A Study in Speaker Dependent Medium Vocabulary Word Recognition: Application to Human/Computer Interface

BY

Moatassem M. Abdallah Abd El Salam

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Electrical and Computer Engineering

APPROVED

_____

Dr. Hugh F. VanLandingham, Chairman (ECPE)

_____ _____

Dr. Sedki M. Riad (ECPE) Dr. Richard L. Moose (ECPE)

_____ _____

Dr. John W. Roach (CS) Dr. A. Lynn Abbott (ECPE)

January 2000
Blacksburg, Virginia

Keywords: Speech Processing, Modular Neural Network, Human/Computer interface

Acknowledgment

I would like to express my gratitude to my advisor, Dr. Hugh F. VanLandingham, for his valuable supervision, support and encouragement throughout my Ph.D. research at Virginia Tech. I have been happy to work under his guidance and have benefited greatly from his advice.

I am deeply grateful to Dr. Sedki M. Riad for his strong support in every possible way.

I also would like to thank Dr. Richard L. Moose, Dr. John W. Roach, and Dr. A. Lynn Abbott who served on my advisory committee. Their time and suggestions towards my dissertation are sincerely appreciated.

I cannot find the words to express my profound feelings of gratitude for my father Dr. Mahmoud Abd El Salam and my mother who have guided me throughout my life.

My greatest thanks goes to my brother Eng. Motaz M. Abdallah for his moral support.

Finally, I want to dedicate this dissertation to my dearest family. Without the love, constant support, and encouragement of my wife Sherine M. El Adel, my education would have never been possible. I would also like to thank my children Mohanad and Hanya for making my graduate years most enjoyable ones.

CONTENTS

List of Figures

List of Tables

Chapter 1

Introduction to Speech Recognition

Speech recognition aims to ascertain and assess the linguistic information comprised in speech, i.e., to comprehend the input speech signal adequately to select a suitable response.

This objective has several difficulties

- Different people speak in different ways.
- Even a single speaker is prone to variations in his pronunciations.
- A noisy acoustic environment may interfere with an otherwise reliable interpretation of the acoustic speech signal.
- The complexities of naturally flowing connected speech.

These problems can be limited by restricting the population of speakers, working with good (controlled) acoustic conditions and avoiding the complexities of fluent speech. This would make the recognition process independent of the variables presented by a speaker identity, manner of speaking and environmental conditions.

1.1 Terminologies of Speech Recognition Systems

The first requirement in the design of a speech recognition system is to define clearly what particular aspects of speech are to be recognized. A brief summary of each aspect of the speech recognition problem in general is given below

* Isolated Word Recognition

Recognition of this type requires the speaker to utter words with pauses after each word, this is adequate for a variety of applications such as package sorting, training air traffic controllers and cockpit communications. However, it is far from being a natural way of communicating.

* Connected Word Recognition

For the connected word recognition problem, it is assumed that a set of reference patterns are available for each unit of the vocabulary (generally the units are isolated words). The connected word problem can be matched by a sequence of isolated reference patterns. Hence, in connected word recognition, the task of spotting the beginnings and ends of words is of major importance.

* Continuous Speech Recognition

For continuous speech recognition, there is generally no fixed set of reference patterns. Instead of *dynamic time warping* for alignment (which is used for both isolated and connected word recognition) a segmentation and labeling scheme is used to provide an estimate of the spoken utterance. Hence, major differences exist in both concept and implementation for connected word and continuous speech recognition systems.

* Word Spotting

A further voice problem is that of ignoring all speech that is uttered until a keyword is spoken. This technique has been used in the military environment and has obvious applications in the sifting of large amounts of archived audio material for relevant sections of conversation. The systems used are derivatives of speech recognizers, tuned to reject any words that do not have a very high correlation to a keyword.

* Speech Understanding Systems

In addition to recognizing vowels, consonants, words and other subunits of speech, systems of this type are also able to “understand” the input. Hence, they are known as “speech understanding systems”. Such systems are usually associated with Artificial Intelligence (AI) techniques used in connected natural language understanding systems. These techniques attempt to interpret the meaning of speech signals using contextual "knowledge" about speech, rather than simply matching patterns as in speech recognition systems. Thus, the basic difference between speech understanding and speech recognition is that speech understanding systems employ the principles of AI, while speech recognition systems typically do not. Speech understanding implies some recognition of the input in terms of external response or internal change.

* Speaker Identification

Speaker identification uses techniques, which are very closely related to those of speech recognition. However, while speech recognition aims to ignore the differences between speakers and recognize only the difference between words, speaker identification aims to highlight the differences between speakers.

1.2 Applications of Speech Recognition

Although commercially available speech recognition systems nowadays are perhaps no more than those based on simple template matching schemes for Isolated Word Recognition (IWR) or Connected Word Recognition (CWR - in the case of small vocabularies), speech recognition is becoming widely applied in commerce, defense and industry. As yet, however, the successful techniques used are not found in the open literature.

The application of speech recognition systems in these areas provides the following advantages [1]: -

- Hands free operation
- Eyes up operation
- Mobility
- Unlimited soft-key facility
- Simple hardware interface
- Rapid entry
- Ease of use
- Thought into action

Further details about the current applications of speech recognition systems together with speech synthesis (speech technology) are given in references [2, 3, 4].

Some of the areas in which speech recognition systems are now being applied to man-machine interactions are detailed below:

* Industry

Applications in industry include:

- Product inspection
- Storage and warehousing
- Robotics [5]
- CAD/CAM/CAE [6]

* Aviation and Space

There is particular interest in the use of speech recognition in avionics, both military and civil, to reduce the workload of pilots. Speech commands can be used to replace sequences of keystrokes (serving an equivalent purpose to that of dedicated function keys on a keyboard) allowing pilots to direct their vision elsewhere and to use their hands for other jobs. Many of the American and European research programs in speech recognition or Direct Voice Input (DVI) focus attention on its application to military and civil avionics. In general, research projects aim to communicate with the aircraft subsystems (e.g. on board weapon systems) via direct speech commands.

Speech recognition is also applied in the most challenging field of human endeavour, space exploration. The operation is again a typical eye and hand busy situation for which the speech recognition system aims the visual and manual loads of the pilots.

An example for this application is to control the space shuttle closed circuit TV (CCTV) by speech, while the operator has to control the Remote Manipulator System (RMS) at the same time with both hands. Another example is the use of speech commands to obtain information about a failed system, instead of the operator having to leave his place to get this information from a Multifunction Cathode-ray-tube Display System (MCDS).

* Communications

Speech recognition together with speech synthesis can be used as a low bit rate communication system. This system has the advantage of securing the communications for application such as ground-to-air communications especially in military environments since the speech is easily encoded.

* Voice Dialing Telephones

Voice dialing telephones are necessary when the user's hands and eyes are busy.

* Applications for the Disabled

Applications include:

- Control of house equipment
- Voice-controlled wheelchairs
- Conversion of spoken messages to written texts for deaf persons to communicate with others through speech

* Bank and Office Automation

* Speaker Verification or Identification

This application is required for security purposes.

* Word Spotting

Word spotting is a useful means for securing military communications or command issue.

* Spoken Message Translators

One of the most significant (and potentially, one of the most difficult problems to solve) is the direct translation of spoken messages-allowing telephone communication to be established between two persons using different languages.

1.3 Approaches to Machine Speech Recognition

Three main approaches to speech recognition can be identified from the literature, namely: -

- The acoustic-phonetic approach
- The pattern recognition approach
- The artificial intelligence approach

We may also consider artificial neural networks as a fourth approach.

1.3.1 The Acoustic-Phonetic Approach

This approach is based on the assumption that there exist finite, distinctive phonetic units in spoken language and that the phonetic units are broadly characterized by a set of properties that are apparent in a speech signal, or its spectrum over time. It is presumed that the rules that dominate the variability can be determined. The acoustic properties of phonetic units are highly variable, both with speakers and with neighboring phonetic units (called coarticulation of sound). The segmentation and labeling phase is considered the first step in the acoustic-phonetic approach to speech recognition because it involves segmenting the speech signal into discrete (in time) regions where the acoustic properties of the signal are typical of one (or possibly several) phonetic units (or classes), and then attaching one or more phonetic labels to each segmented region according to its acoustic properties. The second requisite step to perform speech recognition attempts to determine a valid word (or string of words) from the sequence of phonetic labels produced in the first step, which should conform with the constraints of the speech-recognition task.

The main problem is the difficulty in getting a reliable phoneme lattice for the lexical access stage. Figure 1.1 shows a block diagram of the acoustic-phonetic approach to speech recognition.

1.3.1.1 Feature Measurement Phase

This is the first step in the processing, the feature measurement method, which provides an appropriate representation of the characteristics of the time-varying speech signal. The most common techniques of spectral analysis are the class of filter bank methods and the class of Linear Predictive Coding (LPC) methods. They both provide spectral descriptions of the speech over time.

1.3.1.2 Feature-Detection Phase

The second step in the processing is the feature-detection stage, by which we try to convert the spectral measurements to a set of features that describe the broad acoustic properties of the different phonetic units. The presence, or absence, of nasal resonance or so called nasality is one of the features proposed for recognition. Another feature is friction which is the presence or absence of random excitation in the speech. We can consider also formant locations (frequencies of the first three resonances), voice-unvoiced classification (periodic or aperiodic excitation), and ratios of high and low frequency energy.

Many proposed features are binary, others are continuous. The feature-detection stage usually consists of a set of detectors that operate in parallel and use appropriate processing and logic to make the decision as to the presence or absence, or value, of a feature. The algorithms used for individual feature detectors are sometimes sophisticated ones that do a lot of signal processing, and other times they are rather trivial estimation procedures.

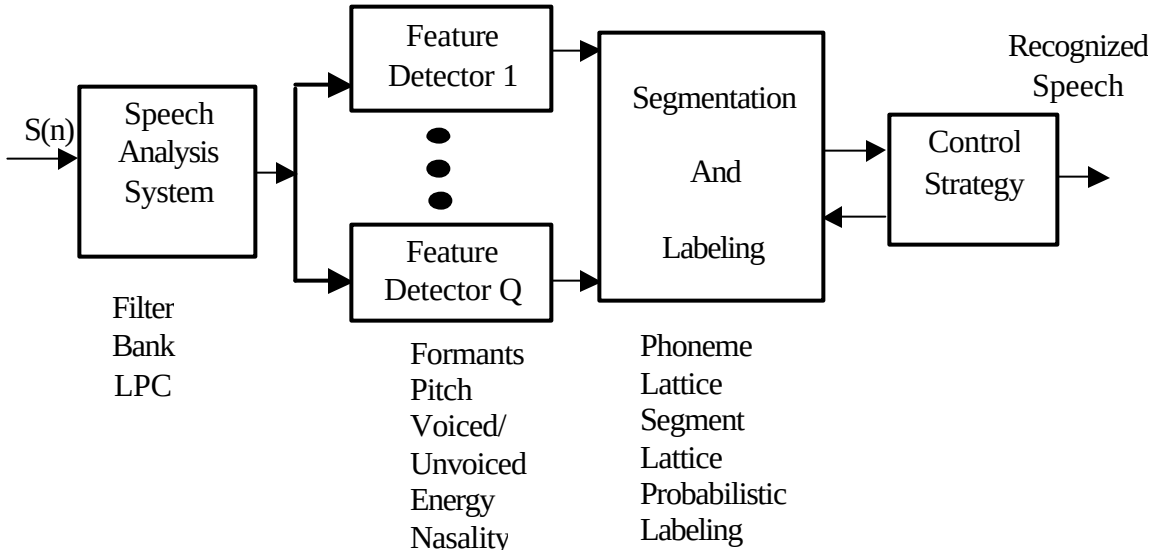

Figure 1.1. Block diagram of acoustic-phonetic speech-recognition system

1.3.1.3 Segmentation and Labeling (Classification) Phase

The system tries to find stable regions (where the features change very little over the region) and then to label the segmented region according to how well the features within that region match those of individual phonetic units. This stage is the essence of the acoustic-phonetic recognizer and is the hardest one to manage reliably; thus various control strategies are used to limit the extent of segmentation points and label possibilities. For example, for single word recognition, the assumption that a word contains at least two phonetic units, and no more than six phonetic units, means that the control strategy need only consider solutions with between 1 and 5 internal segmentation points. The labeling strategy can exploit lexical constraints on words to contemplate only words with n phonetic units whenever the segmentation gives $(n - 1)$ segmentation points. These constraints are often powerful ones that lessen the search space and remarkably increase accomplishment (accuracy of segmentation and labeling) of the system.

The outcome of the segmentation and labeling step is usually a phoneme lattice from which a lexical access procedure concludes the best matching word or sequence of words. By integrating vocabulary and syntax constraints into a control strategy, other types of lattices (e.g., syllable, word) can also be utilized. The quality of the matching of the features, within a segment, to phonetic units can be used to designate probabilities for the labels, which then can be used in a probabilistic lexical access procedure. The ultimate output of the recognizer is the word, or word sequence, that best matches, in some well-defined sense, the sequence of phonetic units in the phonetic lattice.

To recognize the labeling procedure on a segment classified as a vowel, we assume; using Figure 1.2, that we detect three features over the segment as first forment, F1, and second formant, F2, and duration of the segment, D. Several tests can be made to separate groups of vowels; they end up with the following:

- According to F1, compact vowels(/ɛ/,/a/,/Λ/,/⊃/).
- According to F2, acute vowels with high F2, and grave vowels according low F2.
- According to duration, tense vowels with large D and lax vowels with small values.
- Flat vowels where F1+F2 exceeds a threshold T and plain vowels for which (F1+F2) falls below the threshold T.

Vowel classification is one of the components of a phonetic labeling procedure, that can be implemented by a binary tree to make a decision as to various broad sound classes. The first detection is a sound/silence split in which the speech features (primarily energy in this case) are compared to selected thresholds, and silence is split off if the test is negative for speech sounds. The second decision is a voiced/unvoiced decision (based on the presence of periodicity within the segment) in which unvoiced sounds are split apart from voiced sounds. A test for unvoiced stop consonants is made (seeing if a stop gap of silence preceded the segment), and this separates the unvoiced stops (/t/, /p/, /k/, /c/) from

the unvoiced fricatives (/f/, /o/, /s/, /s/). A high-frequency/low-frequency (energy) test separates voiced fricatives (/v/, /o/, /z/, /z/) from other voiced sounds. Voiced stops are separated out by checking to see whether the preceding sound is silence (or silence like). Finally, a vowel/sonorant spectral test (searching for spectral gaps) separates vowels from sonorants (nasal consonants and /w/, /l/, /r/, and /y/). The vowel classifier of Figure 1.1 can then be used for finer vowel distinctions. But these tests are incomplete as there is no way of distinguishing diphthongs from vowels, practically. Every decision in this binary tree is subject to scrutiny as to its utility in any practical system.

1.3.1.4 Issues in the Acoustic Phonetic Approach

There are many problems associated with the acoustic-phonetic approach to speech recognition. These problems are related to the lack of success in practical speech-recognition systems. Some of them are the following:

- The method requires extensive knowledge of the acoustic properties of phonetic units. (Recall that the existence of phonetic units is assumed *a priori* in the acoustic phonetic approach, and knowledge of acoustic properties of these phonetic units often is established in an *a posterior* manner.) This knowledge is, at best incomplete, and at worst totally unavailable for all but the simplest of situations (e.g., steady vowels).
- The choice of features is based on very *ad hoc* considerations. For most systems the choice of features is based on premonition and is not optimal in a well-defined and significant sense.
- The design of sound classifiers is also not optimal. *Ad hoc* methods are generally used to construct binary decision trees. Lately, Classification And Regression Tree (CART) methods have been used to make the decision trees more robust [8]. Still, since the choice of features is most likely to be sub-optimal, optimal implementation of CART is barely achieved.

There is no well defined, automatic procedure for tuning the method (i.e., adjusting decision thresholds, etc.) on real, labeled speech. In fact, there is not even an ideal way of labeling the training speech in a manner that is both consistent and agreed on uniformly by a wide class of linguistic experts. Because of all these problems, the acoustic-phonetic method of speech recognition is still considered as a stimulating idea, but needs much more investigation and understanding before it can be used successfully in actual speech-recognition problems.

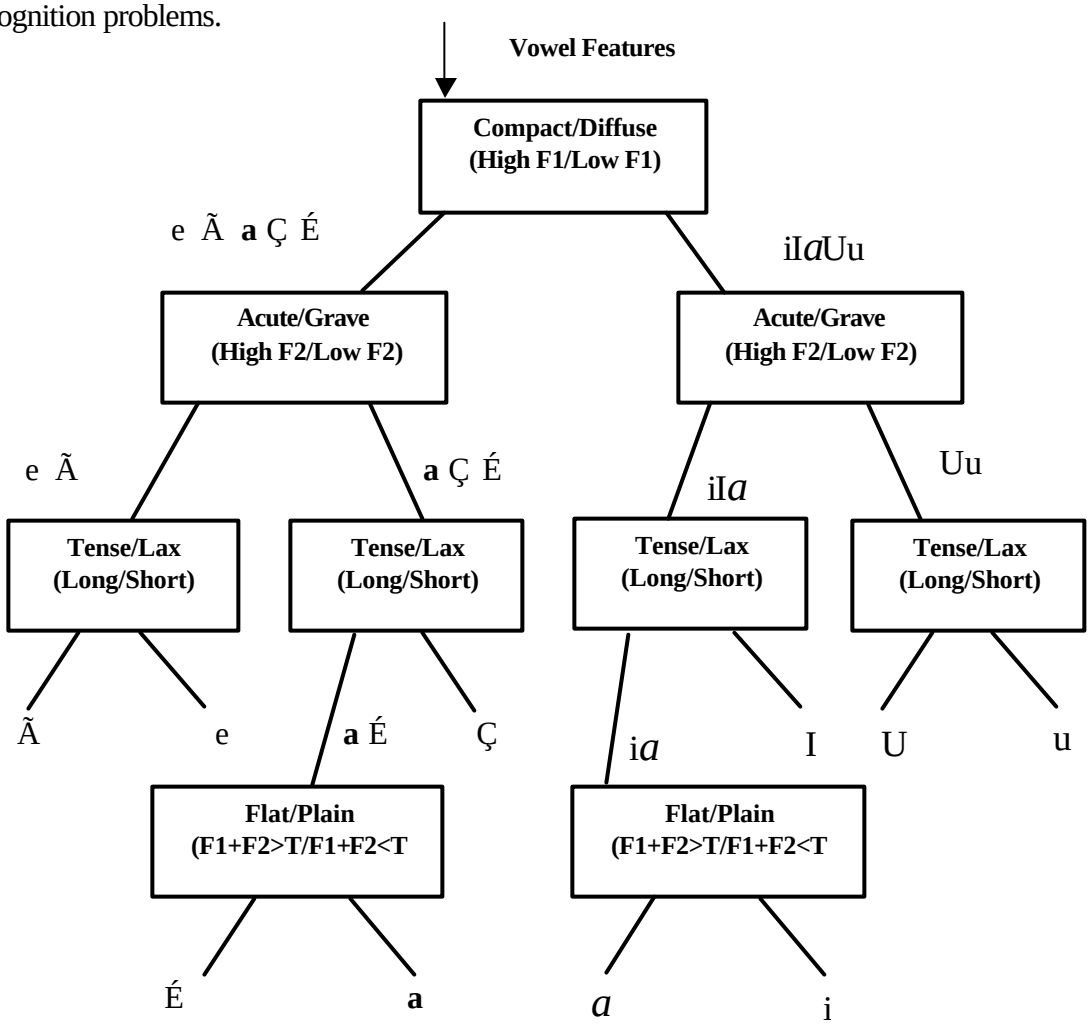

Figure 1.2. Acoustic-phonetic vowel classifier

1.3.2 The Statistical Pattern-Recognition Approach

This method is fundamentally one in which the speech patterns are used directly without explicit feature extraction (in the acoustic phonetic sense) and segmentation. As in most pattern-recognition approaches, the method consists of two steps: -

- Training of speech patterns, and
- Recognition of patterns via pattern comparison.

Speech "knowledge" is brought into the system through the training procedure. The concept is that if enough versions of a pattern to be recognised are presented in a training set, the training procedure should be able to satisfactorily characterize the acoustic properties of the pattern (with no regard for, or knowledge of, any other pattern presented to the training procedure). This type of characterization of speech via training is called pattern classification because the machine learns which acoustic properties of the speech class are reliable and repeatable across all training tokens of the pattern. The use of the method is the pattern-comparison stage, which does a direct comparison of the unknown speech (the speech to be recognised) with each possible pattern learned in the training phase and classifies the unknown speech according to the goodness of match of the patterns.

This approach has gained its importance for three reasons:

- This method is easy to use and to understand, it is rich in mathematical and communication theory justification for individual procedures used in training and decoding.
- It is robust and/or invariant to different speech vocabularies, users, feature sets, pattern comparison algorithms and decision rules. These properties make the algorithm appropriate for a wide range of speech units (ranging from phoneme-like units all the way through words, phrases, and sentences), word

vocabularies, speaker populations, background environments, transmission conditions, etc.

- This approach consistently provides high performance on any task that is reasonable for the technology and provides a clear path for extending the technology in a wide range of directions such that the performance degrades gracefully as the problem becomes more and more difficult.

In Figure 1.3 a block diagram of a generic pattern-recognition approach to speech recognition is shown. There are four main steps to the pattern-recognition paradigm to be discussed:

- *Feature measurement*, in which a sequence of measurements is made on the input signal to define the "test pattern". For speech signals, the feature measurements are usually the output of some type of spectral analysis technique, such as a filter bank analyser, a linear predictive coding analysis, or a Discrete Fourier Transform (DFT) analysis.
- *Pattern training*, in which one or more test patterns corresponding to speech sounds of the same class are used to create an ensemble pattern representative of the features of that class. The resulting pattern, generally called a reference pattern, can be an exemplar or template, derived from some type of averaging technique, or it can be a model that characterizes the statistics of the features of the reference pattern.
- *Pattern classification*, in which the unknown test pattern is compared with each (sound) class reference pattern and each reference pattern; and, a measure of similarity (distance) between the test pattern and each reference pattern is computed. To compare speech patterns (that consist of a sequence of spectral vectors), we require both a local distance measure, in which local distance is defined as the spectral "distance" between two well-defined spectral vectors, and a global time alignment procedure (often called a

dynamic time warping), which compensates for different rates of speaking (time scales) of the two patterns.

- *Decision logic*, in which the reference pattern similarity scores are used to decide which reference pattern (or possibly which sequence of reference patterns) best matches the unknown test pattern.

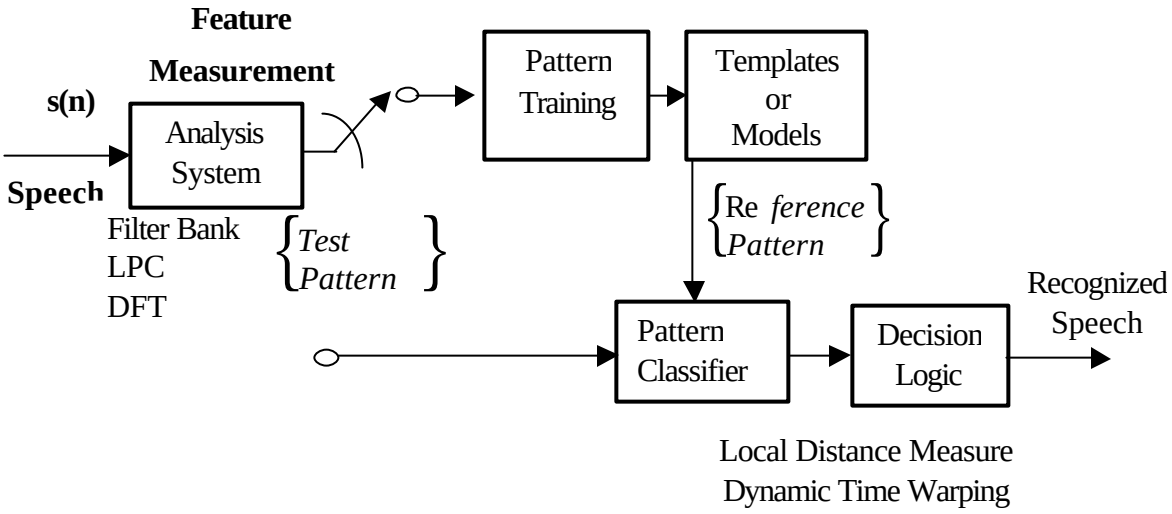

Figure 1.3. Block diagram of pattern-recognition speech recognizer

The factors that distinguish different pattern-recognition approaches are the types of feature measurement, the choice of templates or models for reference patterns, and the method used to create reference patterns and classify unknown test patterns.

The strong and weak points of the pattern recognition model include the following:

- Sensitivity of the performance of the system to the amount of training data available for creating sound class reference patterns; generally the more training, the higher the performance of the system for virtually any task.

- Because the speech spectral characteristics are affected by transmission and background noise, the reference patterns are sensitive to the speaking environment and transmission characteristics of the medium used to create the speech.
- No speech-specific knowledge is used in the system; consequently the method is relatively insensitive to choice of vocabulary words, task, syntax, and task semantics.
- The computational load for both pattern training and pattern classification is, generally, linearly proportional to the number of patterns being trained or recognised; hence, computation for a large number of sound classes could and often does become prohibitive.
- Because the system is insensitive to sound class, the basic techniques are applicable to a wide range of speech sounds, including phrases, whole words, and sub word units.
- It is relatively straightforward to incorporate syntactic (and even semantic) constraints directly into the pattern-recognition structure, thereby improving recognition accuracy and reducing computation.

1.3.3 The Artificial Intelligence Approach

This technique is a combination of the acoustic-phonetic approach and the pattern-recognition approach in that it exploits ideas and concepts of both methods. This approach attempts to mechanize the recognition procedure according to the way a person applies his/her intelligence in visualizing, analyzing, and finally making a decision on the measured acoustic features. Among the techniques used in this class of methods is the use of an expert system for segmentation and labeling, so that the difficult steps can be performed with more than just the acoustic information used by pure acoustic-phonetic methods.

The basic idea of the artificial intelligence approach to speech recognition is to compile and incorporate knowledge from a variety of knowledge sources and to bring it to bear on the problem at hand. Thus, for example, the AI approach to segmentation and labeling would be to augment the generally used acoustic knowledge with phonetic knowledge, lexical knowledge, syntactic knowledge, semantic knowledge, and even pragmatic knowledge. There are different knowledge sources to be defined:

- *Acoustic knowledge*–evidence of which sounds (predefined phonetic units) are spoken on the basis of spectral measurements and presence or absence of certain features.
- *Lexical knowledge*–the combination of acoustic evidence so as to postulate words as specified by a lexicon that maps sounds into words (or equivalently decomposes words into sounds).
- *Syntactic knowledge*–the combination of words that form grammatically correct strings (according to a language model) such as sentences or phrases.
- *Semantic knowledge*–understanding of the task domain so as to be able to validate sentences (or phrases) that are consistent with the task being performed, or which are consistent with previously decoded sentences.
- *Pragmatic knowledge*–inference ability necessary in resolving ambiguity of meaning based on ways in which words are generally used.

There are several ways to combine knowledge sources within a speech recognizer:

- The "bottom-up" processor (Figure 1.4), in which the lowest-level processes (e.g., feature detection, phonetic decoding) precede higher-level processes (lexical decoding, language model) in a sequential manner so as to constrain each stage of the processing as little as possible.
- The so-called "top-down" processor is considered as an alternative, in which the language model generates word hypotheses that are matched against the speech signal, and syntactically and semantically meaningful

sentences are built up on the basis of the word match scores. Figure 1.5 shows a system that is often implemented in the top-down mode by combining the unit matching, lexical decoding, and syntactic analyses modules into a consistent framework.

- The so-called blackboard approach, as illustrated in Figure 1.6. In this approach, all Knowledge Sources (KS) are considered independent; a hypothesis-and-test paradigm serves as the basic medium of communication among KSs; each KS is data driven, based on the occurrence of patterns on the blackboard that match the templates specified by the KS. The system activity operates synchronously, containing assigned cost and utility considerations and an overall ratings policy to combine and propagate rating across all levels. The blackboard approach was extensively studied at CMU in the 1970s [9].

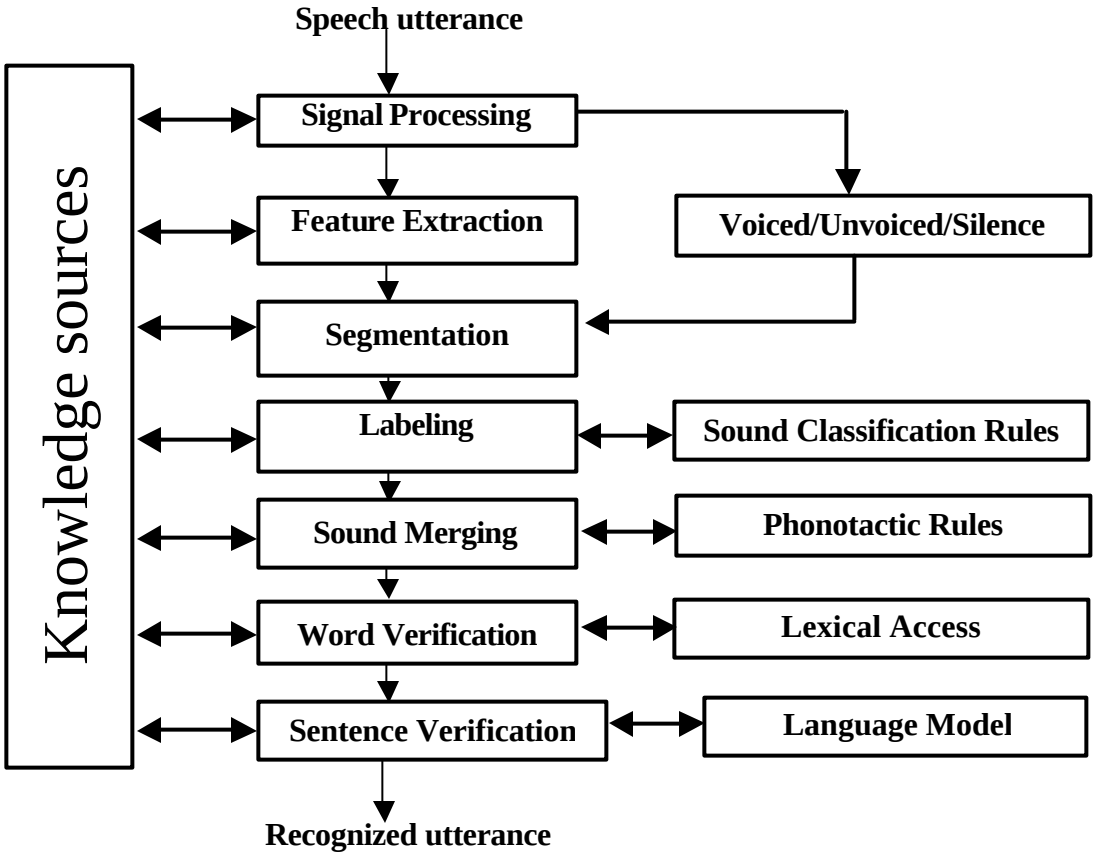

Figure 1.4. A bottom-up approach to knowledge integration for speech recognition

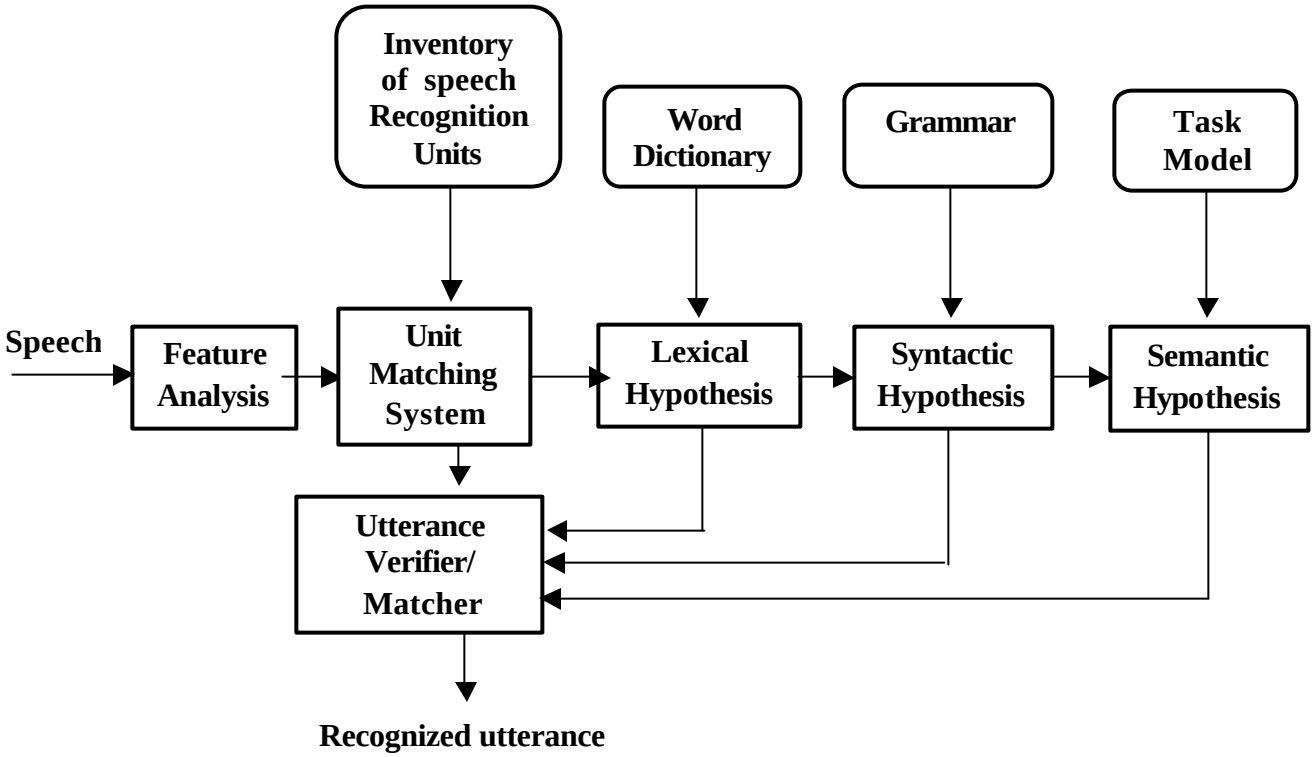

Figure 1.5. A top-down approach to knowledge integration for speech recognition

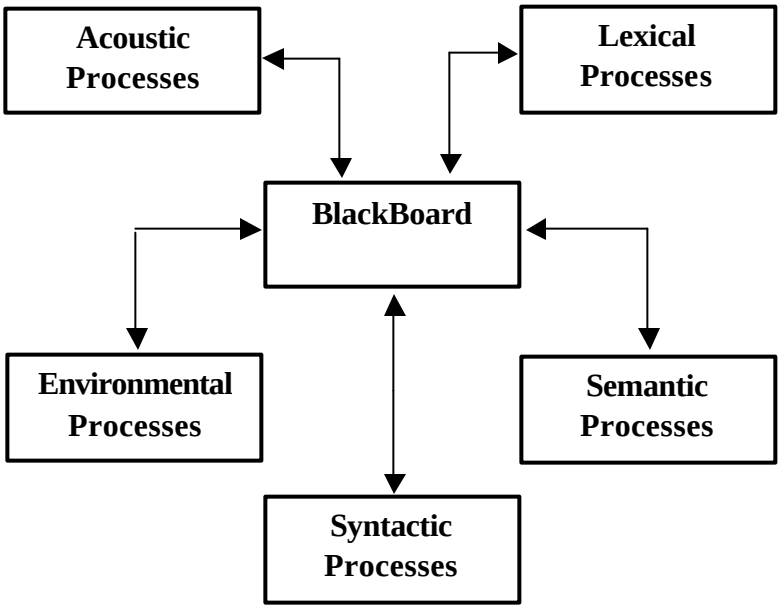

Figure 1.6. A blackboard approach to knowledge integration for speech recognition (after Lesser et al. [9])

1.3.4 Neural Networks and Their Application to Speech Recognition

Neural networks can be used for learning the relationships between phonetic events and all known inputs (including acoustic, lexical, syntactic, semantic, etc.) as well as for discrimination between similar sound classes. The use of neural networks could represent an individual structural approach to speech recognition, or be regarded as an implementation architecture that may be incorporated in any of the above three classical approaches. The concepts and ideas of applying neural networks to speech-recognition problems are relatively new.

A variety of knowledge sources need to be established in the AI approach to speech recognition. Therefore, two key concepts of artificial intelligence are automatic knowledge acquisition (learning) and adaptation. One way in which these concepts have been implemented is via the artificial neural network approach. We will discuss the motivation for why people have studied artificial neural networks and how they have been applied to speech-recognition systems.

Figure 1.7 shows a conceptual block diagram of a speech understanding system loosely based on a model of speech perception in human beings. The acoustic input signal is analyzed by an "ear model" that provides spectral information (over time) about the signal and stores it in a sensory information store. Other sensory information (e.g., from vision or touch) is available in the sensory information store and is used to provide several "feature-level" descriptions of the speech. Both long-term (static) and short-term (dynamic) memory are available to the various feature detectors. Finally, after several stages of refined feature detection, the final output of the system is an interpretation of the information in the acoustic input.

The system of Figure 1.7 is meant to model the human speech understanding system. The auditory analysis is based loosely on our understanding of the acoustic

1.4 Objectives

The objectives of this research are to investigate how unique individual speech features can be extracted [9], to minimize the computational processing by decreasing the required number of features, to overcome the limitations of multi-layer perceptrons by using modular neural networks, and ultimately to design a human/computer interface (HCI), i.e. an effective, and reliable real-time speech recognition system for isolated words. The system performance goal is to discriminate between the command words of the system. In summary, the following points form the basis of this research: -

- How to extract and select effective speech features.
- How to speed up the speech recognition system.
- Design and build an HCI system with high recognition accuracy.
- How to overcome the limitation of multi-layers perceptrons (MLPs) by using modular artificial neural networks (MANNs).
- Perform system performance assessment.

In this research, linear predictive coefficients (Lpc), partial correlation coefficients (ParCor), cepstral (Cep) coefficients, and cepstral weighting (CepW) coefficients are employed to represent the spectral features of speech. The linear predictive coding method models the present speech signal as a linear combination of the past values of the speech signal. This all-pole representation can also be used to model the human vocal tract.

A modular artificial neural network structure is introduced to provide decisions for the correct action. The design must be extensible, i.e. the system should be capable of learning to recognize new commands without having to retrain or redesign the whole system.

1.5 Contribution and Work Organization

The following contributions are results of this research: -

- A new effective, but simple, speech endpoint detection algorithm was formulated.
- A new compressed feature set to speed up the computational processing with high recognition rate was used.
- An effective design for a modular artificial neural network to solve the inherent limitation of multi-layer perceptrons was achieved.
- A human/computer interface (HCI) system with high recognition rate was successfully built with 92.7% accuracy rate.

The research work is presented in five chapters. Terminologies of speech recognition systems, application of speech recognition, approaches to automatic speech recognition by machine, and objectives are presented in Chapter 1. Chapter 2 describes speech preprocessing, the speech detection problem, endpoint detection approach, and speech endpoint procedure. Chapter 3 provides some important speech feature extraction techniques. Comparisons are made between them and a new median linear predictive coding technique. Experiments are made with the new features using dynamic programming and multi-layer perceptrons. Chapter 4 describes the following: multi-layer perceptrons, modular artificial neural networks, and how modular artificial neural networks are more capable than multi-layer perceptrons. Experimental work is also presented there. The structure of the system is a novel approach called Pyramidal Modular Neural Network (PMNN). The human computer interface system design is presented in Chapter 5. In chapter 6 are the results and conclusions.

References

[1]. Wilson J., *"Applications Of Speech Recognition In Industrial And Military Environments,"* 'Voice Processing - The New Revolution', Proceedings of the International Conference, London, July 1986.

[2]. Hollingum J. and Cassford G., *"Speech Technology At Work,"* IFS Publications, UK, 1988.

[3]. Bristow G. (Ed.), *"Electronic Speech Recognition*: *Techniques," Technology and Application*, Collins, 1986.

[4]. Waterworth J. A. and Talbots M., *"Speech And Language- Based Interaction With Machines,"* Ellis Horwood, 1987.

[5]. Conway J. J., *"The Development Of A Speech Recognition System For The PUMA 560 Industrial Robot,"* MSc Thesis, Cranfield Institute Of Technology, 1986.

[6]. Kafantaris A. E., *"CAD Application Software For A Speech Workstation,"* MSc Thesis, Cranfield Institute of Technology, 1986.

[7]. B. S. Atal and S. L. Hanauer, *"Speech Analysis and Synthesis by Linear Prediction of the Speech Wave,"* J. Acoust. Soc. Am., 50 (2): 637-655, August 1971.

[8]. L. Breiman,J.H. Freedman, R.A. Olshen, and C.J. Stone, *"Classification and Regression Trees,"* Wadsworth & Brooks, Monterey, CA, 1984.

[9]. V. R. Lesser, R. D. Fennell, L. D. Erman, and D. R. Reddy, *"Organization of the Hearsay-II Speech Understanding System," IEEE Trans. Acoustic, Speech, Signal Proc.*, ASSP-23(1): 11-23, 1975.

Chapter 2

Speech Preprocessing

2.1 Introduction

Detecting the presence of speech in a background noise is an important initial task to achieve high accuracy in speaker recognition system. Several algorithms have been implemented for the detection of endpoints of a speech signal [1-8]. Simple endpoint detection uses the short-time energy and zero crossing rate algorithms. Sophisticated endpoint detection uses the short-time energy, pattern comparison, adaptive level quantization, and a decision rule for noisy environments. Short time energy is used to distinguish among voiced and background silence speech. The zero crossing rate (ZCR) provides a rough voiced / unvoiced classification feature since unvoiced speech has a higher ZCR than voiced speech. In a voiced HCI system, speech endpoints are used to remove intervals of silence, which helps to reduce the amount of processing of the speech data.

2.2 Speech Detection Problem

Accurate detection of speech is considered a fairly simple problem concerning speech production in the most benign circumstances, that is, cautiously articulated and spoken in a relatively noise-free environment. In practice, however, there is difficulty in detecting endpoints caused by more than one problem. One class of these problems is connected to the speaker and his manner of producing the speech, e.g., heavy breathing and mouth clicks and pops.

The environmental conditions in which the speech is produced is considered the second difficulty in detecting speech endpoints reliably. The ideal environment for talking is a quiet room with no acoustic noise or signal generators other than produced by the speaker. Such an ideal environment is not always practical; so, one must consider speech produced in noisy background (as with fans or machinery running), in nonstationary environment (as in the presence of door slams, irregular road noise, car horns), with speech interference (as from TV, radio or background conversions), and in hostile circumstances (when the speaker is stressed, such as when navigating an aeroplane or while moving at high speeds). Some of these interfering signals possess as much speech like quality as the desired speech signal itself, making accurate endpoint detection quite difficult.

Finally there remains one source of signal degradation, the distortion introduced by the transmission system over which the speech signal is sent. Factors like crosstalk, intermodulation distortion, and different types of tonal obstruction occur to various degrees in a communications channel, again adding to the constitutional difficulties in reliably detecting speech endpoints [9].

Accordingly, it is not simple to separate speech from background noise except in high signal-to-noise ratio conditions, where the energy of the lowest-level speech (e.g., weak fricatives, low level voice portions...etc.) exceeds the energy of the background noise, and a simple energy-based measure is enough to acquire the essential objective. To ease the task some precautions have to be taken during recording as follows: -

- High-pass filtering of speech above 100 Hz to eliminate the 50 or 60 in Hz hum originated in subsystem powering.
- Low-pass filtering at about 4 kHz to allow sampling at 10 kHz for further digital processing of speech signal.

The following examples are given to clarify the problems associated with determination of the beginning and the end of speech block. Before automating

the endpoint detection procedure, we visually inspect the signals to locate the end points by observing significant changes in the signal pattern as depicted in Figure 2.4 and 2.5. In these Figures, as well as in the following ones, each point on the horizontal axis is of 10ms duration.

In Figure 2.1 is shown the waveform of the word "CLOSE", where the detection of the beginning of the word is done easily as the energy changes significantly at the transition from silence to speech.

In Figure 2.2 the beginning of the word "TOOLS" is indicated easily as the frequency of the speech is totally different than the frequency of the background noise.

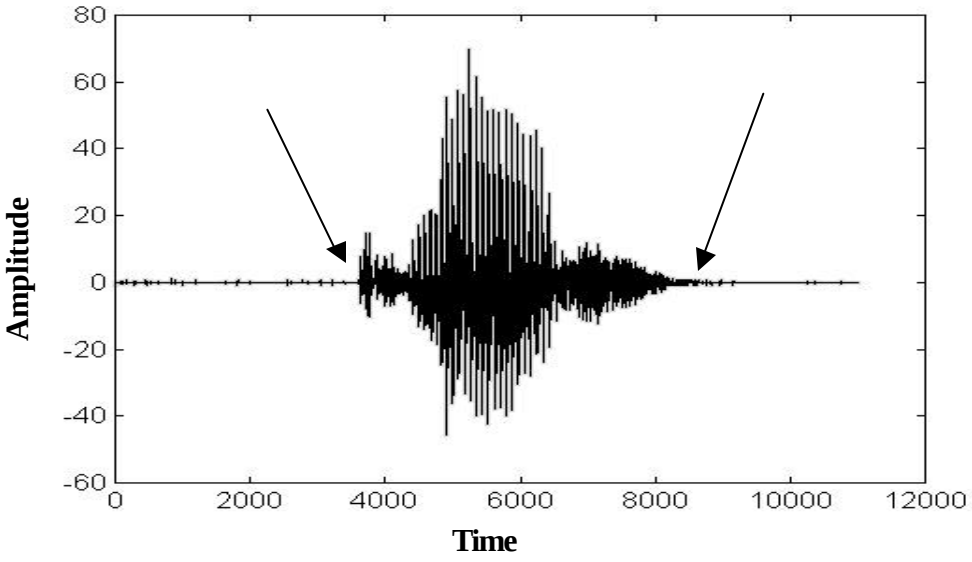

Figure 2.1. Waveform of word "CLOSE"

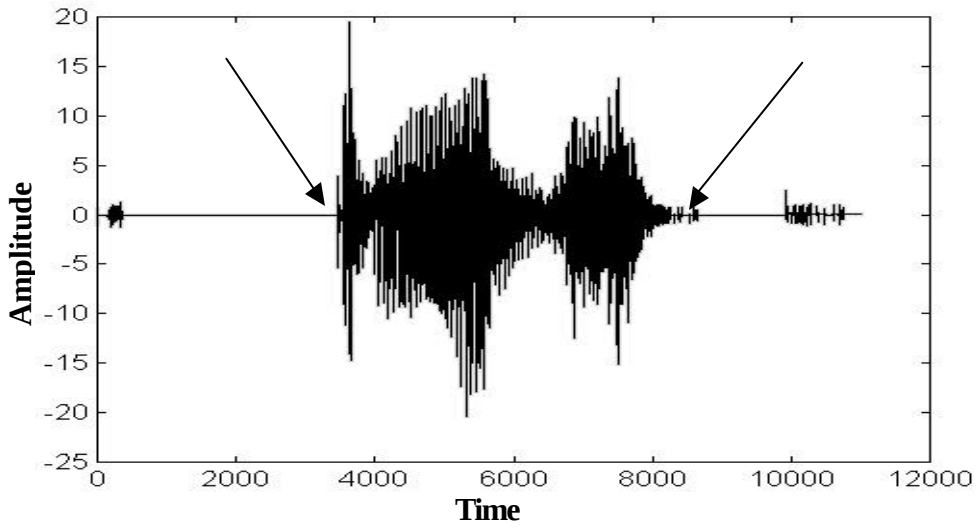

Figure 2.2. Waveform of word "TOOLS"

Alternatively, a high rate of zero crossings of the waveform is discernible.

Figures 2.3 to 4.5 illustrate some of the detection problems associated with the following aspects:

- The presence of weak fricatives (F, TH, H) at the beginning or end of a word. (see Figure 2.3).
- The presence of letters that become devoiced by speaker at the end of speech. (see Figure 2.4).
- The presence of a final nasal. (see Figure 2.5).

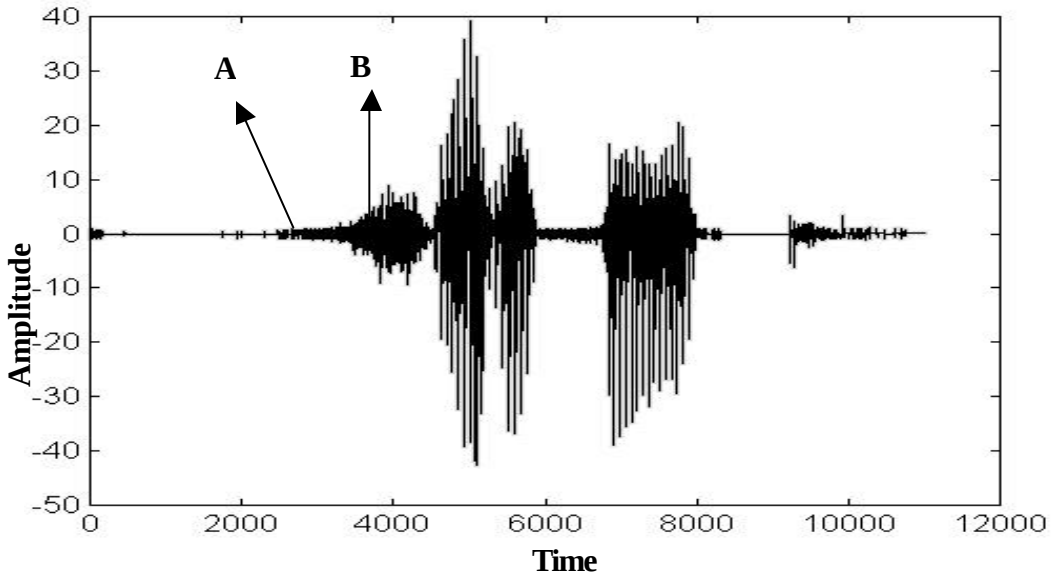

Figure 2.3. Waveform of word "FORMAT"

Figure 2.3 shows the waveform of the word "Format". If we select the point "B" as a beginning of the word, as the eye may be deceived because that word begins with a weak fricative, the correct position where the word starts is at the point "A".

Figure 2.4 shows the waveform of word "FIVE". Here the eye might select the point "A" as the endpoint of the word, while the real endpoint is approximately at the point "B".

The new endpoint detection scheme can produce the correct labeling in all these cases.

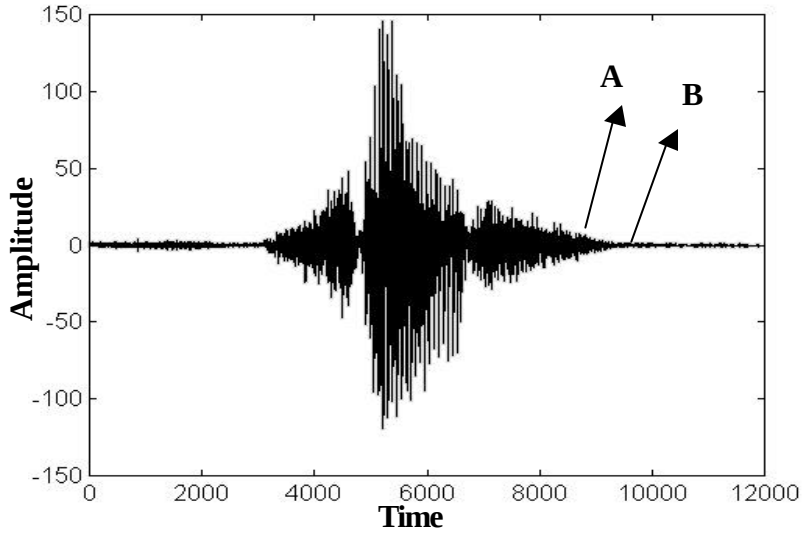

Figure 2.4. Waveform of word "FIVE"

This incorrect detection was due to the change in letter "V", being converted to "F", and e is devoiced. Thus, as the letter "F" is a weak fricative, the amplitude is clearly weak. Hence, if the word begins or ends with a weak fricative, there is a great difficulty to locate the endpoints for the speech word.

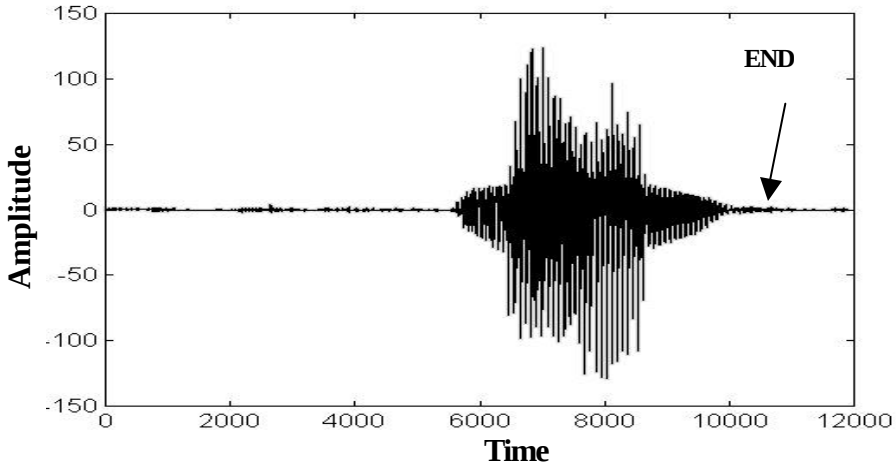

Figure 2.5. Waveform of word "NINE"

Figure 2.5 is the waveform of the word “NINE”, where it is quite difficult to locate where the final nasal ends and where the noise background begins. The most accurate position is that marked with "END".

It is important to point out here that for accurate recognition of the word, it is not necessary to locate exactly the beginning and the end of the word; but, instead, it is necessary to include the features of the significant isolated events within the word [5]. For example the word “TOOLS”, it is not important if the trailing unvoiced energy is omitted; but, for a word like “FOUR” it is very important to accurately locate and include the weak start fricative “F” as a significant event in isolation. Yet, experience has proved that it is not necessary to include the whole initial unvoiced energy, only from 30 to 50 ms of unvoiced energy is enough for most word recognition purposes. This will be discussed in detail while handling the endpoint detection approaches.

2.3 Endpoint Detection Approaches

According to the pattern matching paradigm and its interaction with each approach, we can classify endpoint detection into main three approaches: -

- The explicit approach
- The implicit approach
- The hybrid approach.

* The Explicit Approach

This approach is built on the premise that speech detection can be accomplished independently of other pattern-matching operations in the later stages of the speech-recognition process. Figure 2.6 shows a block diagram of the explicit approach to endpoint detection.

The speech signal is first processed followed by feature measurements. So, the location and definition of the speech events are considered the main results of the speech detection method. Detected speech is sent to the pattern-comparison algorithm; and, finally, the decision mechanisms chooses the recognised word. Although the speech detection module is likely to use the same measured features as those used for recognition, it is independent of the recognition processing and can use a different set of features, if appropriate. For signals with a stationary, low-level noise, acoustic background, the approach produces reasonably good detection accuracy. A noisy environment, or a nonstationary interference may lead to the failure of the approach. With an explicit model of the background signal included in the reference templates, a signal in the form of "background-speech-background" can still be compared and classified. Figure 2.7 shows a block diagram of the implicit approach to speech detection.

The unmarked signal sequence is processed by pattern matching in which all (or a large set of all) possible endpoint sets are considered; and, the decision mechanism provides an ordered list of the candidate words, as well as the corresponding speech locations. The final result is considered the best estimate and its associated endpoints. The implicit approach, discussed next, suffers from a heavy computational load, but offers a potentially higher detection accuracy than the explicit approach.

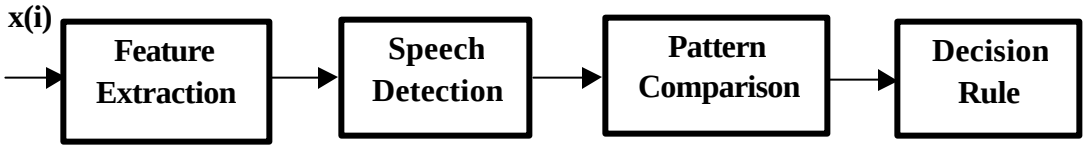

Figure 2.6. Block diagram of the explicit approach to speech endpoint detection

* The Implicit Approach

From Figure 2.8, we can see an example of a measured log-energy contour of an utterance and two possible sets of word boundary locations. The example of

Figure 2.8a corresponds to the case in which the initial high-energy transient (labeled "A" in the Figure) is classified as part of the background; the word boundaries are as shown.

(This case corresponds to the word without an initial stop constant, such as the word "even"). The example of Figure 2.8b corresponds to the case in which the initial high- energy transient is treated as a stop consonant and classified as part of the word (e.g., the initial part of a word like "acquire"").

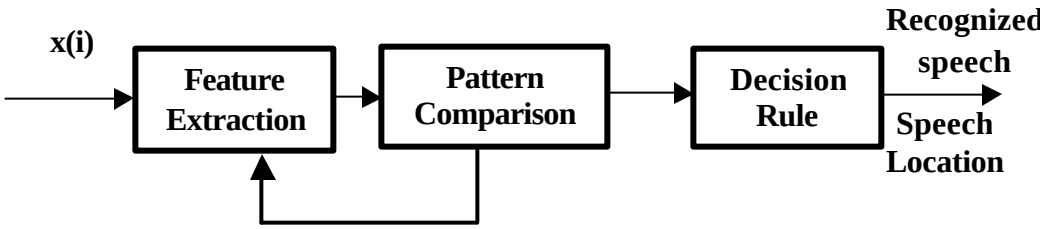

Figure 2.7. Block diagram of the implicit approach to speech endpoint detection

Therefore, depending on the word to be recognised, the boundary locations could innately be different with the implicit method, whereas with the explicit method boundary locations are made by a single choice.

By considering a small reasonable set of estimates of the endpoint set, the computational complexity of the implicit approach can be significantly reduced. This leads us to the next method, the hybrid approach.

* The Hybrid Approach

This approach gets from the explicit method its way of obtaining several potential endpoint sets for recognition processing, and from the implicit method how to choose among the alternatives. The most likely candidate word and the corresponding endpoints, as in the implicit approach, are provided by the decision box. Figure 2.9 shows a block diagram of the hybrid method.

The hybrid approach has a computational load equivalent to the explicit method, but with accuracy comparable to the implicit method.

These three approaches to endpoint detection differ in their degree of interaction with the recognition process. The basic speech-detection algorithm used to obtain an estimate of the endpoints often involves feed-forward processing of the measured short- time energy level. Figure 2.10 shows a block diagram of such a processing algorithm.

The adaptive level equalization module estimates the level of the acoustic background and uses the result to equalize the measured energy contour. Preliminary energy pulses, which are speech-like bursts of energy during the recording interval, are then detected from the equalized energy contour. Finally, these potential energy pulse endpoints are ordered, according to their likelihood, to determine the possible sets of word endpoint pairs. Extensive experimentation is necessary in deciding the best values of required set of thresholds and the ordering logic to provide the most reasonable set of endpoints.

2.4 Implementation of Endpoint Detection:

In the HCI system, speech endpoint detection is used to detect the presence of speech to remove the silences in the background. The endpoint algorithm would be based upon:

- Energy, and
- Zero crossing rate.

Moreover, a simple and reliable decision logic is required. It is to be noted that evaluation is done under the following conditions:

Low frquency cut-off of 100 Hz.,

Low frequency cut-off of 4000 Hz., and

48 db per octave skirts.

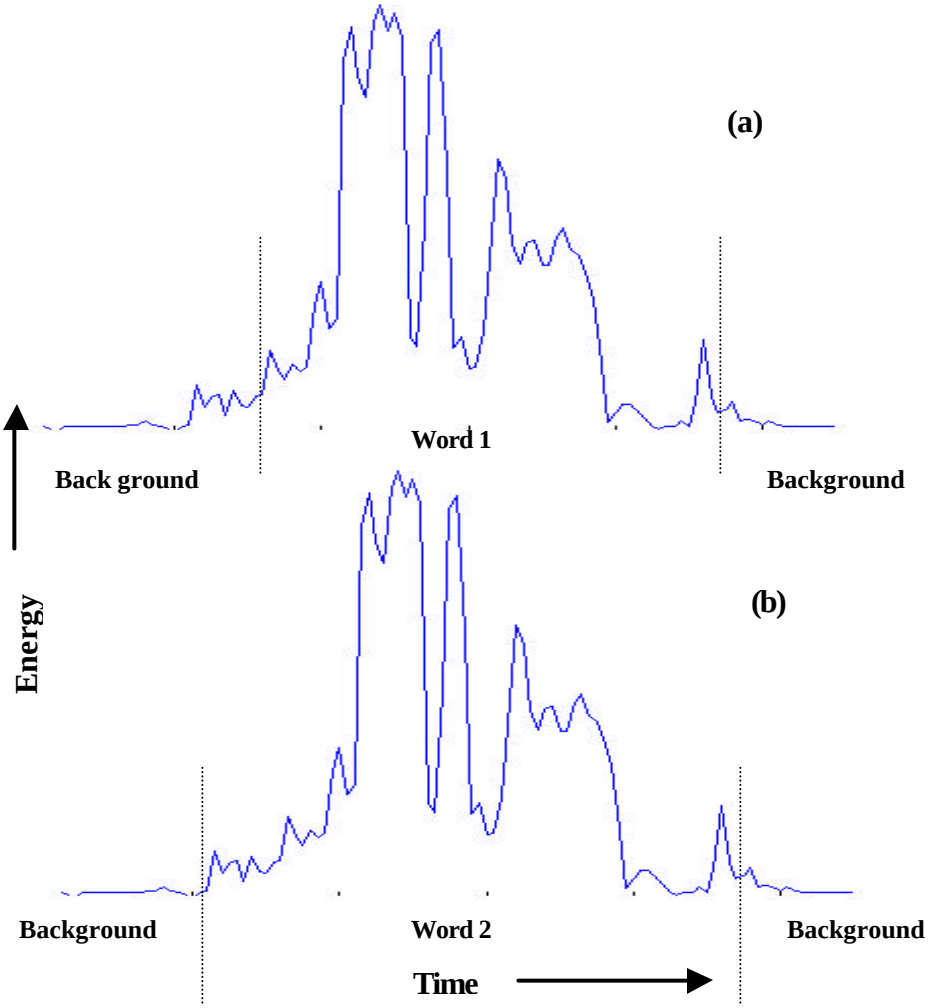

Figure 2.8. Examples of word boundaries as determined by implicit endpoint detection. (a) Wrong endpoint according to energy. (b) Correct endpoints.

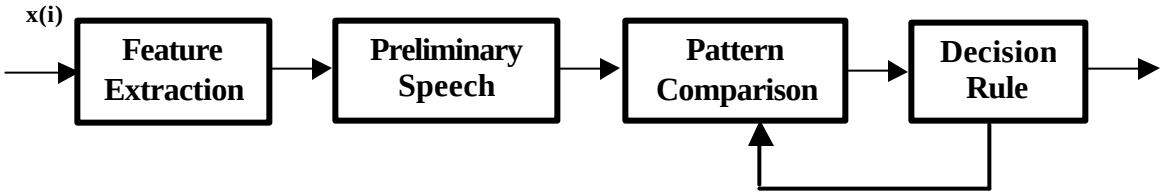

Figure 2.9. Block diagram of the hybrid approach to speech endpoint detection

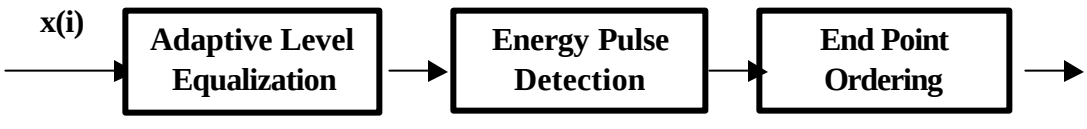

Figure 2.10. Block diagram of typical speech activity detection algorithm

2.4.1 Energy-Based Endpoint Algorithm: -

The "energy" $E(n)$ is defined as the sum of the magnitudes of 10 ms of speech centered on the measurement intervals, i.e. for a 10 kHz sampling rate, and 100 sample values (10 ms):

$$E(n) = \sum_{i=-50}^{50} s(n+i)^2 \tag{2.1}$$

where $s(n)$ is the speech sample value.

The use of the magnitude function (instead of the squared magnitude function) and the choice of a 10ms window for computing permits computing in integer arithmetic, and thus increases the speed of processing.

Moreover, using magnitudes "bounds" the large amplitude speech variations and makes the energy function much smoother (see Figure 2.11).

Figure 2.11 illustrates the energy function for the words "close" and "tools", calculated every 10ms (100 times / sec.).

The sequence of finding an end-point is as follows:

The energy function $E(n)$ is found; and both

The peak energy IMX and the silence energy IMN are found.

IMX and IMN are mainly used to set two thresholds, ITL & ITU as follows: -

$$I_1 = 0.03 * (IMX - IMN) + IMN \tag{2.2}$$

$$I_2 = 4 * IMN \tag{2.3}$$

$$ITL = min(I_1, I_2) \tag{2.4}$$

$$ITU = 5 * ITL \tag{2.5}$$

Figure 2.12 describes the algorithm for an energy-based first guess for the endpoint locations, where the search continues until the lower threshold is

exceeded. This point is labeled as the beginning of the speech, but in case the energy falls below ITL before it rises above ITU, a new beginning will be searched for where the energy exceeds ITL, and then exceeds ITU before falling again below ITL. At this stage a point is located, N_1 (see Figure 2.13 a). A similar algorithm is used to get another point N_2 signaling where speech stops.

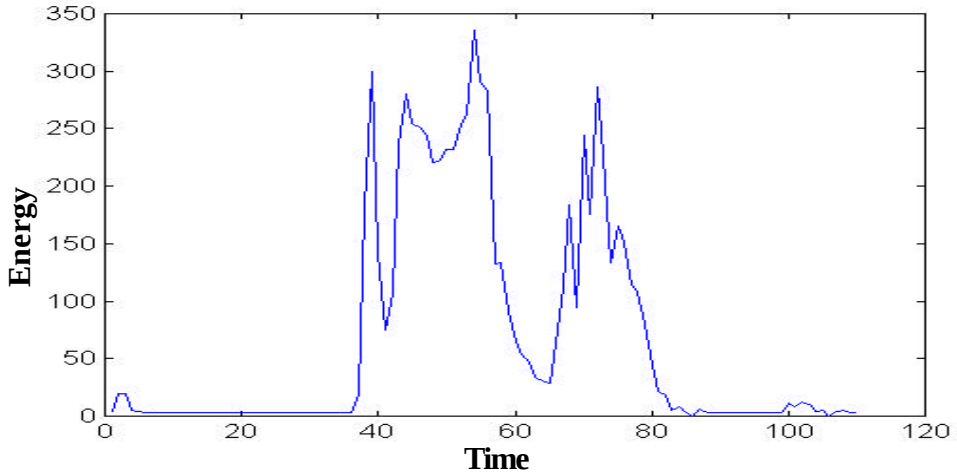

Figure 2.11-(a) Energy for the word "Close"

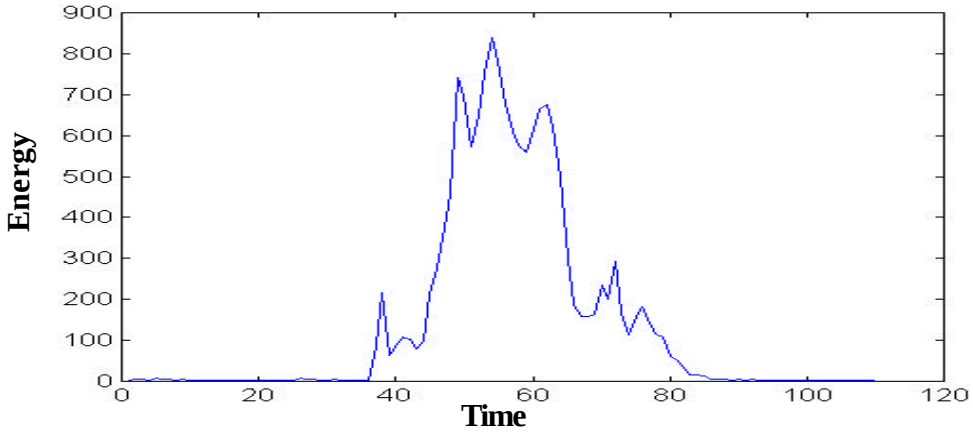

Figure 2.11-(b) Energy for the word "Tools"

```
Variables
        Inputs
          E           Energy Function.
          ITL         Lower threshold for speech energy.
          ITU         Upper threshold for speech energy.
        Outputs
          N1          Begining point.
        Internal
        i,m           Counters.
Line    Procedure BeginPoint
1       m⇐1
2       Repeat
3               if (E(m)<ITL) then
4                       m⇐m+1
5                       continue
6               EndIf
7               i⇐m
8               Repeat
9                       If (E(i)<ITU) then
10                              i⇐i+1
11                              continue
12                      EndIf
13                      N1 ⇐ i
14                      If i<>m then
15                              N1⇐ N1-1
16                      EndIf
17              Until   (E(i)>=ITU)
18      Until   (E(m)>=ITL)
19      end BeginPoint
```

Figure 2.12. Energy-Based Initial Estimate of Beginning Point

2.4.2 Zero-Crossing-Rate-Based modification of the Endpoint:

In spite of the fact that the zero crossing rate is highly vulnerable to 50 Hz hum, dc offset,...etc., it is still a good measure for the presence or the absence of unvoiced speech[10], and will be used here to modify the results of the energy-based endpoint detection algorithm. The zero crossing rate is defined here as the number of crossings of the zero level per 10 ms interval. It is assumed that the first 100 ms of the recording is silence and the statistics of this silence are estimated, mainly by the average energy and zero-crossing rate parameters. If any of the measurements are deemed excessive, the algorithm will halts with a warning. The zero crossing threshold is taken for unvoiced speech as IZCT. This is evaluated as follows:

$$IZCT = min(IF, \overline{IZC} + 2s_{IZC}) \quad (2.6)$$

With

IF being a fixed threshold of 25 crossing per 10 ms.,

$\overline{IZC}$ being the mean zero crossing rate during silence, and

s_{IZC} being the standard deviation of the zero crossing rate during silence.

Application of this zero crossing rate threshold on the calculated zero crossing rate is shown in Figure 2.13b. In this Figure, the energy level yields the first guess N_1 and N_2, while the zero-crossing rate is used for recalculation of point N_1 and updating to $\overline{N}_1$, while keeping N_2 as obtained from the energy-based critetion. Updating to $\overline{N}_1$ proceeds by examining the interval N_1 .. N_1 -25 i.e. 250 ms before the begining of the detected start point. The number of times that the zero crossing rate exceeds the threshold IZCT is found and checked against three. When it is greater than three, the start point is moved to where the threshold was exceeded, else the point N_1 is kept. Similarly, a procedure is used to find the endpoint location by finding any unvoiced energy in the interval N_2 ..

N_2+25, if the zero crossing test succeeds, the point is adjusted, otherwise the old point N_2 is kept (see Figure 2.14 for final algorithm).

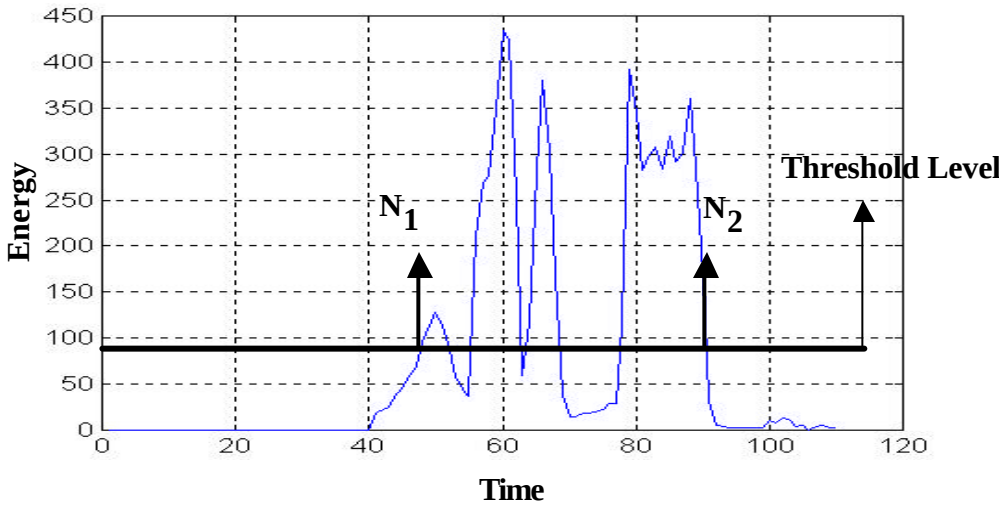

Figure 2.13-(a) Energy for word "Format"

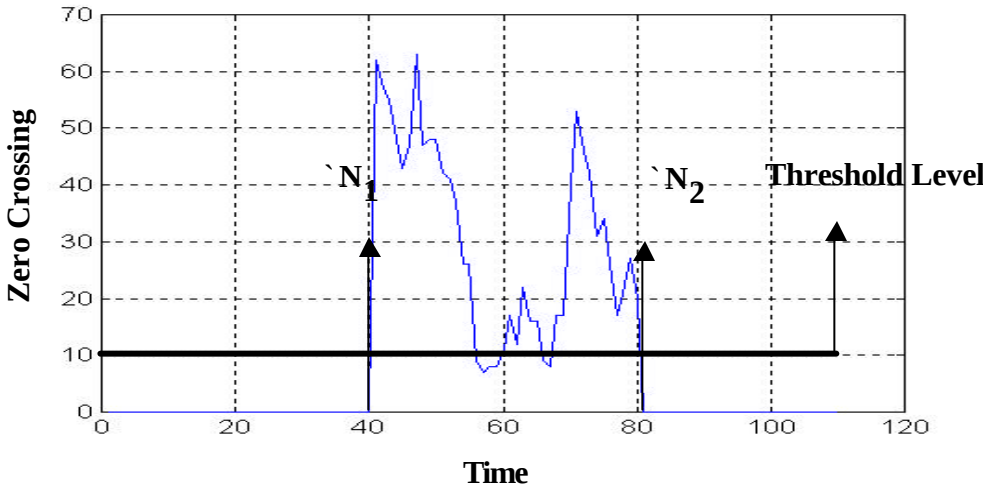

Figure 2.13-(b) Zero Crossing for word "Format"

Figure 2.13. EndPoint Localization
a - On the energy plot
b - On the zero crossing plot

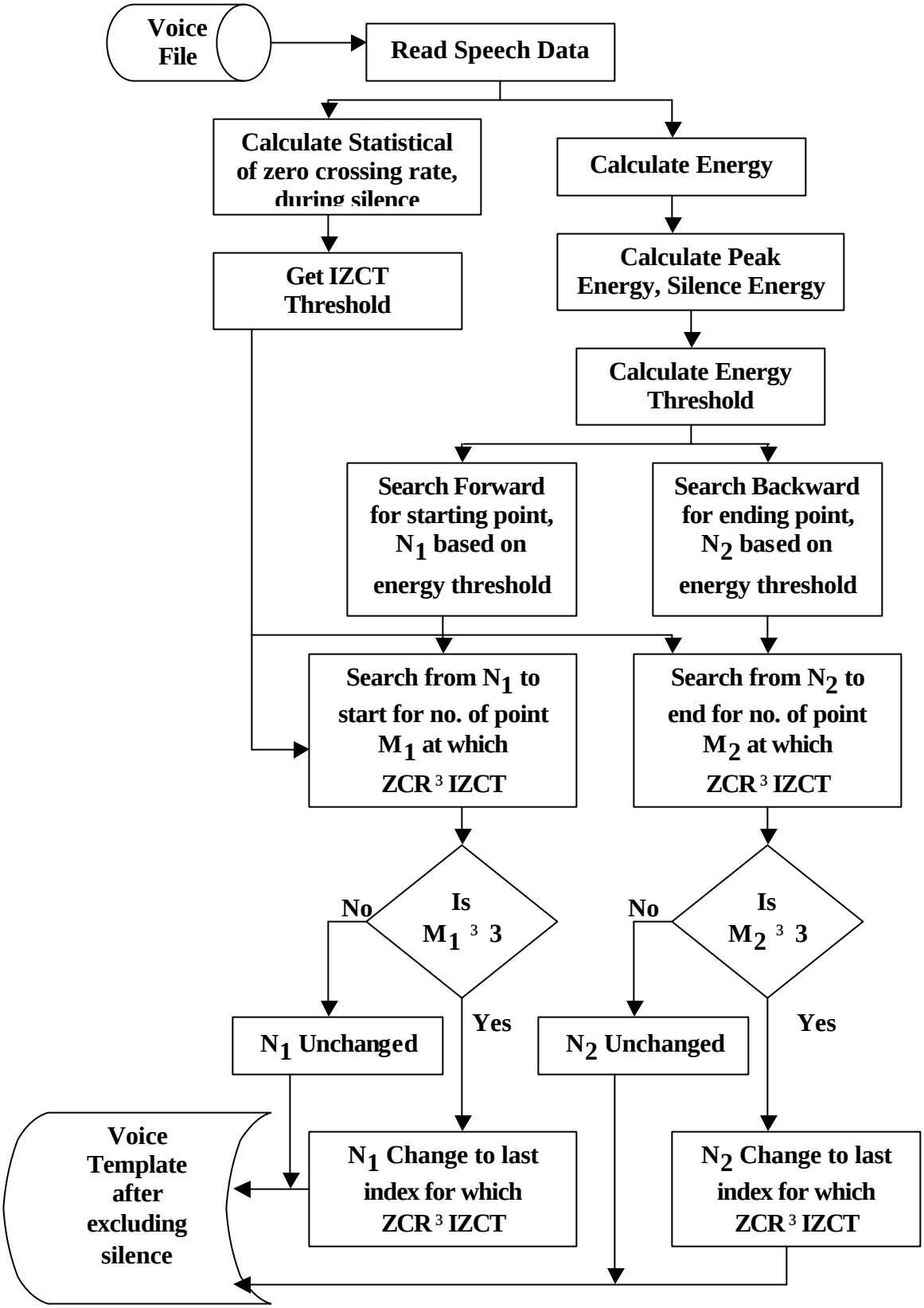

Figure 2.14. Final Algorithm for Ending Point

References

[1]. B.S. Atal and L.R. Rabiner, *"A Pattern Recognition Approach to Voiced-Unvoiced-Silence classification with Application to Speech Recognition,"* IEEE Trans. ASSP, pp. 201-212, June 1976.

[2]. E.S. Dermatas, N.D. Fakotakis, and G.K. Kokkinakis, *"Fast Endpoint Detection Algorithm for Isolated Word Recognition in office Environment,"* Proc. Intl. Conf. Acoust., Speech & Signal Processing, pp. 733-736, July 1991.

[3]. L.F. Lamel, L.R. Rabiner, A.E. Rosenberg, and J.G. Wilpon, *"An Improved Endpoint Detector for Isolated Word Recognition,"* IEEE ASSP, vol. 29, pp. 777-785, August 1981.

[4]. H. Ney, *"An Optimization Algorithm for determining the Endpoints of Isolated utterances,"* Proc. Intl. Conf. Acoust., Speech & Signal Processing, pp. 720-723, 1981.

[5]. L.R. Rabiner and M.R. Sambur, *"An Algorithm for Determining the Endpoints of Isolated Utterances,"* Bell Syst. Tech. J., vol. 54, no. 2, pp. 297-315, Feb. 1975.

[6]. L.R. Rabiner, C.E. Schmidt, and B.S. Atal, *"Evaluation of a Statistical Approach to Voiced-Unvoiced-Silence Analysis for Telephone-Quality Speech,"* Bell System Tech. J., pp. 455-487, Mar. 1977.

[7]. L.R. Rabiner and M.R. Sambur, *"Voiced-Unvoiced-Silence Detection Using the Itakura LPC Distance Measure,"* IEEE ASSP, 1977.

[8]. M.H. Savoji, *"A Robust Algorithm for Accurate Endpointing of Speech Signals,"* Speech Communication, vol. 8, pp. 45-60, 1989.

[9]. Bell Telephone Laboratories, *"Transmission System for Communications,"* Rev. 4th ed., New Jersey, 1970.

[10]. J.G. Wilpon, L.R. Rabiner, and T.B. Martin, "An Improved word-detection algorithm for telephone-quality speech incorporationg both syntactic and semantic constraints," At&T Tech. J., 63(3):479-498, March 1984.

Chapter 3

Feature Extraction Techniques

3.1 Introduction

Dautrich et al. (1983)[1] found that the performance of speech recognizers, based on Linear Predictive Coding (LPC) front ends, is comparable to or better than that of recognizers based on filter-bank front ends. This was explained in two ways:

- Filter banks inevitably represent a fixed quantization of the frequency range, while LPC is adaptive. We can say that if a significant energy concentration in some word falls on the boundary between two filters, with a slight variation it may land in one filter's band during training and in the adjacent filter's band during recognition. Thus, a small difference in a pole location can cause an exaggerated distance measure between two versions of the same word and can thereby lower recognition accuracy. On the other hand LPC, will place its poles where they belong in either case, and a small shift in the pole's location will not greatly increase a distance measure.
- Dautrich's results were obtained with telephone-bandwidth speech. (Their filter banks covered a range from 200 to 3200 Hz). Most commercial recognition systems, however, are not intended for use with telephone transmitted speech, and their filter banks typically cover a frequency range twice as wide. The additional information provided by filters in the range above 3200 Hz offsets any degradation due to the quantization of the frequency range.

In the previous section we compressed a huge amount of speech data by detecting the beginning and the end points of the speech. Although we had reduced the amount of the speech data needed to be stored (to the size of the speech data reduced from leading and trailing noise), we did not yet reduce the speech data to a reasonable size. This will

force us to find another terminology beside end point detection that can shrink the speech data size. In this section we shall give consideration to solving this problem by using LPC. Before describing a general LPC front-end processor for speech recognition, it is worthwhile to review why LPC has been so widely used.

These reasons include the following:

- LPC provides a good model of the speech signal. This is especially true for the quasi steady-state voiced regions of speech in which the all-pole model of LPC provides a good approximation to the vocal tract spectral envelope. During unvoiced and transient regions of speech, the LPC model is less effective than for voiced regions, but it still provides an acceptably useful model for speech-recognition purposes.
- The way in which LPC is applied to the analysis of speech signals leads to a reasonable source/vocal-tract separation. As a result, a parsimonious representation of the vocal tract characteristics (which we know are directly related to the speech sound being produced) becomes possible.
- LPC is an analytically tractable model. The method of LPC is mathematically precise and is simple and straightforward to implement in either software or hardware. The computation involved in LPC processing is considerably less than that required for an all-digital implementation of the bank-of-filters model.
- The LPC model works well in recognition applications.

In this section we are going to discuss:

- Linear predictive coding.
- Some linear function methods, and how they are solved.
- Median linear predictive coding and some experiment results.

3.2 Linear Predictive Coding

Linear predictive coding (LPC) attempts to model the present speech signal as a linear combination of past signal values, which translates into an all-pole transfer function representation of the speech production system, Figure 3.1. Therefore, the speech signal $s(n)$ can be represented by the difference equation below

$$s(n) = Gu(n) + \sum_{k=1}^{P} a_k s(n-k) \tag{3.1}$$

Where $u(n)$ is an unknown excitation and G is a gain of excitation.

The transfer function $H(z)$ associated with Eq. (3.1) is given by

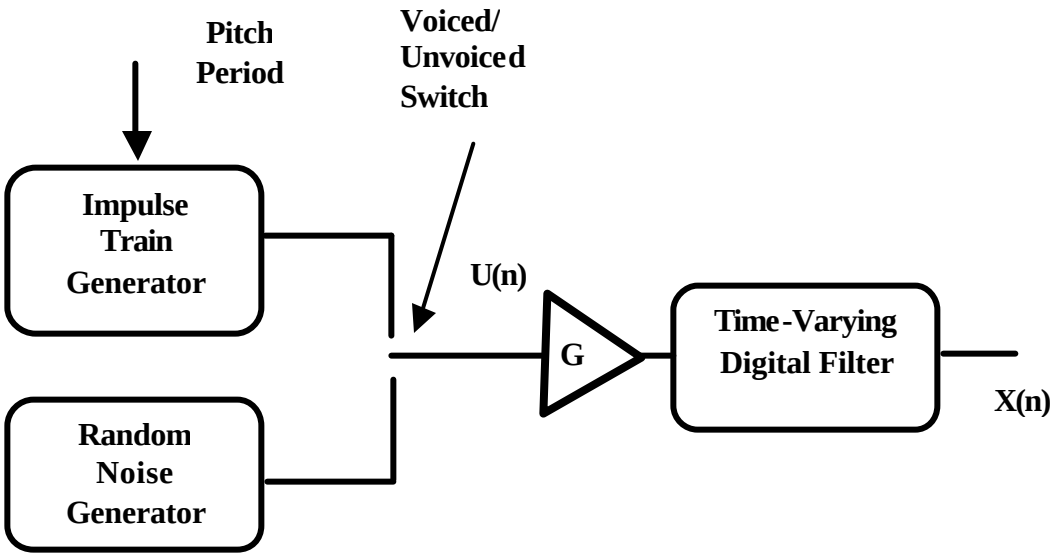

Figure 3.1. Speech Synthesis model based on LPC model

$$H(z) = \frac{S(z)}{U(z)} = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} = \frac{1}{A(z)} \tag{3.2}$$

The estimated signal $\hat{s}(n)$ and the prediction error $e(n)$ are defined as

$$\hat{s}(n) = \sum_{k=1}^{P} a_k s(n-k), \tag{3.3}$$

$$e(n) = s(n) - \hat{s}(n) = s(n) - \sum_{k=1}^{P} a_k s(n-k) \tag{3.4}$$

Then the total mean-squared error is given by

$$E = \sum_n e^2(n) = \sum_n \left[s(n) - \sum_{k=1}^{P} a_k s(n-k) \right]^2 \tag{3.5}$$

Using the method of least squares, the parameters a_k can be determined by setting

$$\frac{\partial E}{\partial a_i} = 0, \qquad i = 1,2,...,P \tag{3.6}$$

to get $$F_{io} = \sum_{k=1}^{P} a_k F_{ik}, \qquad i = 1,2,...,P \tag{3.7}$$

where $$F_{ik} = \sum_n s(n-i)s(n-k) \tag{3.8}$$

There are two principal methods to solve Eq. (3.7) for the a_k coefficients: (1) the autocorrelation method and (2) the covariance method.

3.2.1 The Autocorrelation Method

The autocorrelation method assumes that the signal exists inside a window of length N, and equals zero outside the window. Thus the speech sample for minimization can be expressed as

$$s(m) = \begin{cases} s(m+n)w(m), & 0 \le m \le N-1 \\ 0, & \text{otherwise} \end{cases} \tag{3.9}$$

where $w(m)$ is a suitably chosen window.

Finally, the problem is to minimize the mean squared error below

$$E_n = \sum_{m=0}^{N-1+P} e_n^2(m) \tag{3.10}$$

The autocorrelation solution to Eq. (3.8) can be expressed as

$$\sum_{k=1}^{P} a_k r(|i-k|) = R(i) \qquad i \le i \le P \tag{3.11}$$

$$G = \sqrt{R(0) - \sum_{k=1}^{P} a_k R(k)} \tag{3.12}$$

where $$R(i) = \sum_{n=0}^{N-1-i} s(n)s(n+1) \tag{3.13}$$

The matrix of Eq. (3.11) is given by

$$\begin{bmatrix} R_0 & R_1 & \dots & R_{P-1} \\ R_1 & R_0 & \dots & R_{P-2} \\ . & . & . & \\ . & . & . & \\ R_{P-1} & R_{P-2} & \dots & R_0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ . \\ . \\ a_P \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \\ . \\ . \\ R_P \end{bmatrix} \tag{3.14}$$

This special (PxP) matrix with equal diagonal elements and symmetry is called a "Toeplitz" matrix (symmetric with all diagonal elements equal) and hence can be solved efficiently through several well-known procedures (Levinson-Durbin's Recursion Figure 3.2 [2], Schur [3], and Burg's solution [4]).

Initialization: l=0

$x^0(m)$ = scaled total energy in the "error" from an "order 0" predictor

= average energy in the speech frame $f(n;m)=s(n)w(m-n)$

= $r_s(0;m)$

Recursion: **For** l = 1,2,…, M,

1. Compute the lth reflection coefficient,
2. Generate the order-l set of LPC parameters,
3. Compute the error energy associated with the order-l solution,
4. Return to step 1 with l replaced by l+1 if $l < M$.

Figure 3.2. Levinson Durbin algorithm applied to window $n \in [m-N+1,m]$

3.2.2 The Covariance Method

An alternative to using a window is to use fixed interval of n=0,1,...,N-1 to compute the mean-squared error and to use the unweighted speech directly

$$E = \sum_{n=0}^{N-1} e^2(n) \tag{3.15}$$

The solution can be expressed as

$$\begin{bmatrix} F_{(1,1)} & F_{(1,2)} & \dots & F_{(1,P)} \\ F_{(2,1)} & F_{(2,2)} & \dots & F_{(2,P)} \\ . & . & . & \\ . & . & . & \\ F_{(P,1)} & F_{(P,2)} & \dots & F_{(P,P)} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ . \\ . \\ a_P \end{bmatrix} = \begin{bmatrix} F_{(1,0)} \\ F_{(2,0)} \\ . \\ . \\ F_{(P,0)} \end{bmatrix} \tag{3.16}$$

The resulting covariance matrix is symmetric but not *Toeplitz*, and can be solved efficiently using a Cholesky decomposition method.

After implementing all these techniques, we found that overall Levinson-Durbin's

recursion in Table 3.1 is the most simple and efficient algorithm for computing the values of a_k once $R(i)$ has been calculated. The experiment was performed for ten verbal commands; five command utterances were used as training and ten for testing for each command word. The accuracy calculated based on using dynamic time warping (DTW).

Table 3.1. Comparison between Algorithms that extract LPC

	Levinson Durbin's	**Schur's**	**Burg's**	**Cholesky**
Time	0.93 sec/10 msec	1.5 sec/10 msec	0.99 sec/10 msec	1.5 sec/10 msec
Accuracy	94%	92%	93%	95%

3.3 LPC Parameter Conversion To Cepstral Coefficients

Other LPC derived parameters include the log-area-ratio coefficients, cepstral coefficients, ParCor coefficients, and parameter weighted cepstral coefficients [5].

The log-area-ratio coefficients are given by

$$g_m = log\left(\frac{1-k_m}{1+k_m}\right) \qquad 1 \le m \le P \tag{3.17}$$

where k_m are the ParCor coefficients and g_m are log area ratio coefficients.

A very important LPC parameter set, which can be derived directly from the LPC coefficient set, is the set of LPC cepstral coefficients, $c(m)$. The recursion used is

$$c_o = \ln s^2 \tag{3.18a}$$

$$c_m = a_m + \sum_{k=1}^{m-1}\left(\frac{k}{m}\right)c_k a_{m-k}, \qquad 1 \le m \le P \tag{3.18b}$$

$$c_m = \sum_{k=1}^{m-1}\left(\frac{k}{m}\right)c_k a_{m-k}, \quad m > P \tag{3.18c}$$

where s^2 is the gain term in the LPC model.

Since the low-order cepstral coefficients are sensitive to the overall spectral slope, and the high-order cepstral coefficients are sensitive to noise, it is desirable to weight these parameters to minimize these effects.

The parameter weighted cepstral coefficients are described by

$$w_m = 1 + (Q/2)\sin(mp/Q) \tag{3.19}$$

$$h_m = w_m c_m \ , \qquad 1 \le m \le Q \tag{3.20}$$

where w_m is a weighting window known as bandpass filter.

The first order temporal cepstral derivative coefficients are

$$c_m(t) \approx m \sum_{k=-K}^{K} k c_m(t+k) \ , \qquad 1 \le m \le Q \tag{3.21}$$

where μ is a normalization constant and K = 3.

To reduce both the processing time and the storage capacity, we will introduce a new set of features based on using an average value of all the frames features. This approach depends on the fact that the speech signal is a short-term quasi-perodic signal, ergodic process. A characteristic of an ergodic process is stationarity of its statistics with respect to time. Therefore, a single equivalent frame of features for each command, based on the average value of all the frames of this word is features, should reflect the statistical properties of interest for this word.

3.4 Median LPC Technique

A median of the LPC analysis frames is calculated to produce MLPC.

Theoretically, after calculating LPCs in each frame from the utterance of the word, one takes the median of all corresponding coefficients over all frames; this will be MLPC. The MLPC represents the coded pronunciation Equation (3.24).

To verify that these parameters are useful for speech recognition, we conducted some isolated word recognition experiments. Two commonly used speech recognizers were implemented, one is based on the classical Dynamic Time Warping (DTW) algorithm, and the other is based on back-propagation multi-layer perceptrons (MLPs) algorithms. Both approaches have been used widely and discussed extensively in the literature [6, 8-11]. The third one is based on the new technique; a modular artificial neural network (MNN) algorithm was designed and will be discussed in the next chapter. The basic conclusion from our experiments is that the MLPC has useful information and can improve the speed calculation with good speech recognition performance.

3.4.1 Algorithm

The LPC-based approach performs spectral analysis with a linear all-pole modeling constraint. It is fast and provides extremely accurate estimates of speech parameters. Here we calculate the MLPC by taking median of LPC based method.

The LPC parameters were calculated from each frame, and finally MLPC are calculated as illustrated in Figure 3.3. We divided the utterance into frames, each frame width 10 msec with 50% overlap of the previous frame, and calculated 30 LPC coefficients for each frame. (Normally, using the standard LPC method, only 12 or 16 coefficients might be calculated). We then take the median of the corresponding coefficients of all frames to obtain the MLPC coefficients.

$$s(n) = \sum_{k=1}^{P} a_k s(n-k) \tag{3.22}$$

Where $s(n)$ is speech samples, a_k are LPC coefficients, P (=30) is the LPC order.

$$L = \begin{bmatrix} P_{11} & P_{12} & \bullet\bullet\bullet & P_{1f} \\ P_{21} & P_{22} & \bullet\bullet\bullet & P_{2f} \\ & & & \\ P_{301} & P_{302} & \bullet\bullet\bullet & P_{30f} \end{bmatrix} \tag{3.23}$$

where $P_{11} - P_{301}$ are the features for the first frame while $P_{1f} - P_{30f}$ are the features for the f frame of the signal. Therefore, the MLPC method creates a vector of coefficients each component of which is the median of one of the rows of L, denoted:

$$MLPC = \text{median}(L) \tag{3.24}$$

where median(L) means the vector medians of each row.

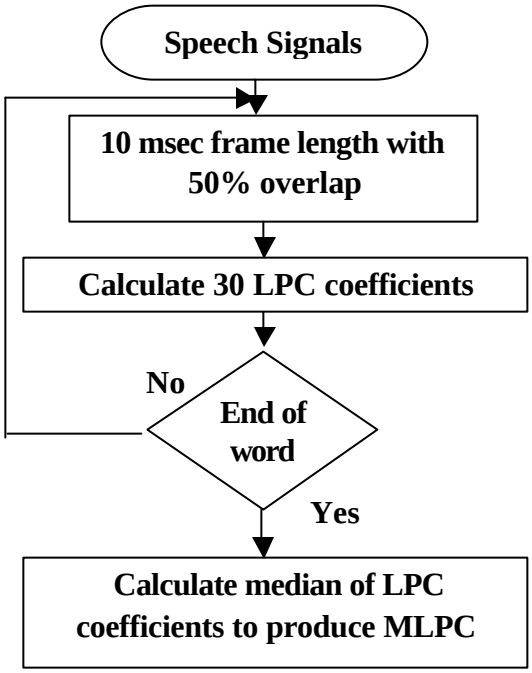

Figure 3.3. Block diagram for calculating LPC and MLPC

3.4.2 Database

The evaluation vocabulary comes from the Microsoft Word command menu (ex. *open*, *close*, *save*, *exit*, etc.). To reduce experimental time, we made use of only 10 verbal

commands (*open, save, close, new, print, help, table, spelling, normal* and *exit*). The database consists of thirty utterances for each command. All experiments were conducted with ten commands used as training data and the remaining twenty commands used as test data.

3.4.3 Speech Recognizer

The objective was to see whether the MLPCs are applicable for speech recognition with good recognition rates (and minimum calculation times). We used only the known speech recognition techniques (which are not optimal for this special recognition task). Two types of speech recognizers were implemented to compare with the MLPC method: DTW and MLP.

*** DTW Recognizer**

The DTW technique, used widely in the area of isolated word recognition, provides an efficient procedure for obtaining a nonlinear time alignment between each reference pattern and the unknown sample [6, 7]. Here, we designed the recognition to depend on extracting features in the normal way in addition to the suggested median linear predictive method. From the first ten utterances of each command we take the median and save them as references. The remaining twenty utterances for each command are used in testing. This applies to both cases.

*** MLP Recognizer**

Due to the complexity of implementing the conventional technique, only the median linear predictive coding will be tested. The MLP design is 30 units input layer, 12 units hidden layer and 10 units output layer, and uses the back-propagation algorithm to update the weights in training multi-layered perceptrons (MLPs).

*** Experimental Results**

After applying endpoint detection on the available command record, we start to extract the LPC, MLPC, and build the recognition system using the DTW technique. We use ten commands for training and twenty commands for testing. We take 10msec frames with 50% overlap. In each frame we take 12 LPC coefficients (only 30 MLPC coefficients are taken for the total utterance). The same procedures are repeated again for the MLP technique, but in this case for testing MLPC only.

*** Conclusion**

The usefulness of MLPC for speech recognition was studied.

Let F_1 and F_2 represent the number of frames for LPC and MLPC techniques, respectively.

Let P_1 and P_2 represent the number of poles (as in Eq. (3.23)) featured for LPC and MLPC, respectively.

Let d_1 and d_2 represent the number of commands used for both techniques.

Then, the relative performance of the proposed technique with respect to the conventional DTW can be evaluated using the complexity function (CF) defined as

$$CF = \left(1 - \left(\frac{F_1 \times P_1 \times d_1}{F_2 \times P_2 \times d_2}\right)\right) \times 100 \qquad (3.25)$$

For simplification, let us assume all utterances have the same length of one second and we take 10 msec as frame length without overlap. In the conventional method 12 LPC coefficients are calculated. This means 120 vector features (12 LPC by 10 frames). In the new technique 30 LPC coefficients are calculated in each of the 10 frames, and finally the median is taken to return 30 coefficients.

After applying Equation (3.25), the MLPC will decrease the complexity by approximately 75% with a recognition rate for Microsoft Word commands of 88%, while 94% recognition rate was achieved in the conventional techniques of Dynamic Time Warping, and 94% in a multi-layer perceptron recognizer. Table 3.2 illustrates the

comparison.

Table 3.2. Comparison between Conventional and new techniques

	Conventional Technique	MLPC Technique
Dynamic Time Warping		
Accuracy	94%	88%
Time/Command	~20 sec.	~30 msec.
Multi-Layer Perceptrons		
Accuracy	?	94%

Reference

[1]. Dautrich, B. A., et al., *"On the effects of varying filter bank parameters on isolated word recognition,"* IEEE Trans., Vol. ASSP-31, no. 4, pp. 793-807, August, 1983.

[2]. Levinson, S. E., *"Some experiments with a linguistic processor for continuous speech recognition,"* IEEE Trans., Vol. ASSP-31, no. 6, pp. 1549-1556, December, 1983.

[3]. Le Roux. J., and C. Guegen, *"A Fixed point computation of partial correlation coefficients in linear prediction,"* Proc. ICASSP-77, IEEE Press, New York, pp.742-743, 1977.

[4]. Burg. J. P., *"Maximum entropy spectral analysis,"* PhD Thesis. Stanford University, May. 1975.

[5]. D.P. Prezas, J. Picone, and D.L. Thomson, *"Fast and Accurate Pitch Detection Using Pattern Recognition and adaptive Time-Domain Analysis,"* Proc. Intl. Conf. Acoust., Speech & Signal Proc., pp. 109-112, April 1986.

[6]. L. R. Rabiner and B. H. Juang, *"Fundamentals of speech recognition,"* Englewood Cliffs, NJ: Prentice Hall, 1993.

[7]. S. B. David and P. Mermelstein, *"Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,"* IEEE Trans. Acoustic. Speech, Signal Processing ASSP-28, pp. 357-366, 1980.

[8]. Hiroaki Sakoe and Seibi Chiba, *"Dynamic Programming Algorithm Optimization for Spoken Word Recognition,"* IEEE Trans. Acoustic. Speech, Signal Processing ASSP-26, pp. 43-496, 1978.

[9]. Cory S. Myers and Lawrence R. Rabiner, *"Connected Digit Recognition Using a Level-Building DTW algorithm,"* IEEE Trans. Acoustic. Speech, Signal Processing ASSP-29, pp. 351-363, 1981.

[10]. Timothy R. Anderson and Harry G. Armstrong, *"Speaker independent Phoneme*

recognition with an auditory model and a Neural Network: A Comparison with Traditional Techniques," IEEE Trans. Acoustic. Speech, Signal Processing, pp. 149-152, 1991.

[11]. Peter T. Brunet, A.S. Pandya and Carlos V. Pinera, *"Artificial Neural Networks for Phoneme Recognition,"* IEEE Trans. Acoustic. Speech, Signal Processing, pp. 4473-4478, 1994.

Chapter 4

Using Modular Neural Networks to Overcome MLP Limitations

4.1 Introduction to Neural Networks

The working of the human brain is still largely a mystery to man. The brain is amazingly powerful in that it can perform a wide variety of solving problems including thinking, talking, remembering, feeling, and learning. In the brain, the neuron is the main cellular unit of the nervous system. Each neuron receives and combines signals from many other neurons and produces an output signal to its axon to perform certain actions or transfer information. Neurocomputing has emerged from this inspiration.

4.1.1 Neuron Model

The field of Artificial Neural Networks (ANN) was born in the 1940s when McCulloch and Pitts [1] proposed a computational model based on a simple neuron-like logical element consisting of many inputs, corresponding to dendrites (cell connections to other neurons) that are connected through synaptic junctions (variable weights). The model is described by:

$$Y = f\left(\sum_{i=0}^{N} x_i w_i\right) \tag{4.1}$$

where x_i = inputs, $i = 0, 1, 2, \ldots, N$,

x_0 = bias = 1,

w_i = variable (adjustable) weights.

In this model, x_0 is used to provide a bias to the activation function $f(.)$. McCulloch and Pitts did not provide any method through which the neuron could adapt its weights in a "learning" process. Donald Hebb [2] described a learning rule for adapting the connection strengths of these artificial neurons. The Hebb rule and later the delta rule became the basis of almost all ANN research that was to follow. Hebbian learning is usually taken as the following weight update:

$$\Delta w_i = mY(x)x_i, \qquad i = 0,1,...,N \tag{4.2}$$

where x represents the vector of $(N+1)$ inputs and μ is the learning paramter. Figure 4.1 illustrates the McCulloch-Pitts neuron model used widely in many different ANN paradigms.

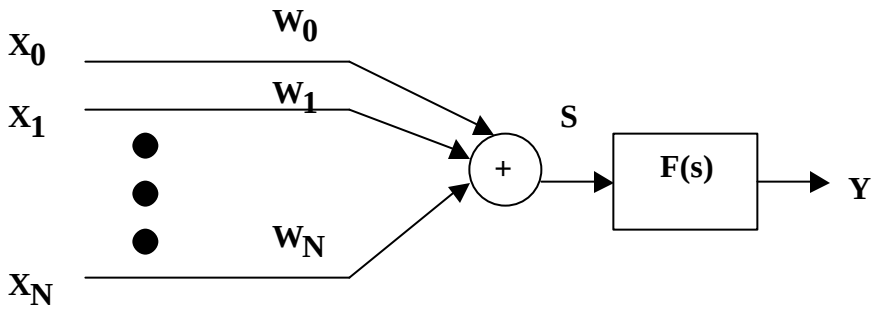

Figure 4.1. The McCulloch-Pitts Neuron

4.1.2 Perceptron

After an extended period of early activity, only a few researchers in the USA, Europe and Asia [3, 4] managed to maintain financial support for their work However, since its rebirth in the early 1980s, the ANN field has experienced extremely rapid growth, attracting followers from many disciplines ranging from physics and engineering through physiology and psychology.

Rosenblatt [3] demonstrated some practical applications using the perceptron. The perceptron is a single level connection of McCulloch-Pitts neurons. The perceptron is

capable of (linearly) separating the input vectors into pattern classes by a hyperplane. Figure 4.2 shows the perceptron of N-features (inputs) and M-classes (outputs). This perceptron can be described by:

$$y_i = f\left\{\sum_{j=0}^{N} w_{ij} x_j\right\}, \qquad i = 1,2,...,M \tag{4.3}$$

where the function $f(x) = 1(x)$, the unit-step function.

Rosenblatt derived a learning rule, based on weights adjusted in proportion to the error between the output neurons and the target outputs. The weight adjustments are given by

$$\Delta w_{ij} = m(y_i^d - y_i)x_j \tag{4.4}$$

where $i = 1, 2, \ldots, M$, $j=0,1,\ldots,N$, and y^d are the desired output vectors.

The perceptron represented a major step in the application of ANNs. However, many problems cannot be solved without incorporating additional layers.

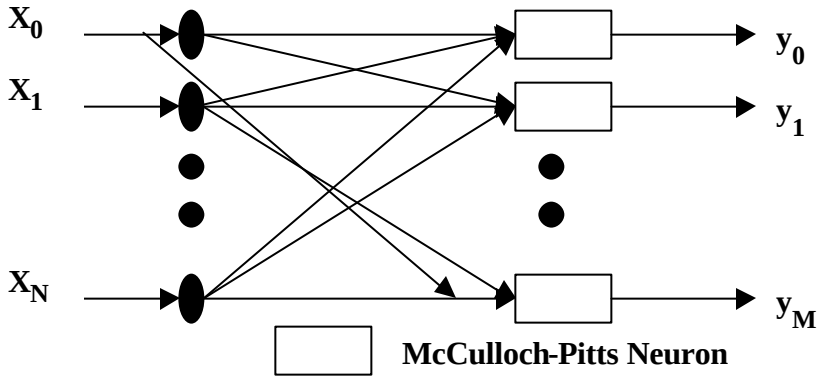

Figure 4.2. Single-Layer Perceptron

4.1.3 Multi-layer Perceptrons

The capabilities of single-layer perceptrons are limited to linear decision boundaries and simple logic functions. The single-layer perceptrons cannot realize e.g. the simple XOR problem. This led (later) to the development of multi-layer perceptrons. In general, multi-layer perceptrons (MLPs) consist of an input layer, one or more hidden layers, and an output layer. Figure 4.3 illustrates a feed-forward multi-layer MLP network. The MLP [6, 7] networks overcome many of the limitations of the single-layer perceptrons. Instead of the hard-limiting activation function, other nonlinear smoothed functions are employed to give a similar amplitude limiting effect, such as the sigmoid, the hyperbolic tangent, etc., defined as follows.

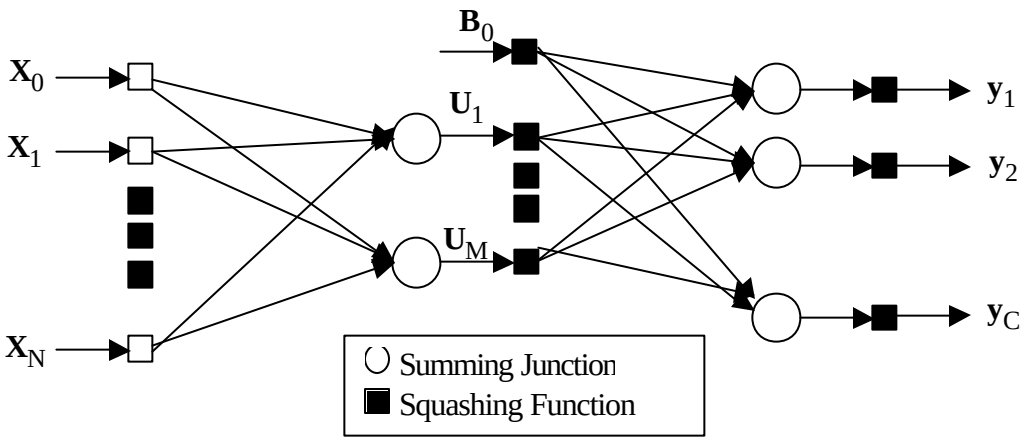

Figure 4.3. Multi-Layer Perceptrons

- Sigmoid function

$$f(v) = \frac{1}{1+e^{-v}}$$

- Hyperbolic tangent function

$$f(v) = \tanh(v) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}$$

Using perceptrons in parallel and cascade structures, arbitrary complex decision boundaries and Boolean functions can be modeled. However, it has been shown that many different artificial neural network structures can be used to model the same mapping. Finding the optimal structure that yields the best performance is not easy to determine. Therefore, there exist rules and algorithms to find an appropriate structure.

The back-propagation algorithm is usually used to train MLPs networks. The back-propagation training algorithm employs an iterative gradient-descent method that minimizes the mean squared error between the desired output and the MLPs output Figure 4.4.

Initialization: Assign initial (random) weight values.

Select learning constant $\eta(0)$ (may be decreased with time).

Recursion: **For** p=1,2,…,[continuously iterate through training patterns]

For l=L,L-1,…,1 (layers)

For all k and j in layer l (weights)

Compute error.

Compute gradient.

Next weight

Next l

Next p

Termination: Stop when weights change negligibly according to some criterion.

Figure 4.4. Back-propagation Algorithm

4.1.4 Neural Network Operation and Benefits

There are two distinct operational modes of neural networks. They are the learning mode and the recall mode. The learning mode is the process of modifying the network weights in response to a set of inputs in such a way that the outputs produced by the network should follow the desired outputs to an acceptable level of error. There are three

types of learning: *supervised*, *unsupervised*, and *reinforcement* [8, 9]. When the desired outputs are used to train the network, the learning process is called *supervised*. When no outputs are presented to train the network, *unsupervised* learning process is observed. The last type of learning is *reinforcement* learning in which only grades of good or bad are presented to the network during training.

The *recall* mode is the process of using the trained network, to find out how well the network has learned. Most often, the network will be presented with a different set of data from the training data. The outputs produced by the neural network during *testing* are compared against the desired outputs to determine the network's performance.

Several useful properties of ANNs include:

- Learning by example: the ability to create their own rules by learning from training patterns presented to the networks.
- Fault tolerance: the response of the networks only changes slightly if some processing elements are defective or damaged, due to the fact that the information is not stored in one place. It is distributed across its numerous weights.
- Input-output nonlinear mappings: ANNS may achieve arbitrary vector mappings.
- VLSI massively parallel implemention ability: If implemented in a parallel architecture, even complex ANNs can have very short recall times..
- Adaptivity: the neural networks have the ability to change their weights to adapt to a new environment.

4.1.5 Problems of MLPs

Artificial neural networks (ANNs), because of their good mapping and generalization capabilities, can provide acceptable recognition rates for a given small set of vocabulary. But the most widely used monolithic ANNs suffer from a serious

drawback of long retraining time and increased complexity when new information is to be added to an already learned vocabulary.

We will overcome this limitation by using *a modular artificial neural network* architecture. This architecture is composed of individual neural networks called expert neural networks; each expert recognizes a word or a subset of words of a given vocabulary. The architecture decomposes the overall input data set in such a way that each expert neural network learns to discriminate between a specific word and the rest of the words in the input data set with high accuracy. Recognition of a set of verbal commands for Microsoft Word is presented as an illustrative example of the proposed methodology.

To summarize, ANNs are an interconnection of simple computational elements or nodes connected by links with variable weights arranged in layers. Each node or neuron, computes a weighted sum of the outputs of the connected nodes from the preceding layer, and passes the sum through an activation or squashing function [10]. The use of a sigmoidal squashing function ensures that the output value of all nodes in the ANN are bounded. A bias is added to the sum in order to shift the threshold of the activation function and expedite the ANNs learning process. ANNs with multiple layers of neurons (at least two) have the ability to learn arbitrary complex decision boundaries.

4.2 Modular Neural Network Definition

A neural network is said to modular if the computation performed by the network can be decomposed into two or more modules (subsystems) that operate on distinct inputs without communicating with each other. The outputs of the modules are mediated by an integrating unit that is not permitted to feed information back to the modules. In particular, the integrating unit both (1) decides how the outputs of the modules should be combined to form the final output of the system, and (2) decides which modules should learn which training patterns.

Modularity permits us to solve a complex computational task by dividing it into simpler subtasks and combining their individual solutions.

A modular network exhibits:

- Supervised Learning:

  Supervised learning is exemplified by an external "teacher" that supplies the desired responses (target patterns) needed to train the different modules of the network. However, the teacher does not specify which module of the network should produce each desired response; rather, it is the function of the unsupervised learning paradigm to do this assignment.

- Unsupervised Learning:

  Unsupervised learning is exemplified by modules "competing" with each other for the right to produce each desired response. The integrating unit has the role of "mediating" among the different modules for this right.

Consequently, the modules of the network tend to specialize by learning different regions of the input space. However, the training is done automatically for each module. In the competition, the winner is the module whose output most closely matches the desired response, and the other modules are the losers. Furthermore, each module receives an amount of training information that is proportional to its relative ability to learn. This means that the winning module receives more training information than the losing modules.

4.2.1 Architecture

The structure consists of K supervised modules called expert networks, and an integrating unit called a *gating network* that performs the function of selecting the expert network.

Let the training examples be denoted by input x of dimension p and desired response (target output) vector d of dimension q. The input vector x is applied to the expert networks and the gating network simultaneously. Let y_i denote the q-by-1 output vector of the i^{th} expert network, let g_i denote the activation of the i^{th} output neuron of the gating network, and let Y denote the q-by-1 output vector of the whole modular network.

Y is the output of module k at which q is the maximum. We may the write

$$Y = \max_{i=1}^{k} (g_i y_i) \tag{4.5}$$

The goal of the learning algorithm used to train the modular network of Figure 4.5 is to model the probability distribution of the set of training patterns {x,d}. We assume that the patterns {x,d} used for training are generated by a number of different regressive processes. To do the learning, the expert networks and the gating network are all trained simultaneously [11].

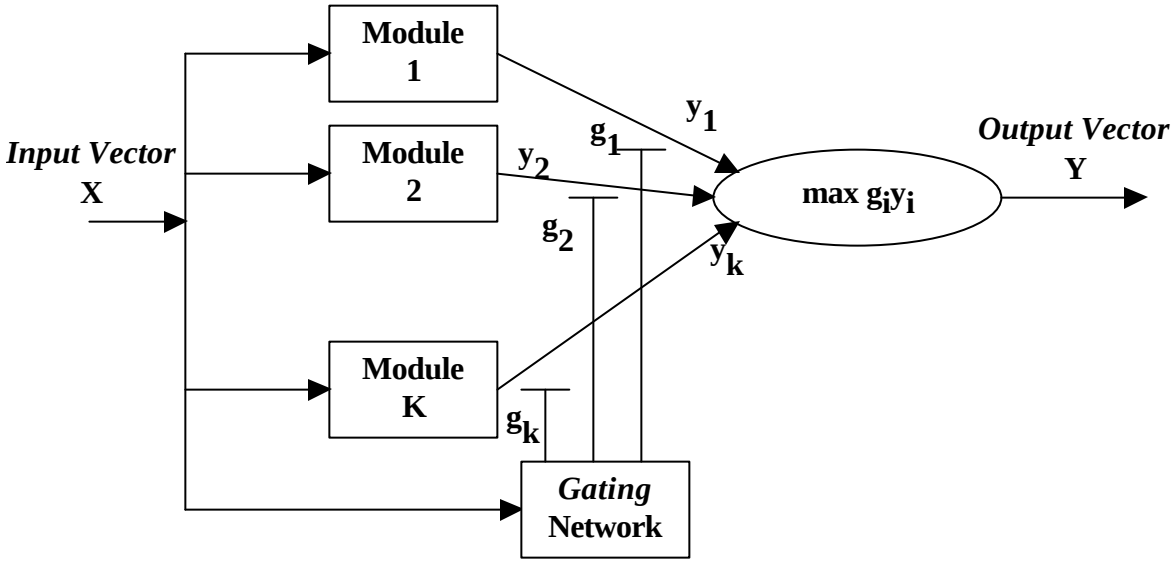

Figure 4.5. Block diagram of a modular network, the outputs of the expert networks (modules) can selected by a gating network

4.2.2 Advantages

- Speed of learning

If a complex function is naturally decomposable into a set of simpler functions, then a modular network has the built-in ability to discover the decomposition. Thus, a modular network is able to learn a set of simpler functions faster than a multi-layer perceptron can learn the undecomposed complex function.

For example, consider the linear function defined as

$$g(x) = \begin{cases} x, & x > 0 \\ -x, & x \leq 0 \end{cases} \qquad (4.6)$$

This function can be learned by a multi-layer perceptron with at least a single hidden layer. Alternatively, a modular network consisting simply of two linear neurons and an integrating unit can learn it. One neuron learns the function $g(x)=x$ for $x>0$, and the other neuron learns the remaining function $g(x)=-x$ for $x£0$. The role of the integrating unit is to select the appropriate neuron in the appropriate context. Assuming that the integrating unit is able to learn its role relatively easily, a modular network should be able to learn the function $g(x)$ faster than a multi-layer perceptron for two reasons:

(1) It has no hidden neurons, and

(2) Each of its two modules (made up of signal neurons) is required to learn only a linear function [11].

- Data Representation

The representation of input data developed by a modular network tends to be easier to understand than in the case of an ordinary multi-layer perceptron by virtue of the ability of a modular network to decompose a complex task into a number of simpler tasks. This property was demonstrated by Rueckl et al. (1989), who used simulated retinal images to perform two relatively independent tasks: object recognition ("what"

task), and spatial location ("where" task). Comparative computations involving two different models were investigated:

- An unsplit model, consisting of a fully connected multi-layer perceptron with a single hidden layer.
- A split model, in which the synaptic weights between the hidden and output layers were partitioned into two subsets, one subset connected only to the output neurons representing the "what" task output and the other subset connected only to the remaining neurons representing the "where" task.

The two models were tested on the same simulated retinal data. At each time step of the simulation, one of nine objects was placed at one of nine corners of a simulated retina. The "what" task is to identify the object, and the "where" task is to identify its location in the retina. When the unsplit model is used to resolve these two relatively independent tasks, the same set of hidden neurons is forced to represent information about both tasks. In the case of a split model, different sets of hidden neurons are used to represent information about the two tasks. The split model was found to develop more efficient internal representations.

- Hardware Constraints

In a brain, there is a physical limit on the number of neurons that can be accommodated in the available space. Ballard (1986) suggested that this limitation compels the brain to adopt a modular structure, and that the brain uses a coarse code to represent multidimensional spaces. To represent a space of dimension K, it is hypothesized that the number of neurons required to do the representation is N^k / D^{k-1}. Where N is the number of just-noticeable differences in each dimension of the space and D is the diameter of the receptive field of each neuron. With a limit imposed on the number of neurons in the cortical area of the brain, the representation of high-dimensional spaces is distributed in different areas that compute different functions. In an analogous manner, it may be argued that, in order to reduce the number of neurons in an artificial neural network, the representation of multidimensional spaces may be distributed among multiple networks [11].

Now, let us examine the MANN vs. MLP for the speech recognition problem.

4.2.3 Vocabulary System

The speech data for this problem are given by a collection of spoken commands from the Microsoft Word command menu, i.e, *Open, Close, Save, Exit*, etc. This data-base consists of ten utterances of each command. Since the linear predictive coefficient (LPC) model is one of the most powerful speech analysis techniques, this technique is used to encode the voice commands into real valued vectors, which will be subsequently used as the input for the neural networks for both training and testing purposes. All experiments were conducted in a speaker dependent fashion. Five command utterances were used as training and ten for testing for each command word.

4.2.4 MANN Training and Operation

The structure of Figure 4.6 illustrates the Modular Artificial Neural Network (MANN) [12].

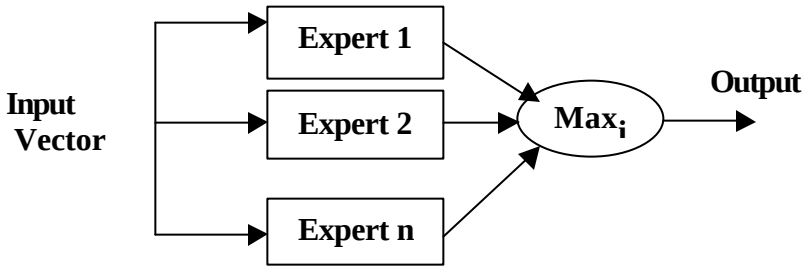

Figure 4.6. Modular Neural Networks

For the training, every expert network receives the same input data during training. However, a specific expert is trained to recognize a single command and to reject all

other commands. So, for example, Expert 1 is trained to provide (ideally) a unity response for all inputs (LPCs) that were derived from the word *'File'* and (ideally) a zero response for all other input, representing commands such as *Edit*, *View*, etc. Keep in mind that several speech utterances for each command word are used to train the MANN. The back-propagation algorithm can be used effectively for the training of each expert network.

Since each expert network can be trained off-line, the system becomes easily extensible, i.e. a new command can be added by adding a new expert network trained with the new command. This assumes that the new word is significantly different from the current vocabulary. If this is not true, then some retraining of the 'neighboring' words may be necessary; but typically this is a small subset of the entire database.

4.2.5 Experimental Results

The first experiment was performed for ten verbal commands. The global monolithic ANN architecture consisted of 30, 12 and 10 nodes in the input layer, hidden layer and output layers, respectively. Inputs are linear predictive coefficients (LPCs) of the speech commands and the training (desired) output is a binary valued vector. This vector has a binary value 1 at the position indicating the class of the pattern presented for training and 0 for rest of the pattern classes. This architecture is optimal if one takes into consideration the architectural complexity and training time. At a training error rate of 0.02, the architecture shows a 90% test accuracy. The training time was approximately 1:00 hour. If a new command is added, then the retraining time was approximately 1:34 hours. At a training error rate of 0.01 the architecture showed 94% test accuracy, the training time was approximately 2:30 hours and the retraining time for a new command was approximately 3:00 hours.

The second experiment was carried out using the modular artificial neural network architecture. The architecture consisted of ten expert neural networks with each expert neural network consisting of 30 units in the input layer, 5 units in hidden layer and 1 output layer. The input vector is the same as that for the monolithic neural network. But the expert networks output only one binary valued number, indicating the class of the input training pattern. At training error rate 0.02, the MANN architecture showed 92% test accuracy. The training time is approximately 20 seconds and when adding a new command the retraining time remains the same, i.e., approximately 20 seconds for the new expert neural network. At a training error rate 0.01 the architecture has a 96% test accuracy; the training time is approximately 50 seconds, and the retraining time is approximately the same. A more complete comparison is illustrated in Table 4.1.

4.2.6 Conclusions

A special structure of modular artificial neural networks (MANNs) is shown to outperform global monolithic neural networks in classification and distinguishing between multiple verbal command patterns. The modular system was even able to perform its task with a very short training time. The system can accommodate and classify new commands in a very convenient way. Whenever a new command needs to be added, a new expert neural network is added to the already exiting modular neural network architecture. The speech data for that command is fed to the system; and, after finding those experts with small classification errors, they are retrained with the new expert for this new command speech data. A 4% error with a MANN can be achieved from the results shown. If there exist two or more experts with the same classification error, then the results are rejected and the user is asked to repeat the command. The presented methodology removes the limitation of extending the system at the expense of large retraining times.

TABLE 4.1. System Comparison		
	MLPs	**MANN**
Training Error rate 0.02		
Training Time	1.0 hours	20 Sec. For each 10 experts
Architectures	30-12-10 MLPs	30-5-1 MLPs
Test Accuracy	90%	92%
Retraining Time	1:34 hour	20 Sec. For new expert and any old expert as necessary
Training Error rate 0.01		
Training Time	2.30 hours	50 Sec. in each 10 experts
Architectures	30-12-10 MLPs	30-5-1 MLPs
Test Accuracy	94%	96%
Retraining Time	~3:0 hour	50 Sec. For new expert and any old expert as necessary

4.3 Pyramidal Modular Neural Network (PMNN) Approach

In order to design a human/computer interface system using a modular technique, a Pyramidal Modular Neural Network (PMNN) architecture was chosen. The PMNN is comprised of a set of pre-trained individual neural networks called "expert" neural networks. Each expert neural network is trained using back-propagation learning to recognize a word or a subset of words from a given vocabulary. The learning process and the way expert neural networks are stacked enables the architecture to decompose the overall input data set in a way that each expert neural network discriminates between a specific word from the rest of the words in the input data set with high accuracy. After the training of the individual expert neural networks is complete, all expert neural networks are stacked in a hierarchical or pyramidical fashion with experts at one level communicating with only experts at the next lower level. This hierarchical architecture speeds up online recognition and increases the accuracy as well. The structure of Figure 4.7 illustrates the Pyramidal Modular Neural Network (PMNN).

4.3.1 Vocabulary System

The vocabulary data used is given by nearly all the commands from the Microsoft Word command menu, i.e., *Open*, *Close*, *Save*, *Exit*, etc.

The PMNN is designed to be in a hierarchical mode in the following way. There are three levels of MANNs. The first level recognizes the top-level menu in Microsoft Word (i.e. *File*, *Edit*, *View*, *Insert*, *Format*, *Tools*, *Table*, *Window* and *Help*). The second level recognizes the sub menu under each command in the first level (e.g. *File* → [*New*, *Open*, *Close*, *Save*, *Save As*, etc.]). The third level is used to recognize the sub menu commands under each of the second level menu commands (e.g., *Insert* → *Picture* → [*Clip Art*, *From file*, *Autoshape*, *Wordart* and *Chart*]). Figure 4.8 illustrates the PMNN the single command to *customize* the *toolbar* (under *view*).

The speech database is organized in a hierarchical manner. This means that if one needs to run the Microsoft Word command '*Open*' then he/she must speak the command '*File*' and then the (desired) command '*Open*'. The command '*File*' will open the file menu from the first hierarchial level. The second level can then recognize the '*Open*' command from the subsequent menu level which contains commands like *New*, *Open*, *Save*, *Save As*, etc.

This database consists of twenty utterances of each individual command.

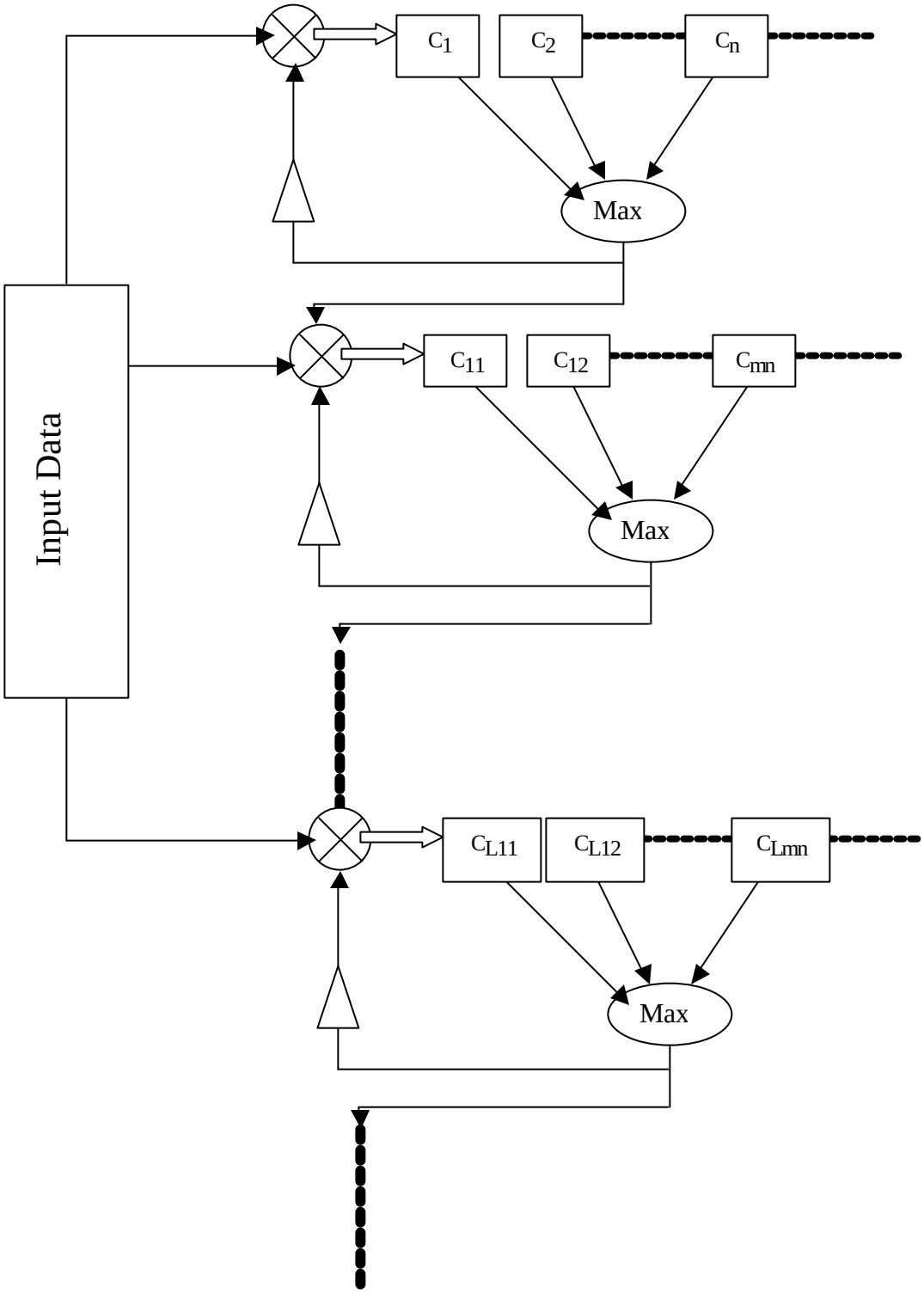

Figure 4.7. Pyramidal Modular Neural Network Architecture

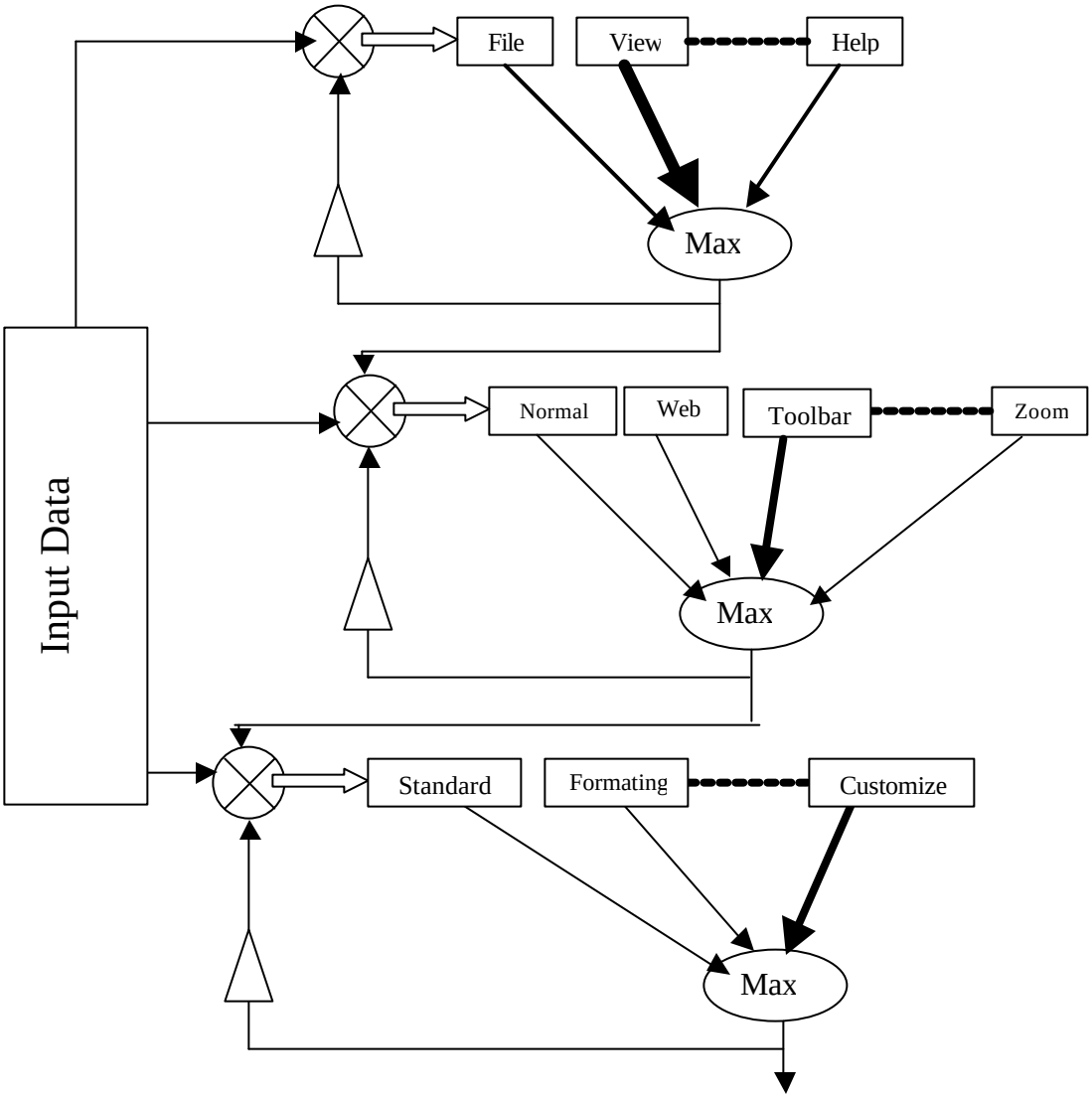

Figure 4.8. PMNN for single command representation

4.3.2 Conclusions

The results indicate that the Pyramidal Modular Neural Network approach achieves 87.2% overall recognition accuracy based on 3000 (150 commands X 20 templates) speaker commands using a 10 kHz sampling frequency.

References

[1]. MeCulloch, W.S. and Pitts, W., *"A Logical calculus of the ideas immanent in nervous activity"*. Bulletin of Mathematical Biophysics. 5,11 5-33, 1943.

[2]. Hebb, D.O., "*The Organization of behavior,*" Wiley, New York, 1949.

[3]. Rosenblatt. F., "*Principles of Neuro-Dynamics,*" Spartan, Washington, DC, 1962.

[4]. Widrow, B. and Hoff, M.E., *"Adaptive switching circuits,"* WESCON Convention Record, Part IV, 96-104, 1960.

[5]. Hopfield, J.J., *"Neural Networks and Physical Systems with emergent collective computational abilities,"* Proc. Nat. Acad. Sci., 74, 2554-8, 1982.

[6]. Rumelhart, D.E., Hinton, G.E. and WiLliams, R., "*Learning internal representation by error propagation,*" Nature, 323.533-6, 1986.

[7]. Sejnowski, T. and Rosenberg, C., *"NETtalk: A parallel network that learns to read aloud,"* The Johns hopkins University EE and CS Technical Report, 1986.

[8]. Lipprnann, R.P., *"Pattern classification using neural networks,"* IEEE Communications Magazine, 27(11), 47-64, 1989.

[9]. Hanson, S.J. and Burr, D.J., *"What connectionist models learn: Learning and representation in connectionist networks,"* Behavioral and Brain Sciences, 13(3), 471-518, 1991.

[10]. Haykin, Simon (1994), "*A Comprehensive Foundation,*" Neural Networks, Macmillan College Publishing Company, New York, NY.

[11]. Jordan, M.I. and Jacobs, R.A. (1994), *"Hierarchical mixtures of experts and the EM algorithm,"* Neural Computation, 6:181-214.

[12]. Moatassem M. Abdallah, Farooq Azam, H.F. VanLandingham, *"A Modular Neural Network Approach to the Speech Recognition Problem"*, Smart Eng. System Design, 1079-1083,1999.

Chapter 5

HCI Design: System Components

In order to design and build a successful Human Computer Interface (HCI) system, many other components need to be designed and built correctly in addition to a good speech recognition method [1]. In general, the following components are required to be understood and designed:

1. System Hardware

2. System Building Blocks

- Data Acquisition Module.
- Preprocessing Module.
- Feature Extraction Module.
- Training and Recognition Module.

5.1 System Hardware

To implement the human computer interface, the following hardware components have to be present. IBM PCs (or compatibles) with minimum hardware configuration for the PC is as follows:

486 or higher processor.

4MByte RAM.

VGA Card and VGA monitor.

Add-on voice card with the following specification (or better):

Sampling rate (minimum): 10 kHz.

Number of bits (minimum) : 8 bits.

Microphone or a combination of microphone-speaker set.

5.2 System Building Blocks

The different components of this system are discussed in this section

5.2.1 Data Acquisition Module

The purpose of data acquisition is the capture of a speech utterance. Speech, which is produced in a form of acoustic waves, is transformed into a form that is suitable for digital processing.

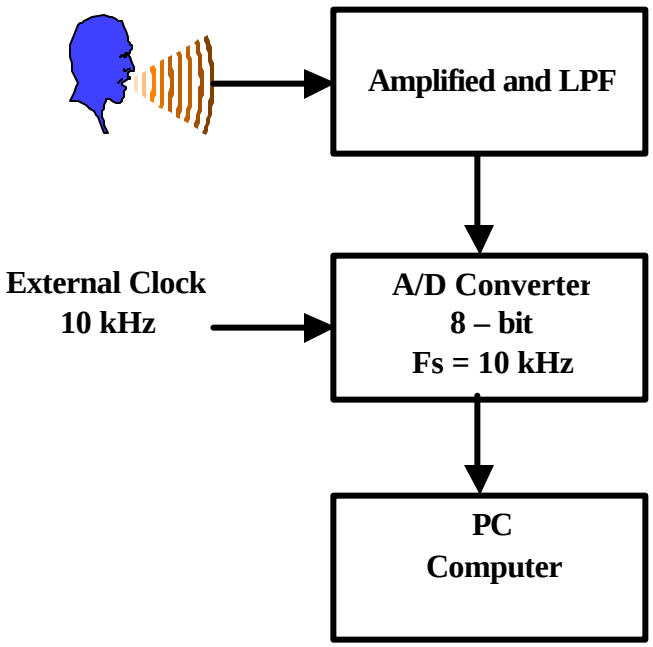

Figure 5.1. Block diagram of Data Acquisition System

The block diagram in Figure 5.1 is the system that has been used to digitize the speech. The words are uttered by the speaker into a microphone connected to a module that contains an amplifier and a Low Pass Filter (LPF). The cutoff frequency of the LPF is set to 5kHz, and a 10kHz sampling rate was used. An 8 bit Analog-to-Digital (A/D) converter was used to digitize the amplified and filtered speech signals. This A/D converter was clocked externally.

The output of the A/D converter is saved as an audio file in a "RIFF" audio wave format. The "RIFF" header format is described below:

"RIFF"	4 bytes (unsigned char).
xxxx	4 bytes (unsigned long) = total file size after this line read.
"WAVE"	4 bytes (unsigned char).
"fmt"	4 bytes (unsigned char).
xxxx	4 bytes (unsigned long).
xx	2 bytes (unsigned int) = format tag PCM = 1.
xxxx	4 bytes (unsigned long)- sampling rate.
xxxx	4 bytes (unsigned long)= average bytes per second.
xx	2 bytes (unsigned int) = block alignment.
xx	2 bytes (unsigned int)= bits per sample.
"Data"	4 bytes (unsigned char).

In this system, the audio wave recorder is used to record a speaker voice with the following parameters:

- Data Format: PCM.
- Sampling Frequency: 10 kHz.
- Sampling Resolution: 8 bits.
- Recording Channel: monaural.

The above recording parameters are close to telephone quality and are commonly used in most HCI systems.

5.2.2 Preprocessing Module

An important problem in speech processing is to detect the presence of speech in background noise. This problem is often referred to as the End Point Detection (EPD) or location problem [2]. By accurately detecting the beginning and end of an utterance, the amount of processing of speech data can be kept to a minimum.

A simple technique for EPD using thresholds is to use the energy and Zero-Crossing Rate (ZCR) to distinguish between noise and speech. In this technique, ZCR is used to overcome the problem of excluding weak unvoiced fricatives in cases when the threshold is set high enough to exclude the noise caused by the speaker. Briefly, the idea of this technique is to determine experimentally two energy threshold levels: the lower threshold (ITL) and the upper threshold (ITU) as well as one ZCR threshold. From speech characteristics, speech signals (either voiced or unvoiced) have higher energy than background noise. But in some cases, when the speaker causes noise, its energy may exceed the weak unvoiced fricatives. For this reason, the ZCR threshold is set such that the unvoiced sounds can be separated from the noise intervals. Also, from the characteristics of unvoiced sounds, they have higher ZCRs than noise. In general, noise has a ZCR higher than voiced sounds. Hence, this technique determines the point at which the beginning or end of the utterance is assumed according to the two energy thresholds. An estimate of the beginning or end of an utterance is obtained when the signal energy exceeds ITL and continues to increase until it exceeds ITU before falling below ITL. When this point is obtained, an interval of the signal (prescribed in length as a number of frames) is updated using the ZCR threshold. The number of frames that have occurred when the ZCR exceeds the set ZCR threshold is counted. If the number of times the threshold is exceeded is three or more, the beginning point is set back to the first point (in time) at which the threshold was exceeded. Figure 5.2 is an example. More details of the EPD algorithm are presented in Section 2.4.

5.2.3 Feature Extraction Module

This module extracts the desired features. Primarily, the linear predictive coefficients and their other coefficients are used as speech features in this system, including the linear predictive coding coefficients (LPC), the partial correlation coefficients (ParCor), Cepstral coefficients (Cep) and weighted Cepstral coefficients (CepW) [3, 4, 5] Figure 5.3.

To evaluate the performance of the system on different methods, it is necessary to build the training speech features database first.

The autocorrelation method and Durbin's recursive method were used to extract the linear predictive parameters.

We divide each command template into equal segments or windows by using a Hamming window of length 10 ms with 50% overlap, extracting 30 coefficients for each window, take the median over all windows and store it to be used for training and testing. More details on the MLPC features can be found in Section 3.4.

5.2.4 Training and Recognition Module

In this module, the user spoke each command five times. Each template command was sampled at 10 kHz, using 8 bits and a monaural channel. The average of these five templates was then taken for use in training Modular Artificial Neural Networks [9, 10] with the original five templates. The same procedure is repeated for all commands. Figure 5.4 illustrates the main operation for training and recognition. The system is designed to be in a hierarchical structure; there are three levels of modular neural networks:

The first level recognizes the top-level menu in Microsoft Word (i.e. *File, Edit, View, Insert, Format, Tools, Table, Window* and *Help*).

The second level recognizes the menu under each command in the first level (e.g. *File* → [*New, Open, Close, Save, Save as,* etc.]).

The third level is used to recognize the menu commands under the each of the second level menu commands (e.g., *Insert* → *Picture* → [*Clip Art, From file, Autoshape, Wordart* and *Chart*]).

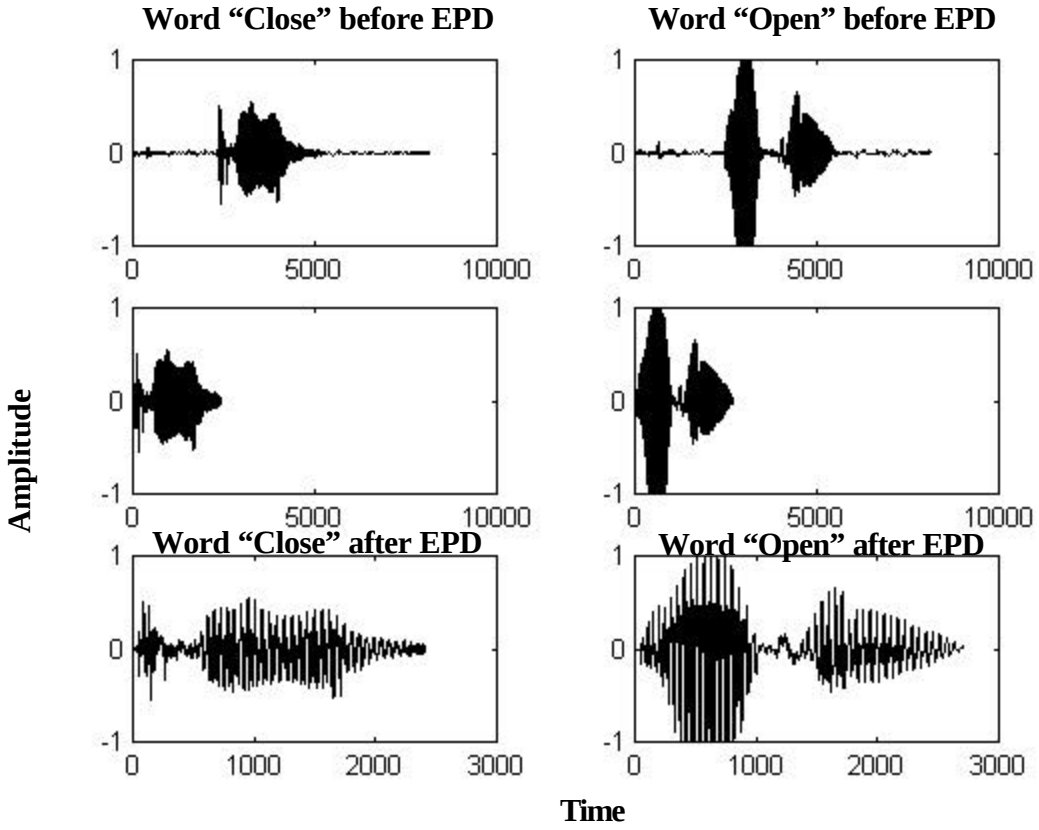

Figure 5.2. End Point Detection for words "Close" and "Open"

Each expert network receives the same input data during training. Testing the structure for each expert will be illustrated in Figure 5.5.

A specific expert is trained to recognize a single command and to reject all other commands. For example, Expert 1 is trained to provide (ideally) a unity response for all inputs (LPCs) that were derived from the word *'File'* and (ideally) a zero response for all other inputs, representing commands such as *Edit*, *View*, etc.

The back-propagation algorithm can be used effectively for the training of each expert network. Since each expert network can be trained off-line, the system becomes easily extensible. New command can be added by adding a new expert network trained with the new command. This assumes that the new word is significantly different from the current vocabulary. If this is not true, then some retraining of the 'neighboring' words may be necessary; but as mentioned previously this is typically a small subset of the entire database.

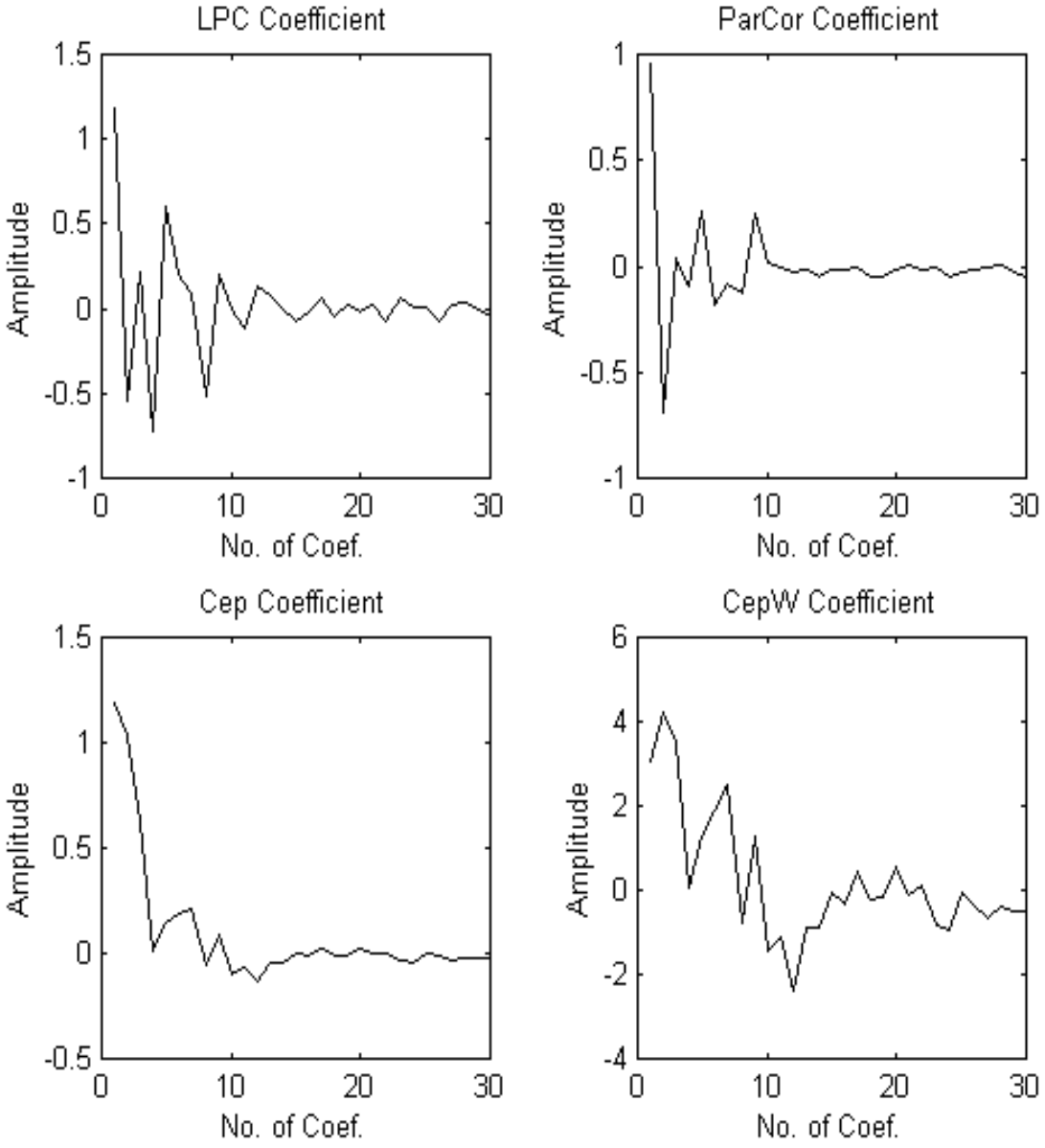

Figure 5.3. Coefficient parameters for word "Close"

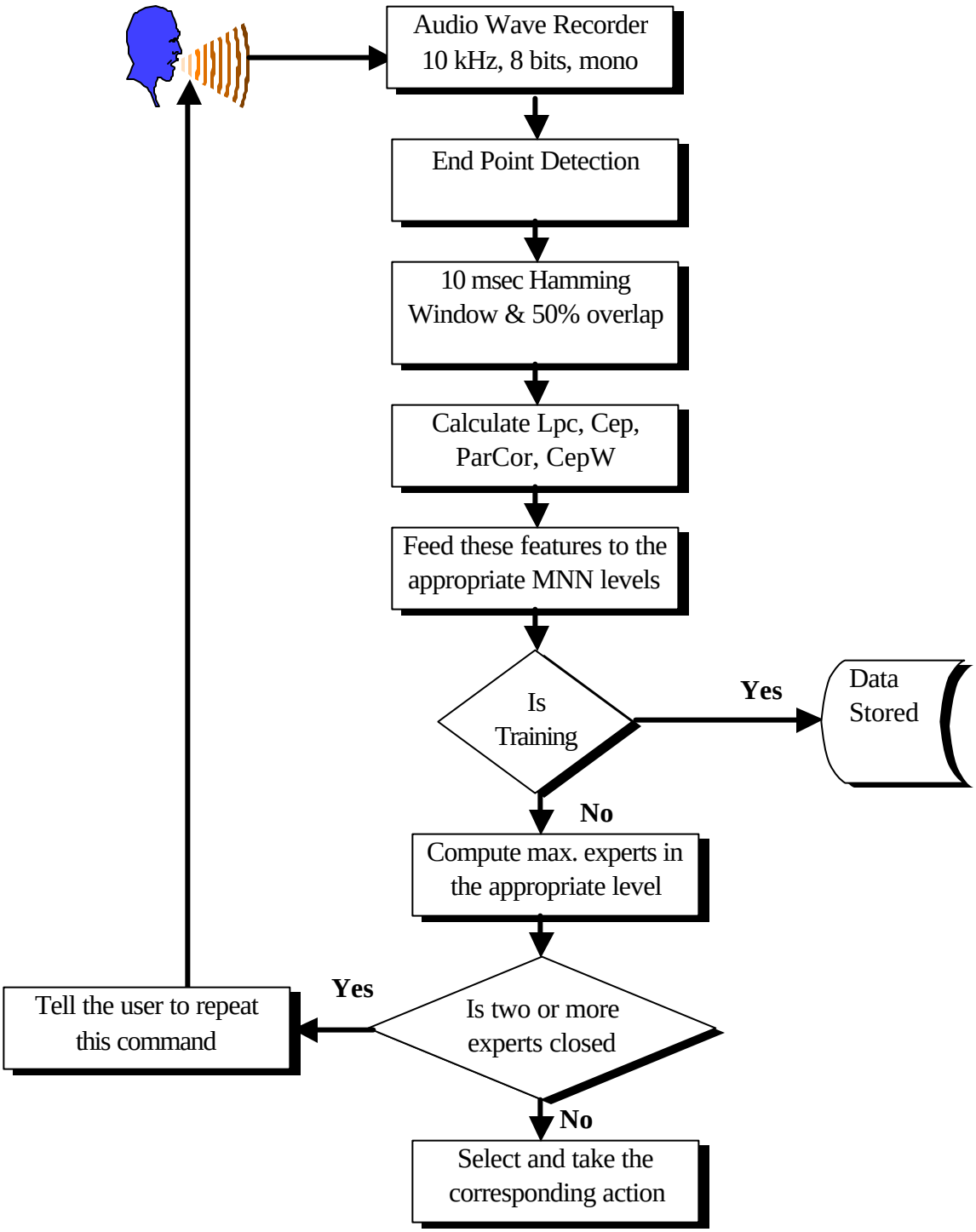

Figure 5.4. Block diagram of Human/Computer Interface System

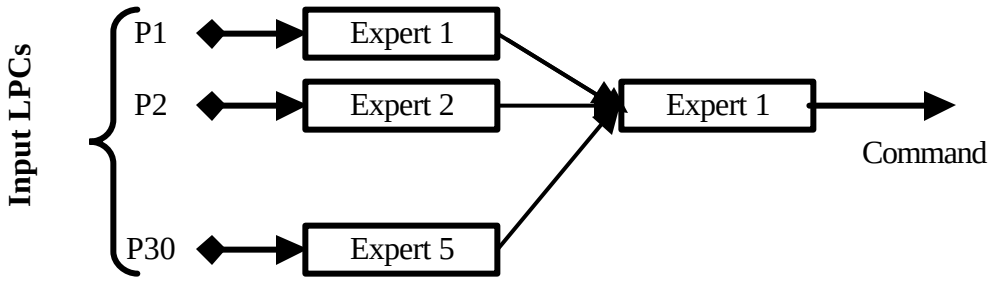

Figure 5.5. Structure for each expert

5.3 HCI System Interface

The software engine for the system is written in Matlab, while the user interface and the system control are written in Visual Basic. Visual Basic tells the Matlab engine to run the appropriate program, receive the result and take the corresponding action. The system menu contains (*File, Setup* and *Help*). The *File* menu contains *New*, *Open*, *Close*, *Control application* and *Exit*. The *New* command is used to make a new command application and to start training it, while the *Open* command is used to open an existing application. The *Close* command is used to close the current application. The *Control application* command starts to use the system to control the specific application. The *Exit* command is used to exit from the HCI system.

The setup menu contains *Environment Noise*. The *Environment Noise* command is used to record a silent one sec and to recognize the noise in the background.

The *Help* menu contains *Help* and *About*. The *Help* command describes everything about the program. The *About* command defines the system configuration and the designers names.

To control Microsoft Word, the user must click on *File→ New* command then the system will bring up two input boxes to write the application name and the path of this application (see Figure 5.6).

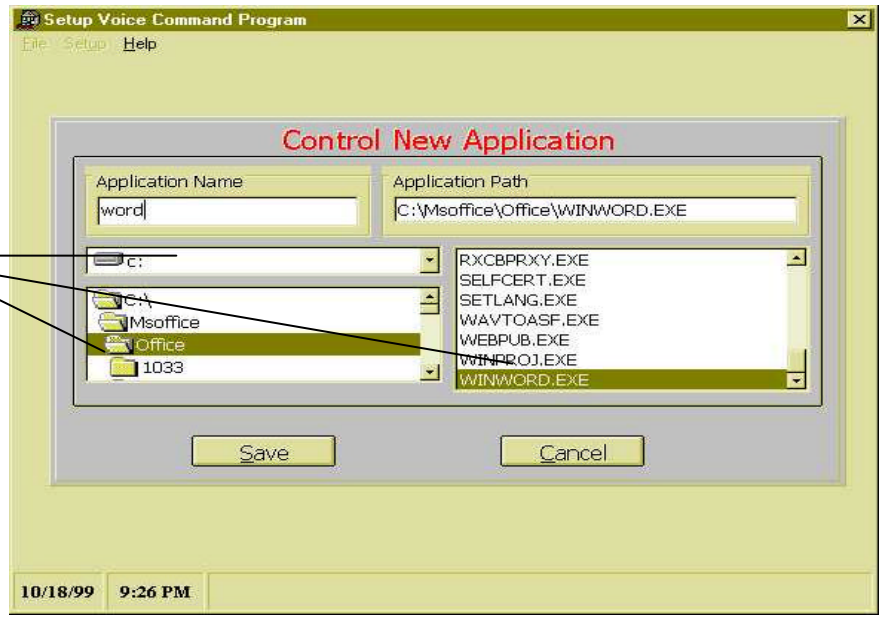

Figure 5.6. Adding a new application for HCI

After the user writes the name and the path to the application, the system brings up the interface to let the user enter the command and the corresponding action to this command. The user can then record the command five times and listen for each command to insure that the command record is correct. The interface contains two input boxes: Command Name and Command Action. The user enters both and clicks on the add button. Then user starts to record the command five times. The data in the Command Name box goes to the Command List box while the Command Action box goes to the Action box. After the user adds all the commands in the same levels and needs to start to record for the next level, he/she can point to the corresponding command in command list box, double click on it to start record commands in the next level and so on (see Figure 5.7).

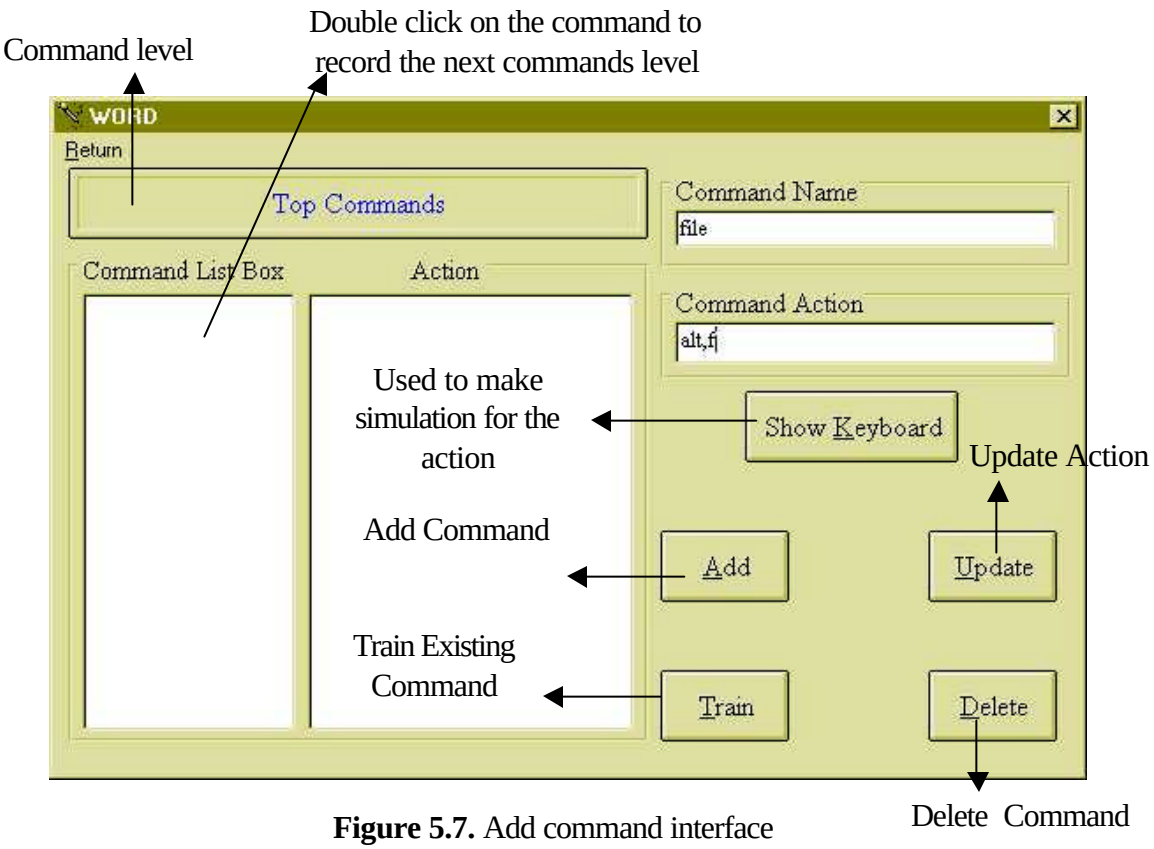

Figure 5.7. Add command interface

When the user finishes recording all commands in each level, he can start to control the application by selecting the *File→Control Application* command menu. This will run the program and put this icon  in the system tray (lower right hand corner of the screen) and open the Microsoft Word application. After the icon appears in the system tray the right button mouse can be used to see the menu (see Figure 5.8. Control application command icon in system tray).

Figure 5.8. Control application command icon in system tray

The menu contains the commands *Disable/Enable*, *Run Setup*, *About*, and *Quit* (see Figure 5.9. Control application menu).

Figure 5.9. Control application menu

The *Disable* command is used to stop recognition and the icon  will replace the previous icon in the system tray. Also, the menu *Disable* command changes to *Enable* command (see Figure 5.10. Control application *Disable* command).

Figure 5.10. Control application *Disable* command

The *Run Setup* command returns to the training program if the user needs to add or retrain any commands.

The *About* command defines the system configuration and gives the designers names, while the *Quit* command is used to close the program.

References

[1]. P. Corsi., *"Speaker Recognition: A Survey,"* Automatic Speech Analysis and Recognition, pp. 277-308, 1982.

[2]. E. S. Dermatas, N. D. Fakotakis, and G. K. Kokkinakis, *"Fast Endpoint Detection Algorithm For Isolated Word Recognition in Office Environment,"* Proc. Intl. Conf. Acoust., Speech & Signal Proc., pp. 733-736, July 1991.

[3]. J. D. Markel and A. H. Gray, Jr., *"Linear Prediction of Speech,"* SpringVerlag, New York, 1976.

[4]. E. M. Hofstetter, *"An Introduction to the Mathematics of Linear Predictive Filtering as Applied to Speech Analysis and Synthesis,"* Tech. Note 1973-36, MIT Lincoln Labs, Jul 1973.

[5]. Le Roux. J., and C. Guegen, *"A Fixed point computation of partial correlation coefficients in linear prediction,"* Proc. ICASSP-77, IEEE Press, New York, pp. 742-743, 1977.

[6]. J. A. Freeman and D. M. Skapura, *"Neural Networks: Algorithms, Applications, and Programming Techniques,"* Addison-Wesly Co., Reading Massachusetts, 1991.

[7]. B. Kosko, *"Neural Networks for Signal Processing,"* Englewood Cliffs, N. J.: Prentic-hall, Inc., 1992.

[8]. S. Y. Kung, *"Digital Neural Networks,"* PTR Prentice Hall, 1993.

[9]. S. Haykin, *"Neural Networks: A Comprehensive Foundation,"* IEEE Press, 1994, Chapter 12.

[10]. M. I. Jordan and R. A. Jacobs, *"Hierarchical mixtures of experts and the EM algorithm,"* Neural Computation, 6:181-214, 1994.

Chapter 6

Results, Conclusions and Future Work

The Human/Computer Interface (HCI) system previously described exhibits a 92.7% overall recognition accuracy based on 3000 (150x20 templates separate from the training set) Microsoft Word spoken commands using a 10 kHz sampling frequency. Moreover, there are 111 out of 219 command errors that can be corrected because the next highest weight expert selects those 111 commands. If these situations can be solved, a 96.4% overall recognition accuracy can be achieved.

TABLE 6.1. Coefficients Comparison

Coefficients	PMNN Accuracy 3000 (150x20 templates Commands)
Median Linear Predictive Coding (MLpc)	87.2%
Median Cepstral Coefficients (MCep)	92.7%
Median Partial Coefficient (MParCor)	72.8%
Median Weighted Cepstral Coefficient (MCepW)	85.3%

Based on this research work, the best recognition results come from the median cepstral coefficients features, 92.7%. While the median linear predictive coding features bring 87.2% and median weighted cepstral coefficients features have 85.3%. The worst recognition results come from partial coefficient code features, 72.8%. See table 6.1.

The following key results can be noted:

- A special structure of modular artificial neural network (MANN) has been shown to outperform global monolithic neural networks in classification and distinguishing between multiple verbal command patterns.
- The modular system was able to perform its task with a very short training time (relative to the non-modular system).
- The system can accommodate and classify a new command in a very convenient way.
- The presented methodology removes the limitation of extending the system at the expense of large retraining times.
- A novel approach called a Pyramidal Modular Neural Network (PMNN), is used to design the Human/Computer Interface (HCI) system.
- Approximately 2% error with a PMNN can be achieved from the results shown. If there exist two or more experts with similar classification errors, then the results are rejected, the user is asked to repeat the command.
- A novel features set of Median Linear Predictive Coefficients (MLPC) was designed to minimize the processing time, storage capacity and complexity. This technique can reduce the processing time by approximately 75% over more conventional feature sets.

In summary, this study was successful in the design of a complete system for a (voice) human/computer interface. During the process of designing the system, several technical contributions were made that will impact the speech processing community. Among these are:

- The use of the PMNN structure.
- Generation and use of MLPC features.
- A hybrid end-point detection algorithm.

Future Work:

At this point the system works well in the mode of a speaker dependent, medium vocabulary, system. Future refinements could easily extend the speaker dependent requirement to a set of multiple users, as in a "network" of users.

In other system applications there may be a need to have a more extensive vocabulary. The modular technique used here is adequate to design a fairly large vocabulary system, depending on the computation available, but this has not been tried. It may be that a suitable need will present itself at a future date. The details of using median features can be further studied to consider grouping and/or combining two, or more types of features (Lpc, Cep, WCep and ParCor) to improve the accuracy. Improved accuracy alleviates having to ask the user to repeat the command. Another improvement on the system that is not currently in place is an error check for appropriate commands. For example, if the user issues a subcommand that is not in the subdirectory of a higher level command, there could be a response to the user that only certain commands are appropriate.

Vita

Born in Cairo, Egypt, Moatassem M. Abdallah graduated from San George High School in 1983. He then attended and later graduated from Military Technical College with a B.S. degree in Electrical and Computer Engineering in 1988. Upon graduation, he joined Military Technical College as a lecturer assistant for eight years. He enrolled in the electrical and computer engineering graduate program at Military Technical College where he later completed the M.S. degree. He came to the United States in 1996 and enrolled in the Electrical and Computer Engineering Ph.D. graduate program at Virginia Tech. He is a member of IEEE, and ISA. He enjoys many activities, including soccer, squash, tennis, billiards, and swimming.