particular event that occurs in the phone-sized or whole-word reference unit being hypothesized for modeling.

Representations aim is to preserve the information needed to determine the phonetic identity of a portion of speech while being as impervious as possible to factors such as speaker differences, effects introduced by communications channels, and paralinguistic factors such as the emotional state of the speaker. They also aim to be as compact as possible.

Representations used in current speech recognizers, concentrate primarily on properties of the speech signal attributable to the shape of the vocal tract rather than to the excitation, whether generated by a vocal-tract constriction or by the larynx. Representations are sensitive to whether the vocal folds are vibrating or not (the voiced/unvoiced distinction), but try to ignore effects due to variations in their frequency of vibration.

Representations are almost always derived from the short-term power spectrum.

The power spectrum is, moreover, almost always represented on a log scale. When the gain applied to a signal varies, the shape of the log power spectrum is preserved; the spectrum is simply shifted up or down. More complicated linear filtering caused, for example, by room acoustics or by variations between telephone lines, which appear, as convolution effects on the waveform and as multiplicative effects on the *linear* power spectrum, become simply additive constants on the log power spectrum. Indeed, a voiced speech waveform amounts to the convolution of a quasi-periodic excitation signal and a time-varying filter determined largely by the configuration of the vocal tract. These two components are easier to separate in the log-power domain, where they are additive. Finally, the statistical distributions of log power spectra for speech have properties convenient for statistically based speech recognition that are not shared by linear power spectra, for example. Because the log of zero is infinite, there is a problem in representing very low energy parts of the spectrum. The log function therefore needs a lower bound both to limit the numerical range and to prevent excessive sensitivity to the low-energy, noise-dominated parts of the spectrum.

Before computing short-term power spectra, the waveform is usually processed by a simple *pre-emphasis* filter giving a 6 dB/octave increase in gain over most of its range to make the average speech spectrum roughly flat.

2.4.1 Linear Prediction Analysis

In linear prediction (LP) analysis, the vocal tract transfer function is modeled by an all-pole filter with transfer function

$$H(z) = \frac{1}{\sum_{i=0}^{p} a_i z^{-i}} \tag{2.1}$$

where p is the number of poles and $a_0 = 1$. The filter coefficients $\{a_i\}$ are chosen to minimize the mean square filter prediction error summed over the analysis window. The autocorrelation method can perform this optimization as follows.

Given a window of speech samples $\{s_n, n = 1, N\}$, the first $p + 1$ terms of the autocorrelation sequence are calculated from

$$r_i = \sum_{j=1}^{N-i} s_j s_{j+i} \tag{2.2}$$

where $i = 0,1,2 \ldots p$. The filter coefficients are then computed recursively using a set of auxiliary coefficients $\{k_i\}$, and the prediction error E which is initially equal to r_0. Let $\{k_j^{(i-1)}\}$ and $\{a_j^{(i-1)}\}$ be the reflection and filter coefficients for a filter of order $i - 1$, then a filter of order i can be calculated in three steps. Firstly, a new set of reflection coefficients are calculated.

$$k_j^{(i)} = k_j^{(i-1)} \tag{2.3}$$

for $j = 1, i - 1$ and

$$k_i^{(i)} = \left\{ r_i + \sum_{j=1}^{i-1} a_j^{(i-1)} r_{i-j} \right\} / E^{(i-1)} \tag{2.4}$$

Secondly, the prediction energy is updated.

$$E^{(i)} = (1 - k_i^{(i)} k_i^{(i)}) E^{(i-1)} \tag{2.5}$$

Finally, new filter coefficients are computed

$$a_j^{(i)} = a_j^{(i-1)} - k_i^{(i)} a_{i-j}^{(i-1)} \tag{2.6}$$

for $j = 1, i - 1$ and

$$a_i^{(i)} = -k_i^{(i)} \tag{2.7}$$

This process is repeated from $i = 1$ through to the required filter order $i = p$.

For speech recognition purpose we can choose LP coefficients $\{a_i\}$ or LP reflection coefficients $\{k_i\}$ by going through the above transformation. The required filter order must be set for use in speech recognition, which is the speech parameter vector size.

2.4.2 Filterbank Analysis

The human ear resolves frequencies non-linearly across the audio spectrum and empirical evidence suggests that designing a front-end to operate in a similar non-linear manner improves recognition performance. A popular alternative to linear prediction based analysis is therefore filterbank analysis since this provides a much more straightforward route to obtaining the desired non-linear frequency resolution.

A Fourier transform based filterbank is designed to give approximately equal resolution on a mel-scale. Fig. 2.7 illustrates the general form of this filterbank. As can be seen, the filters used are triangular and they are equally spaced along the mel-scale which is defined by

$$\text{Mel}\ (f) = 2595 \log_{10}(1 + \frac{f}{700}) \qquad \text{For f > 1000} \qquad (2.8)$$

To implement this filterbank, the window of speech data is transformed using a Fourier transform and the magnitude is taken. The magnitude coefficients are then *binned* by correlating them with each triangular filter. Here binning means that each log DFT magnitude coefficient is multiplied by the corresponding filter gain and the results accumulated. Thus, each bin holds a weighted sum representing the spectral magnitude in that filterbank channel.

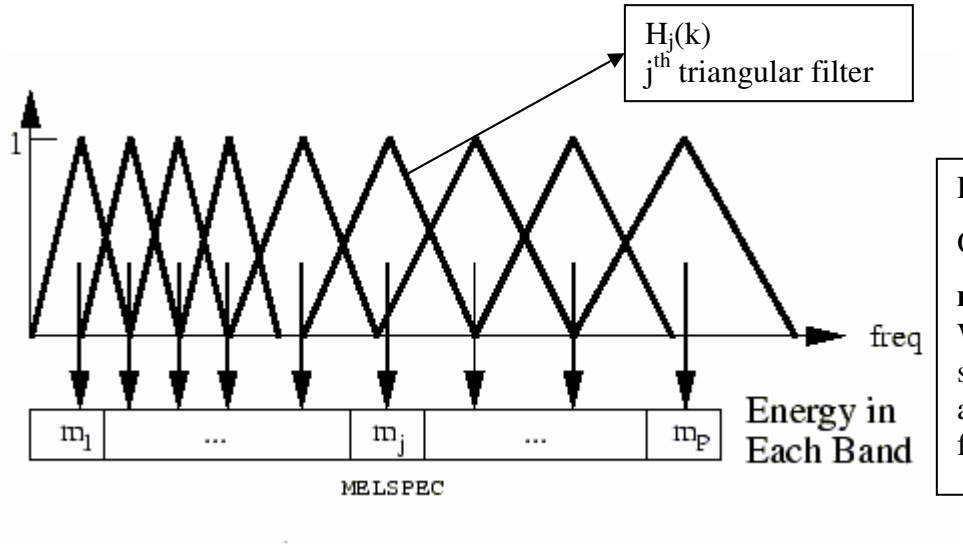


Figure 2.7 Mel-Scale Filter Bank

Output of the j^{th} Filter bank, m_j

$\mathbf{m_j} = \sum \log|S(k)| * Hi(k)$

Where |S(k)| is the DFT of the frame of speech

and $H_i(k)$ is weighting coefficients of the i^{th} filter

Normally the triangular filters are spread over the whole frequency range from zero up to the Nyquist frequency. However, band-limiting is often useful to reject unwanted frequencies or avoid allocating filters to frequency regions in which there is no useful signal energy. For filterbank analysis only, lower and upper frequency cut-offs can be set to avoid allocating filters to frequency regions. For example,

Lower frequency = 300 Hz

Higher frequency = 3400 Hz

might be used for processing telephone speech. When low and high pass cut-offs are set in this way, the specified number of filterbank channels are distributed equally on the Mel-scale across the resulting pass-band such that the lower cut-off of the first filter is at 300 and the upper cut-off of the last filter is at 3400.

2.4.3 Cepstral Features

Most often, however, cepstral parameters are required and these are found successful in speech recognition systems.

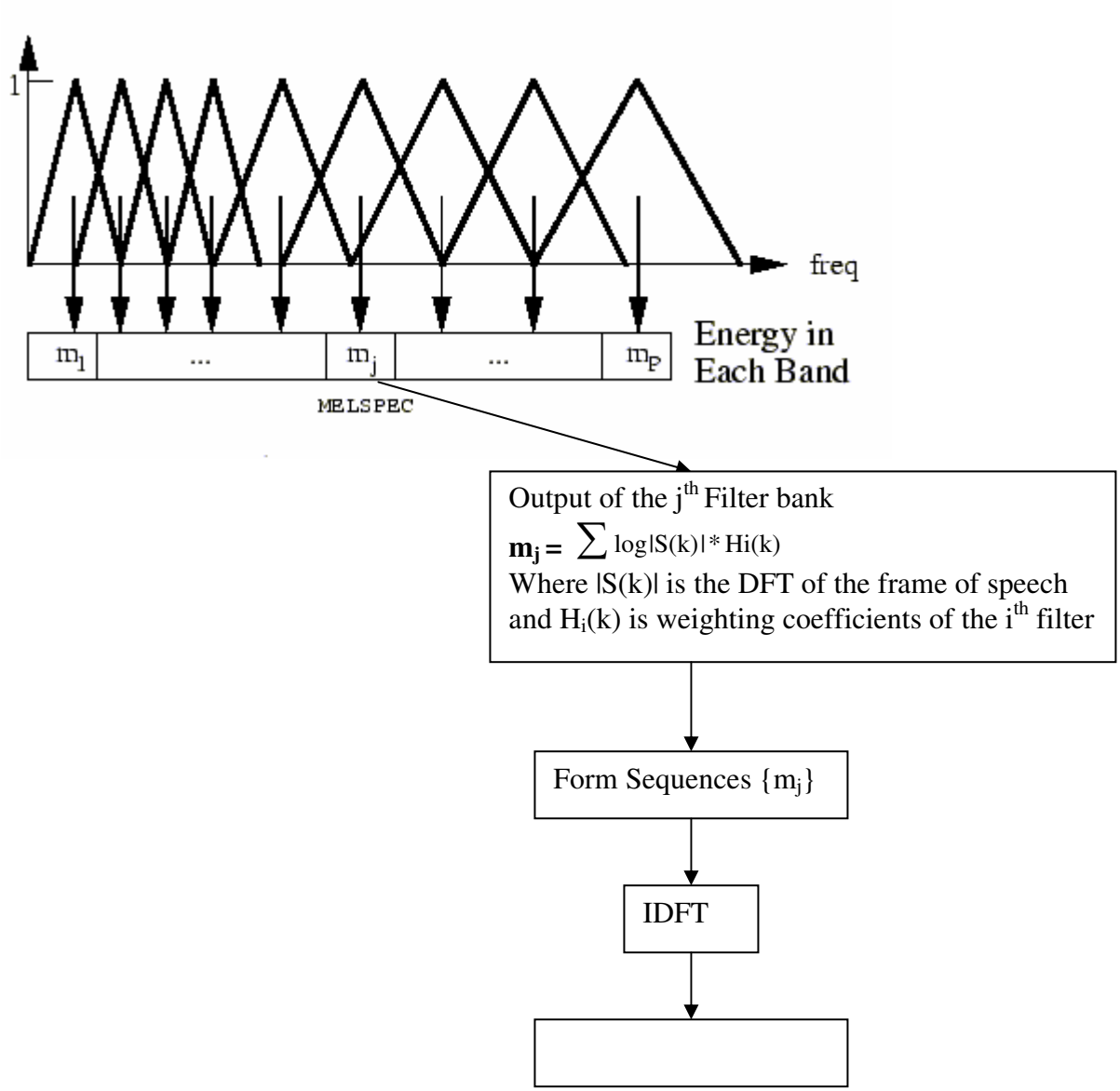


Figure 2.8 MFCC coefficients.

MFCC standing for Mel-Frequency Cepstral Coefficients (MFCCs) are the most widely used parameters. These are calculated from the log filterbank amplitudes $\{m_j\}$ using the Discrete Cosine Transform

$$c_i = \sqrt{\frac{2}{N}} \sum_{j=1}^{N} m_j \cos\left(\frac{\pi i}{N}(j - 0.5)\right) \tag{2.9}$$

where N is the number of filterbank channels. The required number of cepstral coefficients is set by the maximum value of the iteration made which is mostly 13.

Chapter Three

Acoustic (speech) Modeling

3.1 Signal Model

Real-world processes generally produce observable outputs, which can be characterized as signals. The signals can be discrete in nature (e.g., characters from a finite alphabet, quantized vectors from a codebook, etc.), or continuous in nature (e.g., speech samples, temperature measurements, etc.). The signal source can be stationary (i.e., its statistical properties do not vary with time), or nonstationary (i.e., the signal properties vary over time). The signals can be pure (i.e., coming strictly from a single source), or can be corrupted by other signal sources (e.g., noise) or by transmission distortions, reverberation, etc.

A problem of fundamental interest is characterizing such real-world signals in terms of signal models. There are several reasons why one is interested in applying signal models. First of all, a signal model can provide the basis for a theoretical description of a signal processing system which can be used to process the signal so as to provide a desired output. For example if we are interested in enhancing a speech signal corrupted by noise and transmission distortion, we can use the signal model to design a system, which will optimally remove the noise and undo the transmission distortion. A second reason why signal models are important is that they are potentially capable of letting us learn a great deal about the signal source (i.e., the real-world process that produced the signal) without having to have the source available. This property is essentially important when the cost of getting signals from the actual source is high.

In this case, with a good signal model, we can simulate the source and learn as much as possible via simulations. Finally, the most important reason why signal models are important is that they often work extremely well in practice, and enable us to realize important practical systems-e.g. prediction systems, recognition systems, identification systems, etc., in a very efficient manner.

There are several possible choices for what type of signal model is used for characterizing the properties of a given signal. Broadly, one can classify the types of signal models into the class of deterministic models, and the class of statistical models. Deterministic models generally exploit some known specific properties of the signal, e.g., that the signal is a sine wave, or a sum of exponentials, etc. In these cases, specification of the signal model is generally straightforward; all that is required is to determine (estimate) values of the parameters of the signal model (e.g., amplitude, frequency,

phase of a sine wave, amplitudes and rates of exponentials, etc.). The second broad class of signal models is the set of statistical models in which one tries to characterize only the statistical properties of the signal. Examples of such statistical models include Gaussian processes, Poisson processes, Markov processes, and hidden Markov processes, among others. The underlying assumption of the statistical model is that the signal can be well characterized as a parametric random process, and that the parameters of the stochastic process can be determined (estimated) in a precise, well-defined manner.

For the applications of interest, namely speech processing, the stochastic signal models have had good success. In this thesis we will concern ourselves strictly with one type of stochastic signal model, namely the hidden Markov model (HMMs). [12] These models are referred to as Markov sources or probabilistic functions of Markov chains in the communications literature. We will first review the theory of Markov chains and then extend the ideas to the class of hidden Markov models using several simple examples. We will then focus our attention on the three fundamental problems for HMM design, namely: the evaluation of the probability (or likelihood) of a sequence of observations given a specific HMM; the determination of a best sequence of model states; and the adjustment of model parameters so as to best account for the observed signal. We will show that once these three fundamental problems are solved, we can apply HMMs to speech recognition problems.

Neither the theory of hidden Markov models nor its applications to speech recognition is new. The basic theory was published in a series of classic papers by Baum and his colleagues in the late 1960s and early 1970s and was implemented for speech processing applications by Baker and by Jelinek and his colleagues at IBM in the 1970s. However, widespread understanding and application of the theory of HMMs to speech processing has occurred only within the past several years. [12]

This chapter is intended to provide an overview of the basic theory of HMMs (as originated by Baum and his colleagues), provide practical details on methods of implementation of the theory, and describe a couple of selected applications of the theory to distinct problems in speech recognition. The chapter presents hidden markov models as a statistical modeling tool for speech recognition problem.

3.2 Discrete Markov Processes

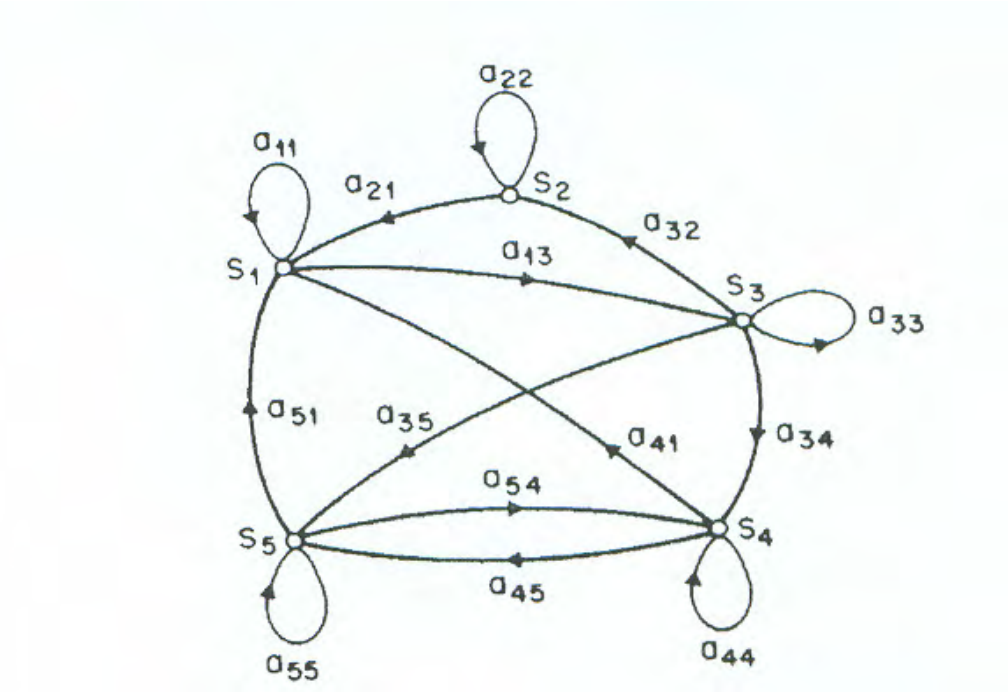


Fig 3.1: A Markov chain with 5 states (labeled S_1, to S_5) with selected state transitions.

Consider a system which may be described at any time as being in one of a set of N distinct states S_1, S_2 …….... S_N, as illustrated in Fig 3.1 (where N = 5 for simplicity). At regularly spaced discrete times, the system undergoes a change of state (possibly back to the same state) according to a set of probabilities associated with the state. We denote the time instants associated with state changes as *t* = 1, 2. . . and we denote the actual state at time *t* as q_t. A full probabilistic description of the above system would, in general, require specification of the current state (at time *t),* as well as all the predecessor states. For the special case of a discrete first order, Markov chain, this probabilistic description is truncated to just the current and predecessor state, i.e.

$$P(q_t = S_j \mid q_{t-1} = S_i , q_{t-2} = S_k , \ldots)$$
$$= P\{ q_t = S_j \mid q_{t-1} = S_i) \qquad (3.1)$$

Furthermore we only consider those processes in which the right-hand side of (1) is independent of time there by leading to the set of state transition probabilities

$$a_{ij} = P\{ q_t = S_j \mid q_{t-1} = S_i) \qquad (3.2)$$

with the state transition coefficients having the properties

$$a_{ij} > 0 \tag{3.3a}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{3.3b}$$

since they obey standard stochastic constraints.

The above stochastic process could be called an observable Markov model since the output of the process is the set of states at each instant of time, where each state corresponds to a physical (observable) event. To set ideas, consider a simple 3-state Markov model of the weather. We assume that once a day (e.g., at noon), the weather is observed as being one of the following:

State 1: rain or (snow)

State 2: cloudy

State 3: sunny.

We postulate that a single one of the three states above specifies the weather on day t, and that the matrix *A* of state transition probabilities is

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

Given that the weather on day 1 ($t = 1$) is sunny (state 3), we can ask the question: What is the probability (according to the model) that the weather for the next 7 days will be "sun-sun-rain-rain-sun-cloudy-sun . . ."? Stated more formally, we define the observation sequence O as

$O = \{S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$

corresponding to time $t = 1,2, \ldots ,8$, and we wish to determine the probability of observation sequence O, given the model. This probability can be expressed (and evaluated) as

$$\begin{aligned} P(O \mid Model) &= p(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3 \mid \text{Model}) \\ &= p(S_3) \cdot p(S_3|S_3) \cdot p(S_3|S_3) \cdot p(S_1|S_3) \cdot p(S_1|S_1) \cdot p(S_3|S_1) \cdot p(S_2|S_3) \cdot p(S_3|S_2) \\ &= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \end{aligned}$$

$$= 1.(0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2) = 1.536 \times 10^{-4}$$

Where we use the notation

$$\pi_j = P\{ q_1 = S_j \} \qquad 1 \le j \le N \qquad (3.4)$$

3.3 Extension to Hidden Markov Models

So far we have considered Markov models in which each state corresponded to an observable (physical) event. This model is too restrictive to be applicable to many problems of interest. In this section we extend the concept of Markov models to include the case where the observation is a probabilistic function of the state-i.e. the resulting model (which is called a hidden Markov model) is a doubly embedded stochastic process with an underlying stochastic process that is *not* observable (it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observations. To fix ideas, consider the following model of some simple coin tossing experiments.

3.3.1 Coin Toss Models: Assume the following scenario. You are in a room with a barrier (e.g., a curtain) through which you cannot see what is happening. On the other side of the barrier is another person who is performing a coin (or multiple coins) tossing experiment. The other person will not tell you anything about what he is doing exactly; he will only tell you the result of each coin flip. Thus a sequence of *hidden* coin tossing experiments is performed, with the observation sequence consisting of a series of heads and tails; e.g., a typical observation sequence would be

$$O = O_1 O_2 O_3 ..... O_T$$
$$= HHTTTHTTH ... H$$

where *H* stands for heads and T stands for tails.

Given the above scenario, the problem of interest is how do we build an HMM to explain (model) the observed sequence of heads and tails. The first problem one faces is deciding what the states in the model correspond to, and then deciding how many states should be in the model. One possible choice would be to assume that only a single biased coin was being tossed. In this case we could model the situation with a 2-state model where each state corresponds to a side of the coin (i.e., heads or tails). This model is depicted in Fig. 3.2(a). In this case the Markov model is observable, and the only issue for complete specification of the model would be to decide on the best value for the bias (i.e., the probability of, say; heads). Interestingly, an equivalent HMM to that of Fig. 3.2(a) would be

a degenerate 1-state model, where the state corresponds to the single biased coin, and the unknown parameter is the bias of the coin.

A second form of HMM for explaining the observed sequence of coin toss outcome is given in Fig. 3.2(b). In this case there are 2 states in the model and each state corresponds to a different, biased, coin being tossed. Each state is characterized by a probability distribution of heads and tails, and transitions between states are characterized by a state transition matrix. The physical mechanism which accounts for how state transitions are selected could itself be a set of independent coin tosses, or some other probabilistic event.

A third form of HMM for explaining the observed sequence of coin toss outcomes is given in Fig. 3.2(c). This model corresponds to using 3 biased coins, and choosing from among the three, based on some probabilistic event.

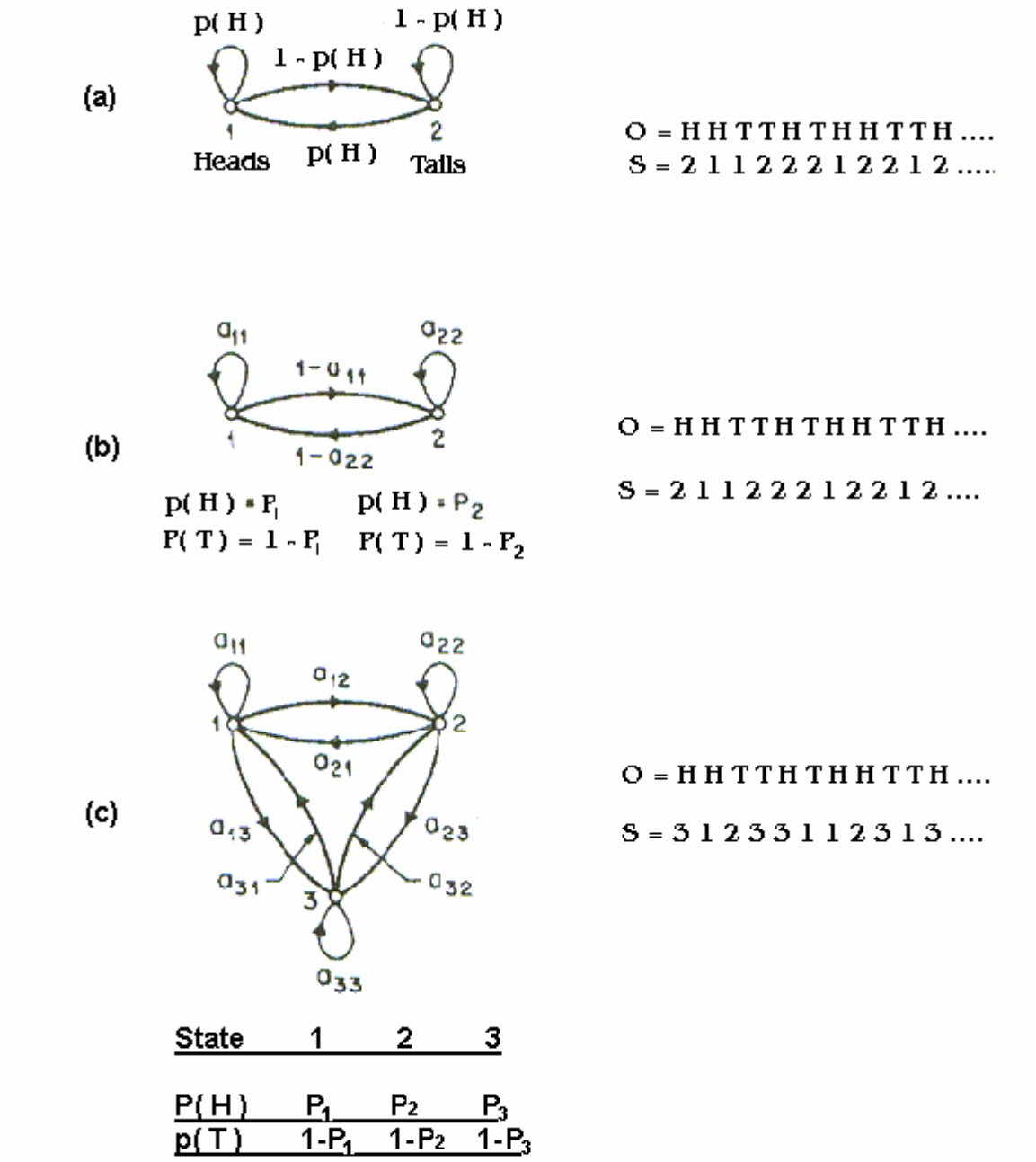


Fig 3.2: Three possible Markov models, which can account for the result of Hidden coin tossing experiments (a) a coin model, (b) 2-coin model (a) 3-coin model

Given the choice among the three models shown in Fig. 3.2 for explaining the observed sequence of heads and tails, a natural question would be which model best matches the actual observations. It should be clear that the simple 1-coin model of Fig. 3.2(a) has only 1 unknown parameter; the 2-coin

model of Fig. 3.2(b) has 4 unknown parameters; and the 3-coin model of Fig. 3.2(c) has 9 unknown parameters. Thus, with the greater degrees of freedom, the larger HMMs would seem to inherently be more capable of modeling a series of coin tossing experiments than would equivalently smaller models. Although this is theoretically true, we will see later in this chapter that practical considerations impose some strong limitations on the size of the models that we can consider. Furthermore, It might just be the case that only a single coin is being tossed. Then using the 3-coin model of Fig. 3.2(c) would be inappropriate, since the actual physical event would not correspond to the model being used.

3.3.2 The Urn and Ball Model: To extend the ideas of the HMM to a somewhat more complicated situation, consider the urn and ball system of Fig. 3.3. We assume that there are N (large) glass urns in a room. Within each urn there are a large number of colored balls. We assume there are M distinct colors of the balls. The physical process for obtaining observations is as follows. A genie is in the room, and according to some random process, he (or she) chooses an initial urn.

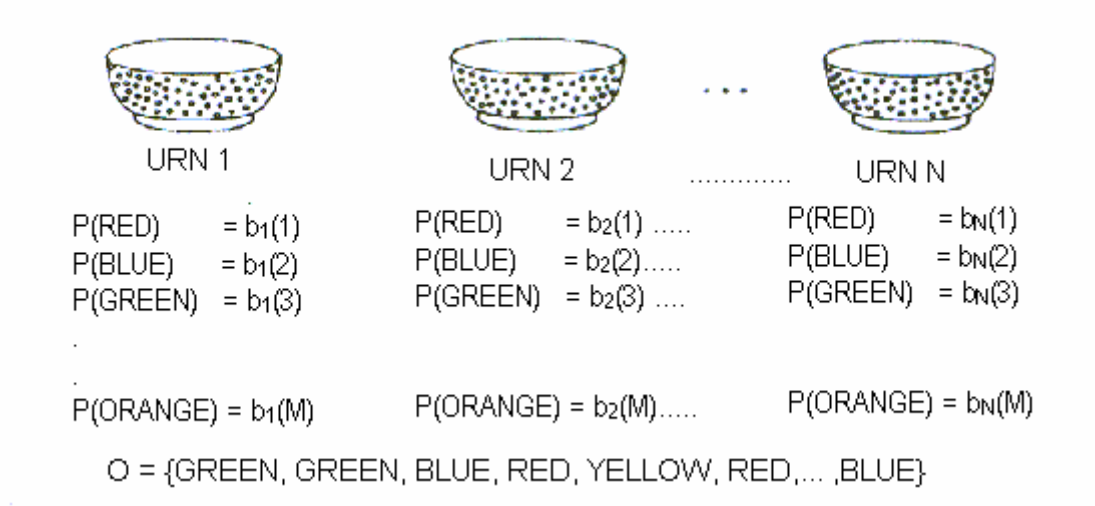


Figure 3.3 An N state urn and ball model which illustrate the general case of discrete symbol HMM.

From this urn, a ball is chosen at random, and its color is recorded as the observation. The ball is then replaced in the urn from which it was selected. A new urn is then selected according to the random selection process associated with the current urn, and the ball selection process is repeated. This entire process generates a finite observation sequence of colors, which we would like to model as the observable output of an HMM.

It should be obvious that the simplest HMM that corresponds to the urn and ball process is one in which each state corresponds to a specific urn, and for which a (ball) color probability is defined for each state. The choice of urns is dictated by the state transition matrix of the HMM.

3.4 Elements of HMM

The above examples give us a pretty good idea of what an HMM is and how it can be applied to some simple scenarios. We now formally define the elements of an HMM, and explain how the model generates observation sequences.

An HMM is characterized by the following:

1) **N, the number of states in the model**. Although the states are hidden, for many practical applications there is often some physical significance attached to the states or to sets of states of the model. Hence, in the coin tossing experiments, each state corresponded to a distinct biased coin. In the urn and ball model, the states corresponded to the urns. Generally the states are interconnected in such a way that any state can be reached from any other state (e.g., an ergodic model); however, we will see later in this chapter that other possible interconnections of states are often of interest. We denote the individual states as

$$S = \{ S_1, S_2 \ldots S_N \}.$$

2) **M, the number of distinct observation symbols per state** i.e. ., the discrete alphabet size. Observation symbols correspond to the physical output of the system being modeled. For the coin toss experiments the observation symbols were simply heads or tails; for the ball and urn model they were the colors of the balls selected from the urns. We denote the individual symbols as

$$V = \{ V_1, V_2 \ldots V_M \}$$

3) **A, The state transition probability distribution**

$$A = \{ a_{ij} \} \text{ where}$$

$$a_{ij} = P\{ q_t = S_j \mid q_{t-1} = S_i \} \qquad (3.7)$$

For the special case where any state can reach any other state in a single step, we have

$$a_{ij} > 0 \text{ for all i, j.}$$

For other types of HMMs, we would have $a_{ij} = 0$ for one or more (i, j) pairs.

4*)* **B, The observation symbol probability distribution** in a given state

$B = \{b_j(k)\}$, where $b_j(k)$ is observation symbol probability distribution in state j and V_k is the K^{th} observation symbol

$$b_j(k) = P(V_k \text{ at } t \mid q_t = S_j) \qquad 1 \leq j \leq N \qquad 1 \leq k \leq M. \tag{3.8}$$

5) π, **The initial state distribution**

$$\pi = \{ \pi_j \}$$

$$\pi_j = P\{ q_1 = S_i \} \qquad 1 \leq j \leq N \tag{3.9}$$

Given appropriate values of N, M, A, B, and π, the HMM can be used as a generator to give an observation sequence

$O = O_1, O_2, \ldots O_T$

(where each observation O, is one of the symbols from V, and *T* is the number of observations in the sequence) as follows:

1) Choose an initial state $q_1 = S_j$ according to the initial state distribution π.

2) Set t = 1.

3) Choose $O_t = V_k$ according to the symbol probability distribution in state S_j i.e., $b_j(k)$.

4) Transit to a new state; according to the state transition probability distribution

5) Set t = t + 1; return to step (3) if t < T; otherwise terminate the procedure.

The above procedure can be used as both a generator of observations, and as a model for how a given observation sequence was generated by an appropriate HMM.

It can be seen from the above discussion that a complete specification of an HMM requires specification of two model parameters *(N* and *M),* specification of observation symbols, and the specification of the three probability measures *A, B,* and π. For convenience, we use the compact notation

$$\lambda = (A, B, \pi) \tag{3.11}$$

to indicate the complete parameter set of the model.

3.5 The three basic problems of HMMs

Given the form of HMM of the previous section, there are three basic problems of interest that must be solved for the model to be useful in real-world applications. These problems are the following:

Problem 1:
Given the observation sequence $O = O_1, O_2, \ldots O_T$ and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O \mid \lambda)$, the probability of the observation sequence, given the model?

Problem 2:
Given the observation sequence $O = O_1, O_2, \ldots O_T$, and the model λ, how do we choose a corresponding state sequence $Q = q_1, q_2, \ldots q_T$ which is optimal in some meaningful sense (i.e., best "explains" the observations)?

Problem 3:
How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O \mid \lambda)$?

Problem 1 is the evaluation problem namely given a model and a sequence of observations, how do we compute the probability that the observed sequence was produced by the model. We can also view the problem as one of scoring how well a given model matches a given observation sequence. The latter viewpoint is extremely useful. For example, if we consider the case in which we are trying to choose among several competing models, the solution to Problem 1 allows us to choose the model which best matches the observations.

Problem 2 is the one in which we attempt to uncover the hidden part of the model. i.e. to find the "correct" state sequence. It should be clear that for all but the case of degenerate models, there is no "correct" state sequence to be found. Hence, for practical situations, we usually use an optimality criterion to solve this problem as best as possible. Unfortunately, as we will see, there are several reasonable optimality criteria that can be imposed, and hence the choice of criterion is a strong function of the intended use for the uncovered state sequence. Typical uses might be to learn about the

structure of the model, to find optimal state sequences for continuous speech recognition, or to get average statistics of individual states, etc.

Problem 3 is the one in which we attempt to optimize the model parameters so as to best describe how a given observation sequence comes about. The observation sequence used to adjust the model parameters is called a training sequence since it is used to "train" the HMM. The training problem is the crucial one for most applications of HMMs, since it allows us to optimally adapt model parameters to the observed training data- i.e., to create best models for real phenomena.

To fix ideas, consider the following simple isolated word speech recognizer. For each word of a W word vocabulary, we want to design a separate *N-state* HMM. We represent the speech signal of a given word as a time sequence of coded spectral vectors. We assume that the coding is done using a spectral codebook with M unique spectral vectors; hence each observation is the index of the spectral vector closest (in some spectral sense) to the original speech signal. Thus, for each vocabulary word, we have a training sequence consisting of a number of repetitions of sequences of codebook indices of the word (by one or more talkers). The first task is to build individual word models. The task is done by using the solution to Problem 3 to optimally estimate model parameters for each word model. To develop an understanding of the physical meanings of model states, we use the solution to problem 2 to segment each word training sequences into states, and then study the properties of the spectral vectors that lead to the observations occurring in each state. The goal here would be to make refinements on the model (e.g., more states, different codebook size, etc.) so as to improve its capability of modeling the spoken word sequences. Finally, once the set of W HMMs has been designed and optimized and thoroughly studied, recognition of an unknown word is performed using the solution to Problem 1 to score each word model based upon the given test observation sequence, and select the word whose model score is highest (i.e., the highest likelihood).

In the next section, we present formal mathematical solutions to each of the three fundamental problems for HMMs. We shall see that the three problems are linked together tightly under probabilistic framework.

3.6 Solutions to the three basic problems of HMM

3.6.1 Solution to problem 1

We wish to calculate the probability of the observation sequence $O = O_1, O_2, \ldots O_T$ given the model λ i.e. P(O | λ). The most straight forward way of doing this is through enumerating every possible state sequence of length T (the number of observations). Consider one such state sequence

$$Q = q_1 q_2 \ldots\ldots q_T \tag{3.12}$$

where q_1 is the initial state. The probability of the observation sequence O for the state sequence of (3.12) is

$$P(O \mid Q, \lambda) = \prod_{t=1}^{T} P(O_t \mid q_t, \lambda) \tag{3.13a}$$

where we have assumed statistical independence of observations. Thus, we get

$$P(O \mid Q, \lambda) = b_{q_1}(O_1).b_{q_2}(O_{21}).\ldots\ldots b_{q_T}(O_T) \tag{3.13b}$$

The probability of such a state sequence Q can be written as

$$P(Q \mid \lambda) = \pi_{q_1}.a_{q_1 q_2}.a_{q_2 q_3} \ldots\ldots\ldots\ldots\ldots\ldots\ldots a_{q_{T-1} q_T} \tag{3.14}$$

The joint probability of O and Q, i.e., the probability that O and Q occur simultaneously, is simply the product of the above two terms, i.e.,

$$P(O, Q \mid \lambda) = P(O \mid Q, \lambda) P(Q, \lambda) \tag{3.15}$$

The probability of observation O (given the model) is obtained by summing this joint probability over all possible state sequences q giving

$$P(O \mid \lambda) = \sum_{all\ Q} P(O \mid Q, \lambda) P(Q, \lambda) \tag{3.16}$$

$$= \sum_{q_1, q_1, \ldots\ldots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \ldots\ldots a_{q_{T-1} q_T} b_{q_T}(O_T) \tag{3.17}$$

The interpretation of the computation in the above equation is the following. Initially (at time t = 1) we are in state q_1, with probability π_{q1} and generate the symbol O_1 (in this state) with probability $bq_1(O_1)$. The clock changes from time t to t + 1 (t = 2) and we make a transition to state q_2 from state q_1 with probability a_{q1q2} and generate symbol O_2 with probability $b_{q2}(O_2)$. This process continues in

this manner until we make the last transition (at time T) from state q_{T-1} to state q_T with probability $a_{qT\text{-}1qT}$ and generate symbol O_T with probability $b_{qT}(O_T)$.

A little thought should convince us that the calculation of $P(O \mid \lambda)$, according to its direct definition (3.17) involves on the order of (2 x T x N^T) calculations, since at every t = 1,2, . . . , T, there are N possible states which can be reached (i.e., there are N^T possible state sequences), and for each such state sequence about (2 x T) calculations are required for each term in the sum of (17). (To be precise, we need (2T - 1) x N^T multiplications, and N^T - 1 additions.) This calculation is computationally unfeasible, even for small values of N and T; e.g., for N = 5 (states), T = 100 (observations), there are on the order of 2 x 100 x $5^{100} \approx 10^{72}$ computations! Clearly a more efficient procedure is required to solve Problem 1. Fortunately, such a procedure exists and is called the forward-backward procedure.

3.6.1.1 The Forward-Backward Procedure: Consider the forward variable $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(O_1, O_2 \ldots\ldots\ldots O_T, q_t = s_i \mid \lambda) \tag{3.18}$$

i.e., the probability of the partial observation sequence $O_1\ O_2\ \ldots\ldots\ O_t$, (until time t) and state S_i at time t, given the model λ.

We can solve for $\alpha_t(i)$ inductively, as follows:

1) Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \le i \le N \tag{3.19}$$

2) Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1}) \qquad \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix} \tag{3.20}$$

3) Termination:

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.21}$$

Step 1) initializes the forward probabilities as the joint probability of state S_i and initial observation O. The induction step, which is the heart of the forward calculation, is illustrated in Fig. 3.4(a). This figure shows how state S_j can be reached at time $t + 1$ from the N possible states, S_i, $1 \le i \le N$, at time

t. Since $\alpha_t(i)$ is the probability of the joint event that $O_1\ O_2$. ……. . O_t, are observed, and the state at time t is S_i. The product $\alpha_t(i) \bullet a_{ij}$ is then the probability of the joint event that $O_1\ O_2$. ……. . O_t are observed, and state S_j, is reached at time $t + 1$ via state S_i at time t. Summing this product over all the 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 accompanying previous partial observations.

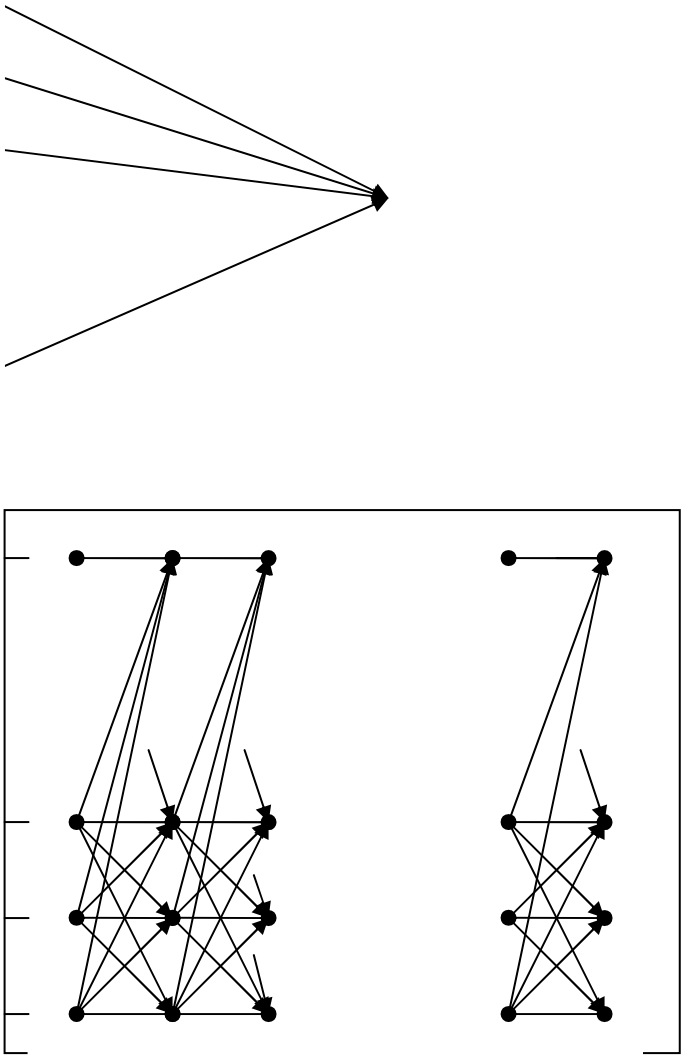


Figure 3.4 (a) Illustration of the sequence of operations required for the computation of the forward variable $\alpha_{t-1}(j)$
(b) Implementation of the $\alpha_t(j)$ in terms of a lattice of observations t, and states i.

Once this is done and S_j, is known, it is easy to see that $\alpha_{t+1}(j)$ is obtained by accounting for observation O_{t+1} in state j, i.e., by multiplying the summed quantity by the probability $b_j(O_{t+1})$. The computation of (3.20) is performed for all states *j,* $1 \le j \le N$, for a given t; the computation is then repeated for t = 1, 2, . . . , T - 1.

Finally, step 3) gives the desired calculation of $P(O \mid \lambda)$ as the sum of the terminal forward variables $\alpha_T(i)$. This is the case since, by definition,

$$\alpha_T(i) = P(O_1 O_2 \ldots\ldots\ldots \ldots O_T, q_T = S_i \mid \lambda) \qquad (3.22)$$

and hence $P(O \mid \lambda)$ is just the sum of the $\alpha_T(i)$'s.

If we examine the computation involved in the calculation of $\alpha_t(j)$, where $1 \le t \le T$, $1 \le j \le N$, we see that it requires on the order of $N^2 \times T$ calculations, rather than $2 \times T \times N^T$ as required by the direct calculation. (Again, to be precise, we need N(N + 1)(T - 1) + N multiplications and N(N - 1)(T - 1) additions.). For $N = 5$, T = 100, we need about 3000 computations for the forward method, versus 10^{72} computations for the direct calculation, a savings of about 10^{69} computations.

The forward probability calculation is, in effect, based upon the lattice (or trellis) structure shown in Fig. 3.4(b). The key is that since there are only N states (nodes at each time slot in the lattice), all the possible state sequences will remerge into these N nodes, no matter how long the observation sequence.

At time t = 1 (the first time slot in the lattice), we need to calculate values of $\alpha_t(j)$, $1 \le j \le N$. At times t = 2, 3, . . . , T, we only need to calculate values of $\alpha_t(j)$, $1 \le j \le N$, where each calculation involves only *N* previous values of $\alpha_{t-1}(j)$ because each of the N grid points is reached from the same N grid points at the previous time slot.

In a similar manner, we can consider a backward variable $\beta_t(i)$ defined as

$$\beta_t(i) = P(O_{t+1} O_{t+2} \ldots\ldots\ldots \ldots O_T \mid q_t = S_i, \lambda) \qquad (3.23)$$

i.e., the probability of the partial observation sequence $O_{t+1}\ O_{t+2}\ O_{t+3}\ldots\ldots\ldots\ O_T$ (from t + 1 to the end), given state S_i at time t and the model λ. Again we can solve for $\beta_t(i)$ inductively, as follows:

1) Initialization:

$$\beta_t(i) = 1 \qquad 1 \le i \le N \qquad (3.24)$$

2) Induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \qquad t = T-1, T-2 \ldots\ldots, 1 \qquad 1 \le i \le N \quad (3.25)$$

The initialization step 1, arbitrarily defines $\beta_t(i)$ to be 1 for all i. Step 2), which is illustrated in Fig. 3.5**,** shows that in order to have been in state S_i at time *t,* and to account for the observation sequence from time t + 1 on,

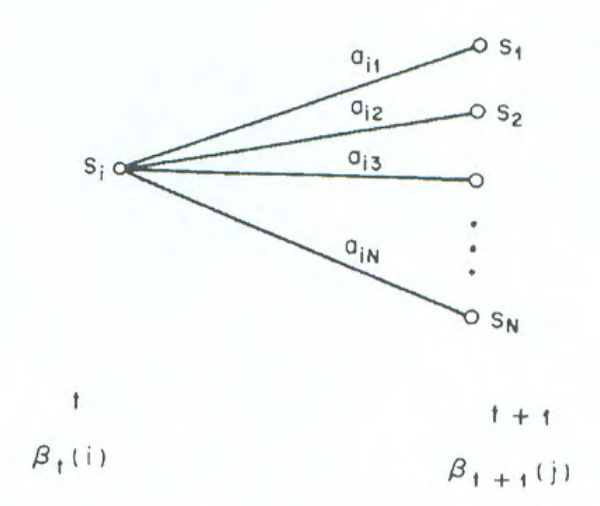


Figure 3.5. Illustration of the sequence of operations required for the computation of the backward variable $\beta_t(i)$.

you have to consider all possible states S_j at time t + 1, accounting for the transition from S_i to S_j, (the a_{ij} term), as well as the observation O_{t+1} in state j (the $b_j(O_{t+1})$ term), and then account for the remaining partial observation sequence from state j (the $\beta_{t+1}(j)$ term). We will see later how the backward, as well as the forward calculations are used extensively to help solve fundamental Problems 2 and 3 of HMMs.

3.6.2 Solution to problem 2

Unlike Problem 1 for which an exact solution can be given, there are several possible ways of solving Problem 2, namely finding the "optimal" state sequence associated with the given observation sequence. The difficulty lies with the definition of the optimal state sequence; i.e., there are several

possible optimality criteria. For example, one possible optimality criterion is to choose the states q_t which is individually most likely. This optimality criterion maximizes the expected number of correct individual states. To implement this solution to Problem 2, we define the variable

$$\gamma_t(i) = P(q_t = S_i \mid O, \lambda) \tag{3.26}$$

i.e., the probability of being in state S_i at time t, given the observation sequence O, and the model λ. Equation (3.26) can be expressed simply in terms of the forward-backward variables, i.e.,

$$\gamma_t(i) = \frac{\alpha_t(i).\beta_t(i)}{P(O \mid \lambda)} = \frac{\alpha_t(i).\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i).\beta_t(i)} \tag{3.27}$$

since $\alpha_t(i)$, accounts for the partial observation sequence $O_1\ O_2\ O_3$ ..... O_t, and state S_i at t, while $\beta_t(i)$ accounts for the remainder of the observation sequence $O_{t+1}\ O_{t+2}\ O_{t+3}$ ..... O_T given state S_i at t. The normalization factor $P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_t(i).\beta_t(i)$ makes $\gamma_t(i)$ a probability measure so that

$$\sum_{i=1}^{N} \gamma_t(i) = 1 \tag{3.28}$$

Using $\gamma_t(i)$ we can solve for the individually most likely state q_t at time t, as

$$q_t = \arg \max_{1 \le i \le N} [\gamma_t(i)] \qquad 1 \le t \le T \tag{3.29}$$

Although (3.29) maximizes the expected number of correct states (by choosing the most likely state for each t). There could be some problems with the resulting state sequence. For example, when the HMM has state transitions which have zero probability ($a_{ij} = 0$ for some i and j), the "optimal" state sequence may, in fact, not even be a valid state sequence. This is due to the fact that the solution of (3.29) simply determines the most likely state at every instant, without regard to the probability of occurrence of sequences of states.

One possible solution to the above problem is to modify the optimality criterion. For example, one could solve for the state sequence that maximizes the expected number of correct pairs of states (q_t, q_{t+1}), or triples of states (q_t, q_{t+1}, q_{t+2}) etc. Although these criteria might be reasonable for some

applications, the most widely used criterion is to find the *single* best state sequence (path), i.e., to maximize P($Q|O$, λ) which is equivalent to maximizing P(Q, O | λ). A formal technique for finding this single best state sequence exists, based on dynamic programming methods, and is called the Viterbi algorithm.

3.6.2.1 Viterbi Algorithm: To find the single best state sequence, $Q = \{q_1\ q_2\ q_3\ \ldots\ldots\ q_T\}$, for the given observation sequence $O = \{O_1\ O_2\ O_3\ \ldots\ldots\ O_T\}$, we need to define the quantity

$$\delta_t(i) = \max_{q_1, q_2 \ldots\ldots, q_{t-1}} P(q_1, q_2, \ldots\ldots, q_t = i, O_1 O_2 \ldots O_t \mid \lambda) \qquad (3.30)$$

i.e., $\boldsymbol{\delta_t(i)}$ is the best score (highest 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

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i).\, a_{ij}\right] b_j(O_{t+1}) \qquad (3.31)$$

To actually retrieve the state sequence, we need to keep track of the argument which maximized (3.31), for each t and j. We do this via the array $\boldsymbol{\psi_t(j)}$. The complete procedure for finding the best state sequence can now be stated as follows:

1) Initialization:

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

$$\psi_1(i) = 0.$$

2) Recursion:

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

$$\psi_t(j) = \operatorname*{argmax}_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}], \qquad 2 \le t \le T, \quad 1 \le j \le N.$$

3) Termination:

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

$$q_T^* = \operatorname*{argmax}_{1 \le i \le N} [\delta_T(i)].$$

4) Path (state sequence) backtracking:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \cdots, 1.$$

It should be noted that the Viterbi algorithm is similar (except for the backtracking step) in implementation to the forward calculation of (3.19)-(3.21). The major difference is the maximization in (33a) over previous states, which are used in place of the summing procedure in (20). It also should be clear that a lattice (or trellis) structure efficiently implements the computation of the Viterbi procedure.

3.6.3 Solution to Problem 3

The third, and by far the most difficult, problem of HMMs is to determine a method to adjust the model parameters (A, B, and π) to maximize the probability of the observation sequence given the model. There is no known way to analytically solve for the model which maximizes the probability of the observation sequence. In fact, given any finite observation sequence as training data, there is no optimal way of estimating the model parameters. We can, however, choose λ = (A, B, π) such that

P($O \mid \lambda$) is locally maximized using an iterative procedure such as the Baum-Welch method (or equivalently the EM (expectation-modification) method ,or using gradient techniques**.** In this section we discuss one iterative procedure, based primarily on the classic work of Baum and his colleagues, for choosing model parameters.

In order to describe the procedure for reestimation (iterative update and improvement) of HMM parameters, we first define $\xi_t(i, j)$, the probability of being in state S_i at time t, and state S_j at time t + 1, given the model and the observation sequence O, i.e.

$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda) \qquad (3.36)$$

The sequence of events leading to the conditions required by (3.36) is illustrated in Fig. 3.6. It should be clear, from the

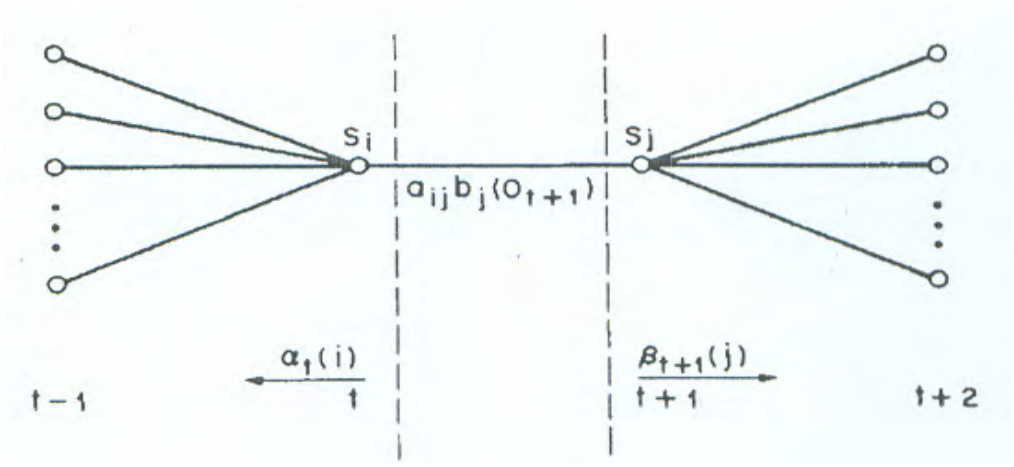


Figure 3.6. Illustration of the sequence of operations required for the computation of the joint event that the system is in state S_i at time t and state S_j at time t + 1.

definitions of the forward and backward variables, that we can write $\xi_t(i, j)$**,** in the form

$$\xi_t(i, j) = \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1}).\beta_{t+1}(j)}{P(O \mid \lambda)}$$

$$\xi_t(i, j) = \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1}).\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)\, a_{ij}\, b_j(O_{t+1}).\beta_{t+1}(j)} \qquad (3.37)$$

where the numerator term is just P(q_t = S_i, q_{t+1} = S_j , $O \mid \lambda$) and the division by P($O \mid \lambda$) gives the desired probability measure.

We have previously defined $\gamma_t(i)$ as the probability of being in state S_i at time t, given the observation sequence and the model; hence we can relate $\gamma_t(i)$ to $\xi_t(i, j)$ by summing over j, giving

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j) \tag{3.38}$$

If we sum $\gamma_t(i)$ over the time index t, we get a quantity which can be interpreted as the expected (over time) number of times that state S_i is visited, or equivalently, the expected number of transitions made from state S_i, (if we exclude the time slot $t = T$ from the summation). Similarly, summation of ξ_t(i, j) over t (from $t = 1$ to $t = T - 1$) can be interpreted as the expected number of transitions from state S_i to state S_j. That is

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from } S_i \tag{3.39a}$$

$$\sum_{t=1}^{T-1} \xi_t(i) = \text{expected number of transitions from } S_i \text{ to } s_j \tag{3.39b}$$

Using the above formulas (and the concept of counting event occurrences) we can give a method for reestimation of the parameters of an HMM. A set of reasonable reestimation formulas for π, A. and B are

$$\bar{\pi}_i = \text{expected frequency (number of times) in state } S_i \text{ at time } (t = 1) = \gamma_i(i) \tag{3.40a}$$

$$a^*_{ij} = \frac{\text{expected number of transitions form } S_i \text{ to state Sj}}{\text{expected number of transitions form } S_i}$$

$$a^*_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$b^*_j(k) = \frac{\text{expected number of transitions form Sj and observing } v_k}{\text{expected number of times in state } S_j}$$

$$b^*_j(k) = \frac{\sum_{t=1}^{T} \gamma_t(j)}{\sum_{s.t\_O_t=V_k}^{T} \gamma_t(j)}$$

If we define the current model as λ = (A, B, π), and use that to compute the right-hand sides of (40a)-(40c), and we define the reestimated model as ***λ* = (A*, B*, π*),*** as determined from the left-hand sides of (40a)-(40c), then it has been proven by Baum and his colleagues , that either

1) The initial model λ defines a critical point of the likelihood function, in which case $\lambda^* = \lambda$; or

2) Model λ^* is more likely than model λ in the sense that $P(O \mid \lambda^*) > P(O \mid \lambda)$, i.e., we have found a new model from which the observation sequence is more likely to have been produced.

Based on the above procedure, if we iteratively use λ^* in place of λ and repeat the reestimation calculation, we then can improve the probability of O being observed from the model until some limiting point is reached. The final result of this reestimation procedure is called a maximum likelihood estimate of the HMM. It should be pointed out that the forward-backward algorithm leads to local maxima only, and that in most problems of interest, the optimization surface is very complex and has many local maxima.

3.6.4 Notes on the Reestimation Procedure: The reestimation formulas can readily be interpreted as an implementation of the EM algorithm of statistics in which the E (expectation) step is the calculation of the auxiliary function $Q(\lambda, \lambda^*)$ and the M (modification) step is the maximization over λ^*. Thus the Baum-Welch reestimation equations are essentially identical to the EM steps for this particular problem.

An important aspect of the reestimation procedure is that the stochastic constraints of the HMM parameters, namely

$$\sum_{i=1}^{N} \pi^* = 1 \qquad (3.41a)$$

$$\sum_{i=1}^{N} a_{ij}^{*} = 1 \tag{3.41b}$$

$$\sum_{k=1}^{M} b_{j}^{*}(k) = 1 \tag{3.41c}$$

are automatically satisfied at each iteration. By looking at the parameter estimation problem as a constrained optimization of $P(O|\lambda)$ (subject to the constraints of (3.41), the techniques of Lagrange multipliers can be used to find the values of π_i, a_{ij} and $b_j(k)$ which maximize $P(O|\lambda)$. Based on setting up standard Lagrange optimization using Lagrange multipliers, it can readily be shown that $P(O|\lambda)$ is maximized when the following conditions are met: [12]

$$\pi = \frac{\pi_i \frac{\partial P}{\partial \pi_i}}{\sum_{k=1}^{N} \pi_k \frac{\partial P}{\partial \pi_k}}$$

$$a_{ij} = \frac{a_{ij} \frac{\partial P}{\partial a_{ij}}}{\sum_{k=1}^{N} a_{ik} \frac{\partial P}{\partial a_{ik}}}$$

$$b_j(k) = \frac{b_j(k) \frac{\partial P}{\partial b_j(k)}}{\sum_{l=1}^{M} b_j(l) \frac{\partial P}{\partial b_j(l)}}$$

By appropriate manipulation of (3.42), the right-hand sides of each equation can be readily converted to be identical to the right-hand sides of each part of (3.40a)-(3.40c), thereby showing that the reestimation formulas are indeed exactly correct at critical points of $P(O|\lambda)$. In fact, the form of (3.42) is essentially that of a reestimation formula in which the left-hand side is the reestimate and the right-hand side is computed using the current values of the variables.

3.6.5 Continuous Observation Densities in HMMs

If the observation does not come from a finite set, but from a continuous space, the discrete output distribution discusses in the previous sections needs to be modified. The difference between the discrete and continuous HMM lies in a different form of output probability functions.

In choosing continuous output probability density function bi 0), the first candidate is the multivariate Gaussian mixture density functions. This is because they can approximate any continuous probability density function. With M Gaussian mixture functions, we have:

$$b_j(O) = \sum_{k=1}^{M} c_{jk} N(O, \mu_{jk}, \sum_{jk}) = \sum_{k=1}^{M} c_{jk} b_{jk}(O) \qquad (3.43)$$

where $N(O, \mu_{jk}, \sum_{jk})$ denotes a single Gaussian density function with mean vector μ_{jk} and covariance matrix $\sum_{jk}$ for state j, M denotes the number of mixture components. The larger M is the better the approximation but greater computations required. And c_{jk} is the weight of the k^{th} mixture component satisfying

$$\sum C_{jk} = 1$$

Without going to the derivations, the reestimation formulas are the following:

$$\mu^*_{jk} = \frac{\sum_{t=1}^{T} \frac{p(O, S_t = j, k_t = k \mid \lambda)}{p(O \mid \lambda)}}{\sum_{t=1}^{T} \frac{p(O, S_t = j, k_t = k \mid \lambda)}{p(O \mid \lambda)}} = \frac{\sum_{t=1}^{T} \varsigma_t(k, j) O_t}{\sum_{t=1}^{T} \varsigma_t(k, j)} \qquad (3.44)$$

$$\sum\nolimits^*_{jk} = \frac{\sum_{t=1}^{T} \frac{p(O, S_t = j, k_t = k \mid \lambda)}{p(O \mid \lambda)} (O_t - \mu^*_{jk})(O_t - \mu^*_{jk})^T}{\sum_{t=1}^{T} \frac{p(O, S_t = j, k_t = k \mid \lambda)}{p(O \mid \lambda)}}$$

$$= \frac{\sum_{t=1}^{T} \zeta_t(k, j)(O_t - \mu^*_{jk})(O_t - \mu^*_{jk})^T}{\sum_{t=1}^{T} \varsigma_t(k, j)} \qquad (3.45)$$

where

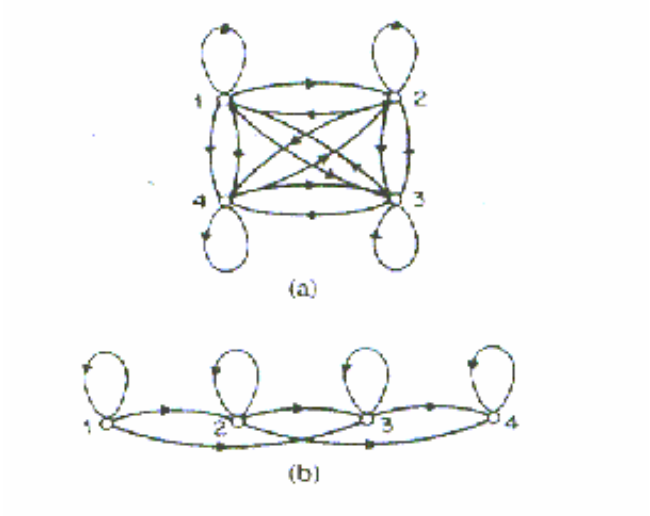
$$\sum_{t=1}^{T} \varsigma_t(k, j) = \sum_{t=1}^{T} \frac{p(O, S_t = j, k_t = k \mid \lambda)}{p(O \mid \lambda)} = \frac{\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} C_{jk}(O_t) \beta_t(j)}{\sum_{i=1}^{N} \alpha_T(i)}$$

and

$$C_{jk}^{*} = \frac{\sum_{t=1}^{T} \zeta_t(k, j)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \zeta_t(k, j)}$$

3.7 Types of HMMs

Upto now, we have only considered the special case of ergodic or fully connected HMMs, in which every state of the model could be reached (in a single step) from every other state of the model. (Strictly speaking an ergodic model has the property that every state can be reached from every other state in a finite a number of steps).



Fig, 3.7. Illustration example of types of HMMs.
(a) A 4-state ergodic model.
(b) A 4-state left-right model.

As shown in fig 3.7a, for an N = 4 state model this type of model has the property that every a_{ij} coefficient is positive. Hence, for the example of Fig. 3.7(a) we have

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

For some applications, other types of HMMs have been found to account for observed properties of the signal being modeled better than the standard ergodic model. One such model is shown in Fig 3.7(b). This model is called a left-right model or Bakis Model because the underlying state sequence associated with the model has the property that as time increases the state index increases (or stays the same), i.e. the states proceed from left to right. Clearly the left-right type of HMM has the desirable property that it can readily model signals whose properties change over time. E.g. Speech. The fundamental property of all left-right HMMs is that the state transition coefficients have the property

$$a_{ij} = 0, \quad j < i \tag{3.45}$$

i.e. no transitions are allowed to states whose indices are lower than the current state. Furthermore, the initial state probabilities have the property

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases} \tag{3.46}$$

since the state sequence must begin in state 1 (and end in state N). Often, with left-right models, additional constraints are placed on the state transition coefficients to make sure that large changes in state indices do not occur; hence a constraint of the form

$$a_{ij} = 0, \quad j > i + \Delta \tag{3.47}$$

is often used. In particular, for the example of Fig 3.7(b), the value of Δ is 2, i.e., no jumps of more than 2 states are allowed. The form of the state transition matrix for the example of Fig. 3.7(b) is thus

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

It should be clear that, for the last state in a left-right model that the state transition coefficients are specified as

$$a_{NN} = 1 \tag{3.48a}$$

$$a_{Ni} = 0, \text{ for } i < N. \tag{3.48b}$$

It should be clear that the imposition of the constraints of the left-right model, or those of the constrained jump model, essentially have no effect on the reestimation procedure.

3.8 Limitations of HMMs

Although use of HMM technique has contributed greatly to recent advances in speech recognition, there are some inherent limitations of this type of statistical model for speech. A major limitation is the assumption that successive observations (frames of speech) are independent, and therefore the probability of a sequence of observations P(O_1, O_2 ... O_T) can be written as a product of probabilities of individual observations, i.e.,

$$P(O_1, O_2, ...O_T) = \prod_{i=1}^{T} P(O_i)$$

Another limitation is the assumption that the distributions of individual observation parameters can be well represented as a finite mixture of Gaussian densities**.** Finally the Markov assumption itself, i.e., that the probability of being in a given state at time t only depends on the state at time t - 1, is clearly inappropriate for speech sounds where dependencies often extend through several states. However, in spite of these limitations this type of statistical model has worked extremely well for speech recognition problems.

3.9 Acoustic and Language Models in Speech Recognition

For a given acoustic observation $O = O_1, O_2, ... O_T$ the goal of speech recognition is to find out the corresponding word sequence $W = W_1W_2...W_m$ that has the maximum posterior probability P(W | O) as expressed by the following equation (3.49).

$$W = \arg\max P(W \mid O) = \arg\max \frac{P(W) \cdot P(O \mid W)}{P(O)} \qquad (3.49)$$

Since the maximization Eq. (3.49) is carried out with the observation O fixed, the above maximization is equivalent to maximization of the following equation.

$$W = \arg\max P(W) \cdot P(O \mid W) \qquad (3.50)$$

where the first term P(W) is the language model

and the second term , $P(O \mid W)$, is the acoustic model.

The practical challenge is how to build accurate acoustic models, $P(O|W)$ and language models, P(W) that can truly reflect the spoken language to be recognized. For large vocabulary speech recognition, since there are number of words, we need to decompose a word into a sub word sequence. So $P(O|W)$ is closely related to phonetic or syllable modeling. $P(O|W)$ should also take into account speaker variation, pronunciation variations, environmental variations and context dependent phonetic coarticulation variations.

The decoding process of finding the best word sequence W to match the input speech signal O in speech recognition systems is much more than simple pattern recognition problem, since in continuous speech recognition there are an infinite number of word patterns to search. But in isolated word recognition the decoding process becomes simple

pattern matching.

(In this thesis, acoustic models are done by training HMMs, which has been explained thoroughly in the above sections. But we will not delve into language models and their

representation in speech recognition because the practical application software developed is for Isolated Word Recognition.)

Chapter Four

Implementation and Experimentation

4.1 Implementation of Speech Recognizers Using HMMs

The purpose of this, and the following sections, is to illustrate how the ideas of HMMs, as discussed in the earlier chapter of this thesis, have been applied to selected problem in speech recognition.

Our main goal here is to show how specific aspects of HMM theory get applied in speech recognition technology.

4.2 Overview of General Recognition System

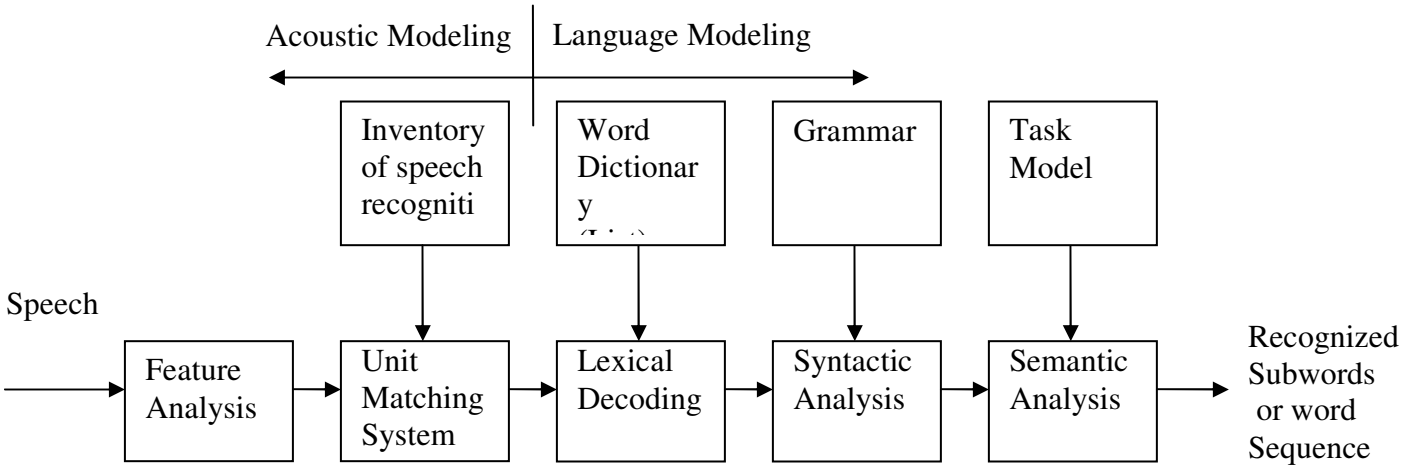


Figure 4.1 shows a block diagram of speech recognition system.

The above diagram shows the overall block diagram of continuous speech recognition system. The key signal processing steps include the following:

1) Signal Processing/Extracting Features*:* A spectral and/or temporal analysis of the speech signal is performed to get observation vectors which can be used to train the HMMs which characterize various speech sounds**.** The role of the signal processing module is to reduce data rate, to remove noises, to extract salient features that are useful for subsequent acoustic matching.

2) *Unit Matching System*: First a choice of speech recognition unit must be made. Possibilities include linguistically based sub-word units such as phones (or phone-like units), triphones, and syllables. Other possibilities include whole word units, and even units which correspond to a group of two or more words (e.g., and an, in the, of a, etc). Generally, the less complex the unit (e.g., phones), the fewer of them there are in the language, and the more complicated (variable) their structure in continuous speech. For large vocabulary speech recognition (involving 1000 or more words), the use of sub-word speech units is almost mandatory as it would be quite difficult to record an adequate training set for designing HMMs for units of the size of words or larger. However, for specialized applications (e.g., small vocabulary, constrained task), it is both reasonable and practical to consider the word as a basic speech unit. Independent of the unit chosen for recognition, an inventory of such units must be obtained via training. Typically each such unit is characterized by some type of HMM

whose parameters are estimated from a training set of speech data. The unit matching system provides the likelihood of a match of all sequences of speech recognition units to the unknown input speech.

3) ***Lexical Decoding:*** This process places constraints on the unit matching system so that the paths investigated are those corresponding to sequences of speech units which are in a word dictionary (a lexicon). This procedure implies that the speech recognition word vocabulary must be specified in terms of the basic units chosen for recognition. In the case where the chosen units are words (or word combinations), the lexical decoding step is essentially eliminated and the structure of the recognizer is greatly simplified.

4) ***Syntactic Analysis****:* This process, much like lexical decoding, places further constraints on the unit matching system so that the paths investigated are those corresponding to speech units which comprise words (lexical decoding) and for which the words are in a proper sequence as specified by a word grammar. Such a word grammar can again be represented by a deterministic finite state network (in which all word combinations which are accepted by the grammar are enumerated), or by a statistical grammar (e.g., a trigram word model in which probabilities of sequences of 3 words in a specified order are given). For some command and control tasks, only a single word from a finite set of equiprobable is required to be recognized and therefore the grammar is either trivial or unnecessary. Such tasks are often referred to as isolated word speech recognition tasks. For other applications (e.g., digit sequences) very simple grammars are often adequate (e.g., any digit can be spoken and followed by any other digit). Finally there are tasks for which the grammar is a dominant factor and, although it adds a great deal of constraint to the recognition process, it greatly improves recognition performance by the resulting restrictions on the sequence of speech units which are valid recognition candidates.

5) ***Semantic Analysis:*** This process, again like the steps of syntactic analysis and lexical decoding, adds further constraints to the set of recognition search paths. Depending on the recognizer state certain syntactically correct input strings are eliminated from consideration. This again serves to make the recognition task easier and leads to higher performance of the system.

4.3 Overview of the practical project

As mentioned in the first chapter only part of the Ge'ez characters (Fig 1.3) is used in the training process. From these characters, words of the following kind can be tested for recognition.

›²² ›cu gg ckK S[S[.....

It is to be noted that one can hardly make a sentence out of these words (although these words alone can be viewed as a degenerate one word sentence). Continuous unseparated utterance of these words does not make sense. So the application to be developed is bound to be **Isolated Word Recognition**. No sentence also means that no Syntactic and Semantic analysis shown in the above diagram are required.

So the simplified diagram will be as follows:

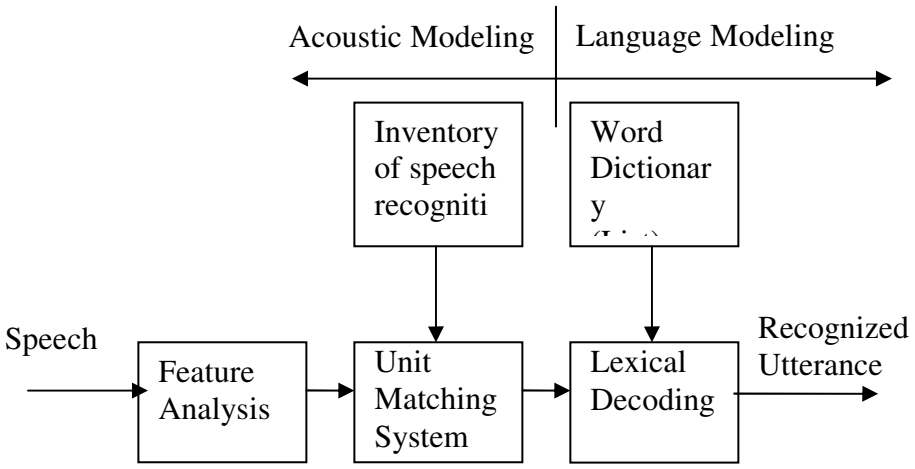


Figure 4.2 shows Block diagram of the practical project for the isolated speech recognition.

4.3.1 Operation of Practical Project

Generally to activate speech signal capture in speech recognition systems, you can use either *push to talk* or *continuously listening* mode. Push to talk uses a special push event to activate and deactivate speech capture, which is immune to potential background noise and can also eliminate use of processing resources to detect speech events. The continuously listening model listens all the time and automatically detects whether there is speech signal or not. This mode makes use of speech-end-

point-detector, which is typically energy threshold based end point detection. It is not critical for the automatic end point detector to offer exact end point accuracy. The key feature required of it is not to interpret speech segments as silence or noise segments. On the other hand if the end point detector interprets noise segments as part of speech, these may be rescued by the speech recognizer if the recognizer has appropriate silence or noise model.

In noisier environments, though, this energy based end point detector does not work properly. So it is assumed a silent environment is required for the application to run properly.

In this Practical Project the **continuously listening** mode is implemented.

4.3.2 Selecting appropriate units for modeling

For Large vocabulary speech recognition, it is difficult to build whole word models because

- There are simply too many words. It is unlikely that we have sufficient recordings (repetitions) of those words to train word models.
- There could be new words, newly invented names etc

There are a number of issues to consider in choosing appropriate modeling units.

- The unit should be **accurate**, to represent the acoustic realization that appears in different contexts.
- The unit should be **trainable**. We should have enough data to estimate the parameters of the unit.
- The unit should be **generalizable**, so a new word can be derived from a predefined unit inventory for speech recognition.

The following table shows comparison of modeling units for large vocabulary and small vocabulary speech recognizers.

Criteria	Sub words(phones	Words
Accuracy	Yes	Yes
Trainability	Yes	Yes/no
Generalizable	Yes	No
	Large Vocabulary	Small Vocabulary

Table 4.1 the above table shows parameters considered in selecting speech units for modeling.

Since this practical project started out as medium to large vocabulary speech recognition system, **Sub word (syllables)** are selected as a unit for modeling.

4.3.3 Signal acquisition

Today's computer can handle the necessary speech signal acquisition tasks in software. For example, most PC sound cards have direct memory access, and speech can be digitized in memory without burdening the CPU with input/output interrupts. The operating system can handle most of the AD/DA functions in real time. In practice, 16 KHz is sufficient for speech bandwidth (8 KHz). Reduced bandwidth, such as the telephone channel, generally increase speech recognition error rate.

The following table shows empirical relative recognition error reduction using different sampling frequencies.

Sampling	Relative Error Rate Reduction
8 KHz	Base line
11 KHz	+10 %
16KHz	+10 %
22 KHz	+0 %

Table 4.2 the above table shows empirical results of relative error rate reduction for different sampling frequency.

In this practical project, two cases of sampling rate of **16 KHz and 8 KHz** are used for signal acquisition during training and testing.

4.3.4 Making the database (Corpus Data)

Statistical approach to speech recognition requires a large amount of data for training. These corpus data then needs to be segmented and labeled manually according to the chosen units for modeling. Normally this is done by recording someone while reading a collection of sentences or paragraph, which should be phonetically balanced.

The above method, although a standard in building Corpus Data, is a labor-intensive task. Therefore, what is done in this project as alternative is to record isolated utterances of the selected characters (syllables) and use end point detection method to make segmentation and labeling (annotation)

automatic. This procedure has its shortcomings i.e. that a character (syllable) uttered in isolation will not have the same acoustic features as where the same character is uttered in a word due to coarticulation effects discussed in chapter two. This will have a negative impact on the recognition but it is time efficient way to make the corpus data.

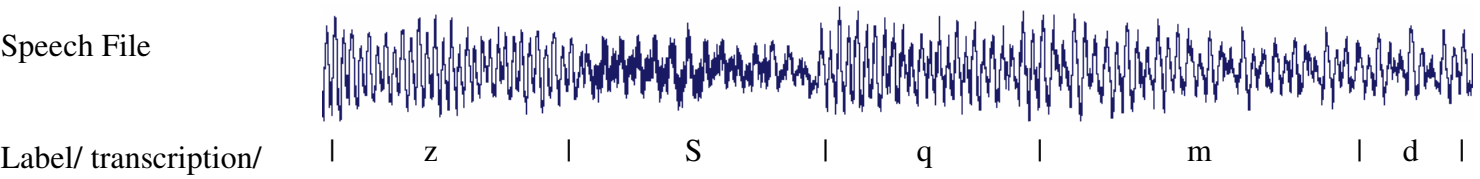


Figure 4.3 Shows the way recorded speech file is segmented. The segmented portions should also be annotated.

If speech recognizer is to be speaker independent, the corpus data should be built from various speakers: male, female, younger, elder etc. If the speech recognizer is also to be environmentally robust, the recording should be done on various environmental conditions. These two tasks cannot be accomplished due to the constraint of time. So the application developed will be **speaker dependent** and requires a fairly **silent environment** for operation.

4.3.5 Coding the Data/ Extracting Features

The final stage of data preparation is to parameterise the raw speech waveforms into sequences of feature vectors (as described in chapter 3). The figure below shows how to get the sequences of speech data vectors: LPC or MFCC depending on the feature extraction used.

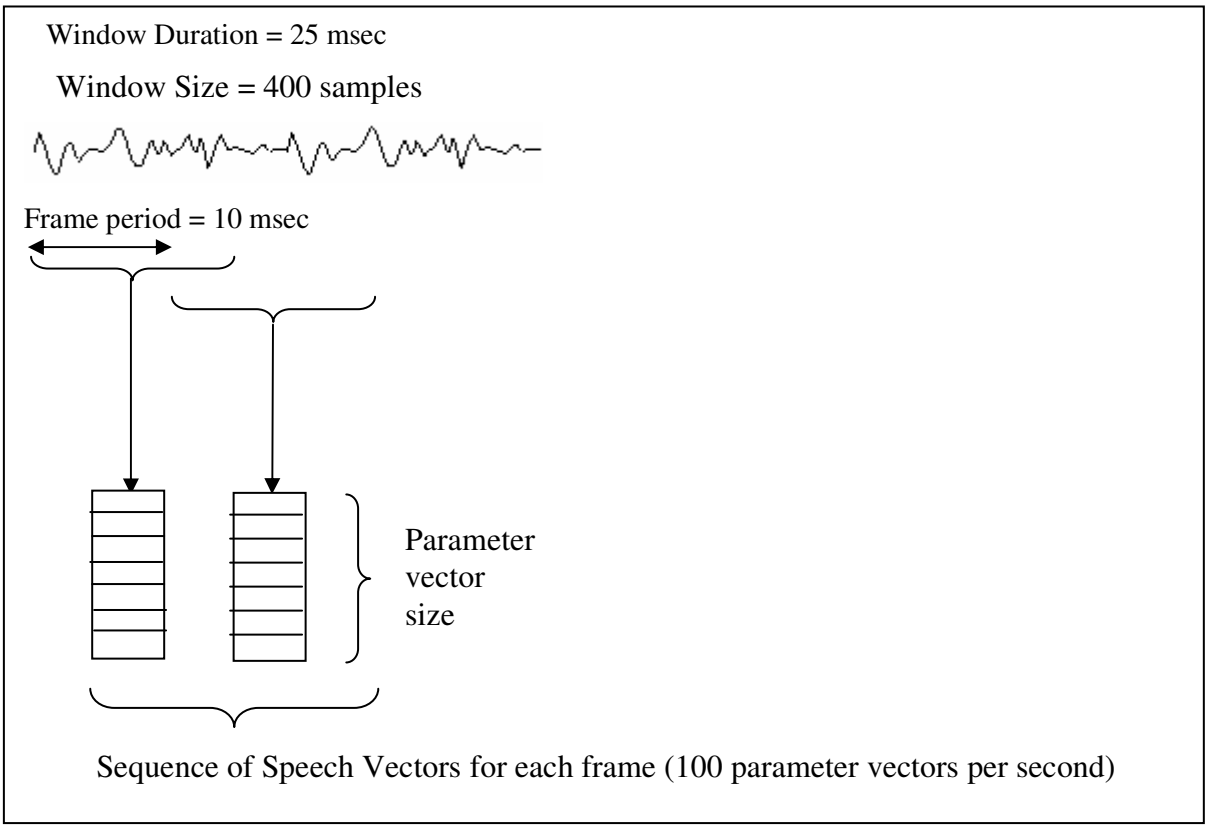


Figure 4.4 feature extraction. A window of size 400 samples is converted to a vector size of 13.

4.3.6 Training the models

The figure below illustrates how training is done for each of the characters (syllables) using the Baum-Welch method. This is an iterative procedure. Once the training is done the models must be tested. If the recognition accuracy level is not satisfactory more training data must be recorded. This process must continue until the desired recognition accuracy is achieved.

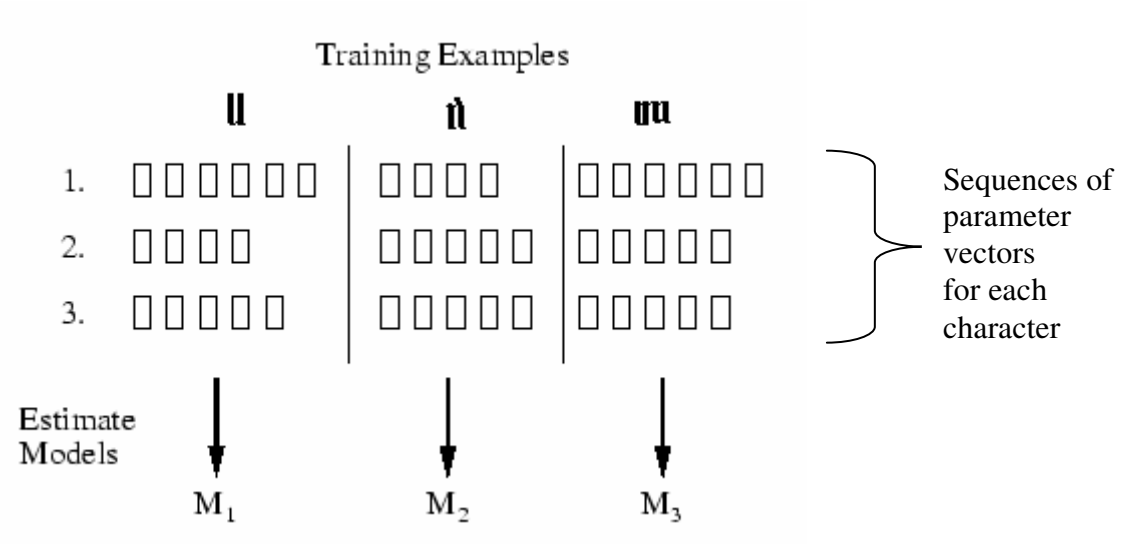


Figure 4.4 Sequences of vectors are used to train the models.

It should be noted that silence is also modeled in the above procedure.

4.4 Isolated word recognition

Consider using HMMs to build an isolated word recognizer. Assume we have a vocabulary of V words to be recognized and that each word is to be modeled by a distinct HMM. Further assume that for each word in the vocabulary we have built a model for each word by concatenation the sub word models.

In order to do isolated word speech recognition, we must perform the following:

1) For each word *v* in the vocabulary, we must build an HMM λ^v,
2) For each unknown word which is to be recognized, the speech processing of Fig. 4.5 must be carried out, namely measurement of the observation sequence $O = O_1, O_2, \ldots O_T$ via a feature analysis of the speech corresponding to the word; followed by calculation of model likelihood for all possible models, $P(O \mid \lambda^v)$, $1 \le v \le V$, followed by selection of the word whose model likelihood is highest, i.e.,

$$v' = \arg\max[p(O \mid \lambda^v)] \qquad 1 \le v \le V \qquad 4.1$$

The probability computation step is generally performed using the Viterbi algorithm (i.e., the maximum likelihood path is used).

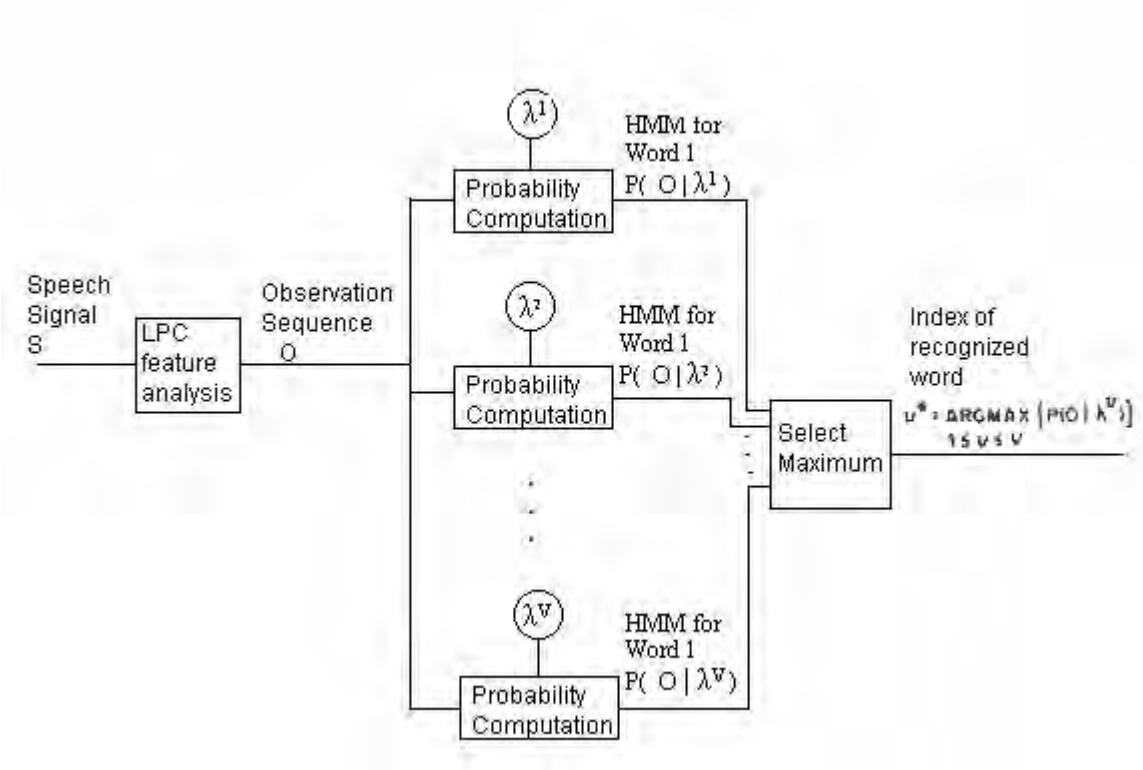


Figure 4.5 Block diagram of isolated word HMM recognizer.

4.5 Testing and Evaluation of Practical Project

The most important measure in speech recognition is the word error rate.

$$\textbf{Word error rate} = \frac{\textbf{Substituted words}}{\textbf{No. of words in the test}} \qquad 4.2$$

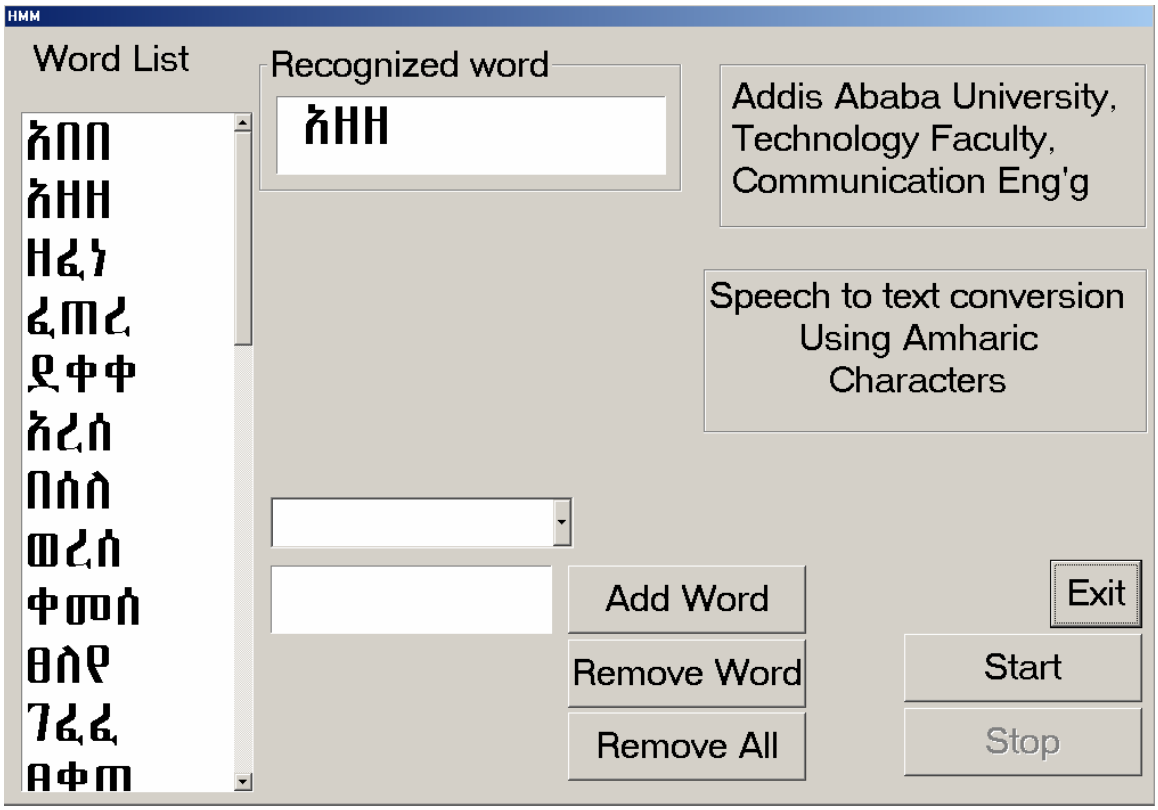
The table below shows word error rate for different sampling frequencies and number of words. This result shows that word error rate increases with number of words. It depends also on the sampling frequency; better performance is achieved at 16 KHz than at 8 KHz.

Sampling	Number	Word Error
8KHz	10	<5%
8KHz	25	15%
8KHz	50	30%
16KHz	10	<5%
16KHz	25	10%
16KHz	50	25%

Table 4.3 the above table shows empirical results of word error rate.

4.6 User Interface of the Practical Project

The user interface of the project is as shown below. Once the start button is pressed the program will listen for speech sound at the microphone. After utterance is made at the microphone, the program will output the recognized word in the "Recognized word" edit box.



Word list: This List box contains selected words for recognition.

Recognized Word: This edit box displays the recognized word after a single utterance.

Add Word: to add a new word press the "Add Word" button after writing a word to be included in word list to the edit box left to this button.

Remove Word: press "Remove Word" button to remove word from the word list after selecting the word to be removed.

Remove All: press "Remove All" button to remove all words from the word list.

Start: Press "Start" button to start recognition.

Stop: Press "Stop" button to stop recognition.

Exit: Press the "Exit" button to exit application.

Chapter Five

Conclusion

5.1 Context Independent training data

Building the training corpus data is mandatory step in speech recognition. Generally, the corpus data is made by *manually* segmenting and labeling each and every character (phones/syllables) out of recorded *continuously* spoken speech. Segmentation and labeling is done by the help of time waveform and the spectrogram in some difficult situation. The training corpus should also be as large as possible to make an accurate estimation of the model. In this project, the need for large amount of data has led to development of the corpus data through the utterance of each and every character in isolated manner so that end point detection algorithm be used in segmenting and labeling the corpus data automatically without requiring *manual* work. This has made the corpus data context independent. The effect of the context independent corpus data on model accuracy will be discussed below.

-The duration factor

The figure below shows the utterance of the characters ሸ and ሽ isolation followed by the word utterance ሸሽሽ.

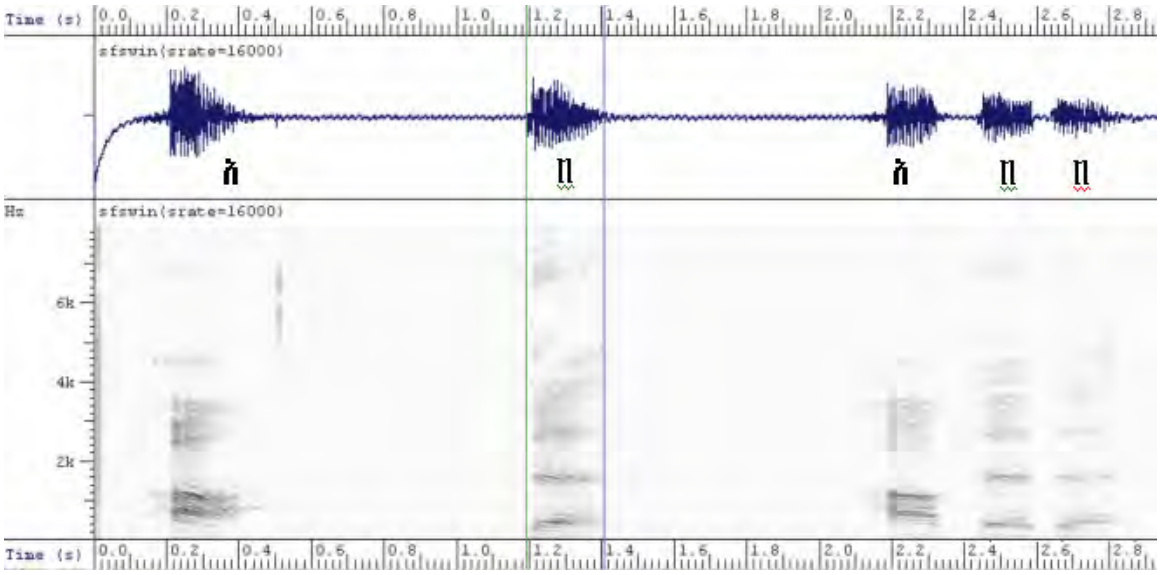


Figure 5.1 shows the difference in duration of characters in isolated vs. word utterances.

Even though the spectral characteristics of these characters are similar, it can be clearly seen that the characters uttered in isolation take longer duration than the characters in word utterance (the

character ጵ has duration of 220ms vs. 150ms: the character ፐ has duration of 200ms vs. 160ms in isolation and word utterance respectively). This difference in duration will decrease the accuracy of the models since the duration of the characters of the training corpus data have a direct effect on the state transition probabilities of the models during training.

-The coarticulation effect

In the figure above, the characters in isolation and word utterances differ not only in duration but they also differ at some level in spectral characteristics. This is attributable to the coarticulation effect. The coarticulation effect, as discussed in chapter two, is the effect of neighboring phones/syllables on the acoustic traits (spectral characteristics) of characters in word utterance (continuous speech). The corpus data of isolated utterance is then context independent since no neighboring phones/syllables exist. In consequence the resulting model will also be context independent. This will have a negative impact on the accuracy of the model and the resulting recognition performance.

5.2 Amharic Vs other languages in speech recognition

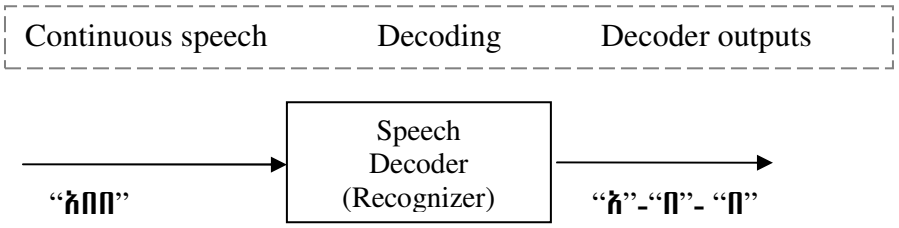
As far as spoken language units and written language units are concerned there is one to one correspondence in Amharic Language which is not present in other languages, for example, the English Language. In Amharic, spoken language units which are the syllables have direct relations with written Amharic language units which are denoted by these Ge'ez characters, ሀ ሰ ..... ፐ. But the English language, for example, does not have a one to one correspondence between the spoken language units (which are about 50 phonemes and the written English language units (which are the English alphabet A, B…..Z). The following table shows the relations between spoken subword units and written alphabetic units.

Amharic Language		**English Language**	
Syllables	**word**	**phonemes**	**word**
ፐ	**ጵፐፐ**	iy	f**ee**l,
ሰ	**ከሰሰ**	eh	p**e**t,
		ih	f**i**ll, h**i**t,
		th	**th**in,
		dh	**th**en,

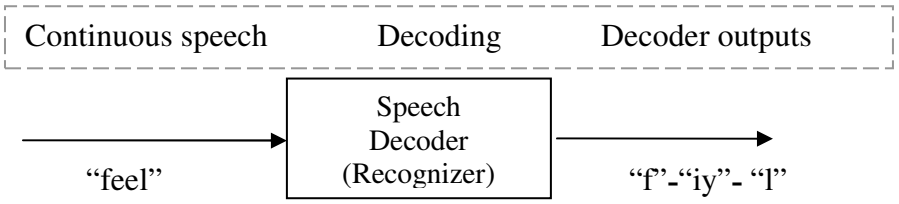
Table 5.1 Comparison of Amharic and English spoken and written language units.

The fact that there is one to one correspondence between spoken and written language units in Amharic should not leads us to conclude that each character in a word or continuous speech is read or spoken by independent concatenation of sequence of the characters. These characters will not be produced independently because our articulators can not move instantaneously from one position to another.

The other fact that should be stated is, in speech to text conversion, the one to one correspondence between spoken and written language units in Amharic might give the idea that this language is suited to speech to text conversion than other languages. But this fact, while facilitate the text rendering process, has nothing to do with being suited to speech to text conversion or performance of the speech recognition. The following diagram shows the decoded speech to text conversion for Amharic and English language for comparison.



Amharic Language decoding (syllable model used)



English Language decoding (phonetic model used)

Figure 5.2 Speech to Text decoder outputs for Amharic and English Languages

The above diagram shows the output of the decoders for Amharic and English languages. In the Amharic Language case, the output of the decoder (syllables) matches characters from the Ge’ez alphabet, so rendering becomes easy. But in the case of the English language, there is a need to maintain a table of that matches decoder outputs (sequence of phones) with written form of the word as shown in the following example.

Sequence of	Written form of
"f"-"iy"- "l"	feel
"p"-"eh"-"t"	pet
"dh"-"eh"-"n"	then
Etc …	

Table 5.2 shows a mapping between sequence of phones and written word.

5.3 Chosen units for modeling and other languages

The units used in speech recognition application could be phones, syllables or words. The chosen units for modeling depend on the application being developed. For example, word models are preferred to phone or syllable models in the case of very small vocabulary. But in the case of large vocabulary, subword units are used because one can not afford to train all the words in the vocabulary. The subword unit case leaves us with phones or syllables. Syllables are preferred to phones because syllables, which can be viewed as intermediate units between phones and word, and being larger units, would avoid contextual dependencies in the central portions of this unit.

Different languages have different number of phones and syllables. The Amharic language has about 200 syllables, 1200 syllables in Chinese and approximately 50 syllables in the Japanese which makes syllable a suitable unit for these languages. But for the English Language the large number of syllables (over 30,000) presents a challenge in terms of trainability. So phones are used as modeling units in English Language.

5.4 Conclusion

Recognizing speech remains a difficult problem. To understand speech, a human considers not only the specific information conveyed to the ear, but also the context in which the information is being discussed. For this reason, people can understand spoken language even when the speech signal is corrupted by noise. However, understanding the context of speech is, in turn, based on a broad knowledge of the world. And this has been the source of the difficulty.

It is difficult to develop computer programs that are sufficiently sophisticated to understand continuous speech by a random speaker. Only when programmers simplify the problem – by isolating words, limitation the vocabulary or number of speakers, or constraining the way in which sentences may be formed – is speech recognition by computer possible.

In most of the cases of implementing speech recognition, a statistical approach is used most often and this approach had given promising results in the recent past. The practical project, taking the Amharic language as an experiment, is an attempt to understand and evaluate the statistical approach (HMM) to speech recognition. Despite the lack of sufficient Corpus Data (recorded speech for training), a promising result was achieved by implementing the statistical approach.

But there is still a long way to go as the following table shows the achievements in this field of study for the English Language.

Task	**Vocabulary**	**Word Error Rate by Machines**
Digits (1 – 10)	10	0.72%
Continuous telephone	2000	36.7%
Dictation (isolated	5000	4.5%

Table 5.3 shows examples of speech recognition word error rate for English Language.

5.5 Recommendation

This is an initial attempt and it can be improved in many ways.

- The corpus data should be as large as possible because in this statistical approach or in the Hidden Markov Model there are probability parameters that need to be estimated.
- The corpus data should be recorded in various environmental conditions to improve performance. Environmental robustness could also be improved by using adaptation methods.
- The corpus data should be recorded from various audiences to make the speech recognizer speaker independent.
- The corpus data should include all the Ge'ez Letters. With this it is possible to make all possible words or may be sentences depending on the application.
- The corpus data, in this practical project, was made to represent the syllables. An alternative approach may be to segment the training speech files in to phones.
- The corpus data should be made from continuous utterance of some written sentence or paragraph. This ensures the capture of speech segments as it is in the natural speech utterance and the models developed will be context dependent.
- In this Practical work of statistical approach HMMs are used but there are also developments using neural networks and mixed types which contain both HMM and neural networks.

- In this application, a conventional Hidden Markov Model is used. But a modified HMM, for Example, inclusion of separate state duration model can improve the performance of speech recognizer.

REFERENCE

[1] Xuedong Huang, Alex Acero, Hsiao-wuen Hon "Spoken language processing"
2001 Prentice Hall PTR

[2] Daniel Jurafsky and James H.Martin "Speech and Language Processing"
2000 Prentice Hall , Upper Saddle River, New Jersey

[3] Christopher D.Manning and Hinrich Schutze "Foundations of Statistical Language Processing"
2002 The MIT Press

[4] Kenneth N. Stevens "Acoustic Phonetics"
2000 MIT Press

[5] Eleanor Kutz "Language and Literacy"
1997 Boynton/Cook Publishers

[6] D.G. Childers "Speech Processing and Synthesis Toolboxes"
2000, John Wiley and Sons

[7] John R. Deller, John H. L. Hansen "Discrete-Time Processing of Speech Signals"
John G. Proakis
1993, Macmillan, New York

[8] N. Rex Dixon, Thomas B.Martin "Speech and Language Processing"
IEEE press (Institute of Electrical and Electronics Engineers Inc. New York)

[9] Theiry Dutoit "An Introduction to Text-To-Speech Synthesis"
1997, Kluwer Academic Publishers

[10] Lawrence R. Rabiner "Digital Processing of Speech Signals"
Prentice Hall, Upper Saddle River, New Jersey

[11] Ronald W. Schafer, John D. Markel "Speech Analysis"
IEEE Press

[12] Lawrence R. Rabiner "A Tutorial on Hidden Markov Model"
IEEE Press

[13] Fant, G. "Acoustic Theory of Speech Production"
1970, The Hague, NL, Mouton

[14] Huang X.D., Y. Ariki and M.A. Jack "Hidden Markov Models for speech Recognition"
1990 Edinburgh, U.K. Edinburgh University Press

[15] Jelenik, F., "Statistical Methods for speech Recognition"
1998, Cambridge M.A. MIT Press.

[16] Acero, A., "Acoustical and Environmental Robustness in Automatic Speech Recognition"
1993 Boston, MA, Kluwer Academic Publishers.

Web Links

[17] http://svr-www.eng.cam.ac.uk/~ajr/SpeechAnalysis/ Dec 1/2005

[18] http://www.speechandhearing.net/laboratory/tools.html Dec 1/2005

[19] http://www.phon.ucl.ac.uk/resource/sfs/rtspect/ Dec 1/2005

[20] http://www.phon.ucl.ac.uk/resource/sfs/enhance.htm Dec 1/2005

[21] http://www.phon.ucl.ac.uk/resource/sfs/wasp.htm Dec 1/2005

[22] http://www.speech.kth.se/wavesurfer/ Dec 1/2005

[23] http://www.sil.org/computing/speechtools/speechanalyzer.htm Dec 1/2005

[24] http://www.phon.ucl.ac.uk/resource/sfs/ Dec 1/2005

[25] http://www.fon.hum.uva.nl/praat/ Dec 1/2005

[26] http://htk.eng.cam.ac.uk/ Dec 1/2005

[27] http://cslu.cse.ogi.edu/toolkit/index.html Dec 1/2005

[28] http://www.microsoft.com/speech/default.mspx Dec 1/2005

[29] http://tcts.fpms.ac.be/synthesis/mbrola.html Dec 1/2005

[30] http://www.research.att.com/sw/tools/fsm/ Dec 1/2005

[31] http://www.cstr.ed.ac.uk/projects/speech_tools/ Dec 1/2005

[32] http://www.cavs.msstate.edu/hse/ies/projects/speech/index.html Dec 1/2005

[33] http://cmusphinx.sourceforge.net/html/cmusphinx.php Dec 1/2005

[34] http://mi.eng.cam.ac.uk/~prc14/toolkit.html Dec 1/2005