SPEAKER DEPENDANT ISOLATED HINDI WORD RECOGNITION

A DISSERTATION

Submitted in partial fulfilment of the requirements for the award of the degree of

MASTER OF TECHNOLOGY

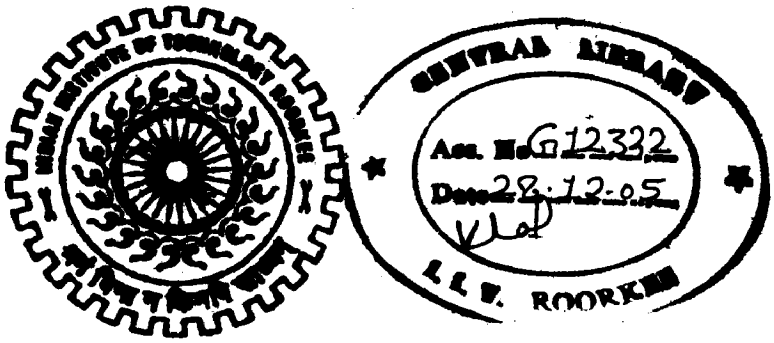
in

ELECTRICAL ENGINEERING

(With Specialization in Measurement & Instrumentation)

By

SHANKAR BABU.D



DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE-247 667 (INDIA)

JUNE, 2005

CANDIDATE'S DECLARATION

I hereby declare that the work that is being presented in this dissertation report entitled "SPEAKER DEPENDENT ISOLATED HINDI WORD RECOGNITION" submitted in partial fulfillment of the requirements for the award of the degree of **Master Of Technology** with specialization in **Measurement & Instrumentation,** to the **Department Of Electrical Engineering, Indian Institute Of Technology, Roorkee**, is an authentic record of my own work carried out, under the guidance of **Dr. R.S Anand**, Asst. Professor, Department of Electrical Engineering.

The matter embodied in this dissertation report has not been submitted by me for the Award of any other degree or diploma.

Date: June 2005
Place: Roorkee

(SHANKAR BABU .D)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

(Dr. R .S. ANAND)
Asst. Professor,
Department of Electrical Engg,
Indian Institute of Technology,
ROORKEE – 247 667,
INDIA.

ACKNOWLEDGEMENT

I wish to express my deep sense of gratitude and sincere thanks to my beloved guide **Dr.** R.S Anand, Asst.Professor. Department of Electrical Engineering, IIT Roorkee, for being helpful and a great source of inspiration. His keen interest and constant encouragement gave me the confidence to complete my work. I wish to extend my sincere thanks for his excellent guidance and suggestions for the successful completion of my project work.

My heartfelt gratitude and indebt ness goes to all the teachers of Measurements and Instrumentation group who, with their encouraging and caring words, constructive criticism and suggestions have contributed directly or indirectly in a significant way towards completion of this dissertation.

I convey my deep sense of gratitude to the Head of Electrical Engineering Department, who directly or indirectly helped me during the work.

I am thankful to Mr. Srikant Reddy, M.Tech student in SEOR group of Electrical Engineering Department for his constant source of encouragement for successful completion of this work. He always boosts my morale

Special thanks to my friends whose support and encouragement has been a constant source of guidance to me.

It is difficult for me to express my gratitude to my parents for their affection and encouragement. Last but not the least; I am indebted to all my classmates from Measurements & Instrumentation group for taking interest in discussing my problems and encouraging me.

(SHANKAR BABU .D)

ABSTRACT

Although speech recognition products are already available in the market at present, their development is mainly based on statistical techniques which work under very specific assumptions. This thesis examines how artificial neural networks can benefit a speaker dependent isolated speech recognition system. Currently, most speech recognition systems are based on hidden Markov models (HMMs), a statistical framework that supports both acoustic and temporal modeling. Despite their state-of-the-art performance, HMMs make a number of suboptimal modeling assumptions that limit their potential effectiveness. Neural networks avoid many of these assumptions, while they can also learn complex functions, generalize effectively, tolerate noise, and support parallelism, while neural networks can readily be applied to acoustic modeling.

Neural Network has several theoretical advantages over a pure HMM system, including better acoustic modeling accuracy, better context sensitivity, more natural discrimination, and a more economical use of parameters. These advantages are confirmed experimentally by a NN that we developed, based on speaker dependent isolated Hindi words on the Resource Management database.

Speech recognition involves recording the input speech signal, extracting the key features of the speech, converting the features into codes and finally classification of the codes. A speech recognizer system comprised of two distinct blocks, a Feature Extractor and a Recognizer. The Feature Extractor block uses a Mel-frequency cepstral analysis which translates the incoming speech into a feature vectors and recognizer block uses neural network. In the course of developing this system, we explored two different ways to use neural networks for audio modeling: prediction and classification. We found that predictive networks yield poor results because of a lack of discrimination, but classification networks gave excellent results. Finally, this thesis reports how we optimized the accuracy of our system with many natural techniques, such as expanding the input window size, normalizing the inputs, increasing the number of hidden units, converting the network's output activations to log likelihoods, optimizing the learning rate schedule by automatic search, backpropagating error from word level outputs, and using gender dependent networks.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter - 1

INTRODUCTION

1.1 Introduction

Speech recognition is a process used to recognize speech uttered by a speaker and has been in the field of research for more than five decades since 1950s [7]. Voice communication is the most effective mode of communication used by humans. In this world of communication, humans interact with each other through speech and even the training of animals in the zoo is also done by using speech. As we can see, voice is the most natural mode of control as it is fast, hands free and eyes free. Voice of a person is very unique just like fingerprint of human beings.

But the question is why does the speech recognition problem attract researchers and funding? Speech recognition is an important and emerging technology with great potential. The beauty of speech recognition lies in its simplicity. This simplicity together with the ease of operating a device using speech has lots of advantages. It can be used in many applications like, security devices, household appliances, cellular phones, ATM machines and computers.

If an efficient speech recognition machine is enhanced by natural language systems and speech producing techniques, it would be possible to produce computational applications that do not require a keyboard and a screen. This would allow incredible miniaturization of known systems facilitating the creation of small intelligent devices that can interact with a user through the use of speech [2].

Speech recognition involves recording the input speech signal, extracting the key features of the speech, converting the features into codes and finally classification of the codes. The most successful techniques used in the past include Dynamic Time Warping and Hidden Markov Model. These two methods are, however, very complex and take up lots of memory space. The complexity of the speech recognition process is due to the fact that a given utterance can be represented by an infinite number of time-frequency patterns.

1.2 Speech recognition problems

The human vocal tract and articulators are biological organs with a nonlinear property, whose operation is not just under conscious control but also affected by factors ranging from gender to upbringing to emotional state. As a result, vocalizations can vary widely in terms of their accent, pronunciation, articulation, roughness, nasality, pitch, volume, and speed; moreover, during transmission, our irregular speech patterns can be further distorted by background noise and echoes, as well as electrical characteristics (if telephones or other electronic equipment are used). All these sources of variability make speech recognition, even more than speech generation, a very complex problem.

Phonemes and written words follow cultural conventions. The speech recognizer does not create its own classifications and has to follow the cultural rules that define the target language. This implies that a speech recognizer must be taught to follow those cultural conventions. The speech recognizer cannot fully self organize. It has to be raised in a society!

The complexity of the speech recognition problem is defined by the following aspects [3]:

- Vocabulary size, *i.e.* the bigger the vocabulary the more difficult the task is. This is explained by the appearance of similar words that start to generate recognition conflicts, *i.e.* 'WHOLE' and 'HOLE'.
- Grammar complexity.
- Segmented or continuous speech, *i.e.* segmented streams of speech is easier to recognize than continuous ones. In the latter, words are affected by the coarticulation phenomenon.
- Number of speakers, *i.e.* the greater the numbers of speakers whose voice needs to be recognized, the more difficult the problem is.
- Environmental noise.

A speech recognition system, sampling a stream of speech at 8 kHz with 8 bit precision, receives a stream of information at 8 Kbits per second as input. After processing this stream, written words come out at a rate of more or less 60 bits per second. This implies an enormous reduction in the amount of information while

preserving almost all of the relevant information. A speech recognizer has to be very efficient in order to achieve this compression rate (more than 1000:1).

Speech recognizers are normally divided into two stages, as shown by the schematic diagram in Fig.1.1. The Feature Extractor (FE) block shown in this figure generates a sequence of feature vectors, a trajectory in some feature space that represents the input speech signal. The FE block is the one designed to use the human vocal tract knowledge to compress the information contained by the utterance. Since it is based on *a priori* knowledge that is always true, it does not change with time. The next stage, the Recognizer, performs the recognition and generates the correct output word. Since this stage uses information about the specific ways a user produce utterances, it must adapt to the user.

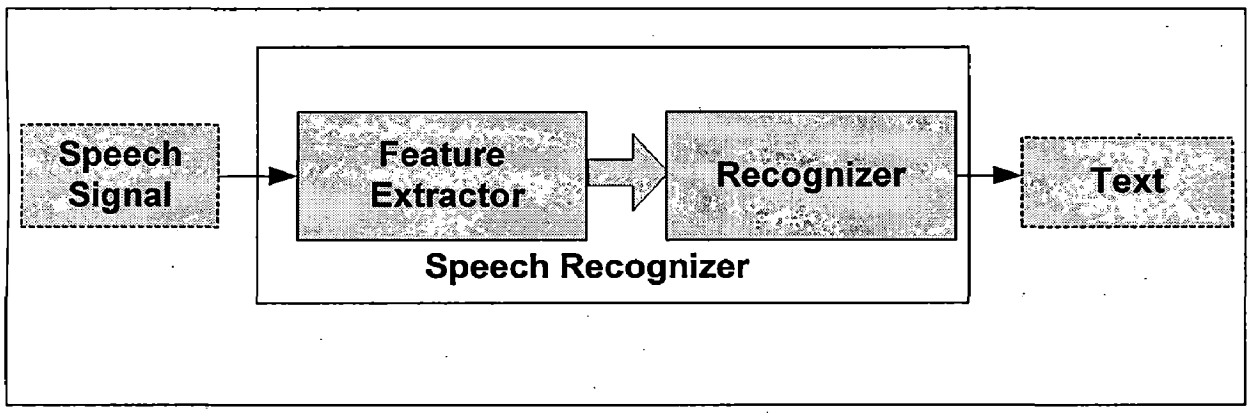


Fig.1.1 Basic building blocks of a speech recognizer.

The FE block can be modeled after the stages evidenced in the human biology and development. This is a block that transforms the incoming sound into an internal representation such that it is possible to reconstruct the original signal from it. This stage can be modeled after the hearing organs, which first transducers the incoming air pressure waves into a fluid pressure wave and then converts them into a specific neuronal firing pattern. After the first stage, comes the one that analyzes the incoming information and classifies it into the phonemes of the corresponding language. This Recognizer block is modeled after the functionality acquired by a child during his first six months of existence, where he adapts his hearing organs to specially recognize the voice of his parents.

Once the FE block completes its work, its output is classified by the Recognizer module. It integrates the sequences of phonemes into words. This module sees the world as if it where only composed of words and classifies each of the incoming trajectories into one word of a specific vocabulary.

The process of correlating utterances to their symbolic expressions, translating spoken language into written language, is called speech recognition. It is important to understand that it is not the same problem as speech understanding, a much broader and powerful concept that involves giving meaning to the received information.

1.3 A Brief history of speech recognition research

Researchers have worked in automatic speech recognition for almost four decades. The earliest attempts were made in the 50's. In 1952, at Bell Laboratories, Davis, Biddulph and Balashek built a system for isolated digit recognition for a single speaker [3]. In 1956, at RCA Laboratories, Olson and Belar developed a system designed to recognize 10 distinct syllables of a single speaker [3]. In 1959, at University College in England, Fried and Dener demostrated a system designed to recognize four vowels and nine consonants[3]. The same year, at MIT's Lincoln Laboratories, Forgie and Forgie built a system to recognize 10 vowels in a speaker independent manner. All of these systems used spectral information to extract voice features [3].

In the 60's, Japanese laboratories appeared in the arena. Suzuki and Nakata, from the Radio Research Laboratories in Tokyo, developed a hardware vowel recognizer in 1961[3]. Sakai and Doshita, from Kyoto University, presented a phoneme recognizer in 1962[3]. Nagata and coworkers, from NEC Laboratories, presented a digit recognizer in 1963. Meanwhile, in the late 60's, at RCA Laboratories, Martin and his colleagues worked on the non-uniformity of time scales in speech events [3]. In the Soviet Union, Vintsyuk proposed dynamic programming methods for time aligning a pair of speech utterances. This work remained unknown in the West until the early 80's. A final achievement of the 60's was the pioneering research of Reddy in continuous speech recognition by dynamic tracking of phonemes [3]. This research spawned the speech recognition program at Carnegie Mellon University, which, to this day, remains a world leader in continuous speech recognition systems.

In the 70's, researchers achieved a number of significant milestones, mainly focusing on isolated word recognition. This effort made isolated word recognition a viable and usable technology. Itakura's research in USA showed how to use linear predictive coding in speech recognition tasks [3]. Sakoe and Chiba in Japan showed how to apply dynamic programming [3]. Velichko and Zagoruyko in Russia helped in the use of pattern recognition techniques in speech understanding [3]. Important also were IBM contributions to the area of large vocabulary recognition [3]. Also, researchers at ATT Bell Labs began a series of experiments aimed at making speech recognition systems truly speaker independent [3].

In the 80's, the topic was connected word recognition. Speech recognition research was characterized by a shift in technology from template-based approaches to statistical modeling methods, especially Hidden Markov Models (HMM). Thanks to the widespread publication of the theory and methods of this technique in the mid 80's, the approach of employing HMMs has now become widely applied in virtually every speech recognition laboratory of the world [3]. Another idea that appeared in the arena was the use of neural nets in speech recognition problems. The impetus given by DARPA to solve the large vocabulary, continuous speech recognition problem for defense applications was decisive in terms of increasing the research in the area [3].

Today's research focuses on a broader definition of speech recognition. It is not only concerned with recognizing the word content but also prosody and personal signature. It also recognizes that other languages are used together with speech, taking a multimodal approach that also tries to extract information from gestures and facial expressions.

Despite all of the advances in the speech recognition area, the problem is far from being completely solved. A number of excellent commercial products, which are getting closer and closer to the final goal, are currently sold in the commercial market. Products that recognize the voice of a person within the scope of a credit card phone system, command recognizers that permit voice control of different types of machines, "electronic typewriters" that can recognize continuous voice and manage several tens of thousands word vocabularies, and so on. However, although these applications may seem impressive, they are still computationally intensive, and in order to make their usage

widespread more efficient algorithms must be developed. Summing up, there is still room for a lot of improvement and of course, research [3].

1.4 Applications

1. Railways, Air Bus, answering enquiries about the reservations, schedules. The PNR number is spoken by the user as continuous speech is recognized by the application software and is presented to the database system, which retrieves out the current status of reservation.
2. Voice activated Windows command controls. This application is an example of isolated digit recognition.
3. Home speech based telephone dialing. Activating appliances by voice.
4. Factory, punching in and out timing by voice, vending machines.
5. Departmental Stores, giving information about services, receiving orders.
6. General applications, Voice controlled wheel chair, commands to computer, data entry etc.

1.5 STATE OF THE ART

An overview of some of the popular methods for speech recognition is presented in this section following the schematic diagram that outlines the constituent blocks as in Fig. 1-1. The functionality of the individual blocks is also described in order to precisely state the contributions that stem from the work outlined in this thesis.

1.5.1 Feature extractors

The objective of the FE block is to transform an input in the signal space to an output in a feature space to achieve some desired criteria. The FE block is usually a static module that once designed will not appreciably change. The criteria to be used depend on the problem to be solved. For example, if a noisy signal is received, the objective is to produce a signal with less noise.

The FE block used in speech recognition should aim towards reducing the complexity of the problem before later stages start to work with the data. Furthermore, existing relevant relationships between sequences of points in the input space have to be preserved in the sequence of points in the output space. The rate at which points in the signal space are processed by the FE block does not have to be the same rate at which

points in the feature space are produced. This implies that time in the output feature space could occur at a different rate than time in the input signal space.

A priori knowledge concerning which are the relevant features that should be used in a speech recognition problem comes from very different sources. Results from biological facts, such as the EIH model (which is based on the inner workings of the human hearing system [3]), descriptive methods (like banks of pass band filters [3]), data reduction techniques (such as PCA [4]), speech coding techniques (such as LPC Cepstrum [5]), and neural networks (such as SOM [6]), have been combined and utilized in the design of speech recognition feature extractors. An important result obtained by Jankowski et al [4], which summarizes all of the above mentioned *a priori* knowledge, suggests that under relatively low noise conditions all systems behave in similar ways when tested with the same classifier. This explains why the speech recognition community has adopted the LPC Cepstrum, which is very efficient in terms of computational requirements, as the method of choice.

1.5.2 Recognizers

Recognizers deal with speech variability and account for learning the relationship between specific utterances and the corresponding word or words. There are several types of classifiers but only two are mentioned: the Template approach and the approach that employs Hidden Markov Models.

The first one, the Template Approach, is described by the following steps:

1. First, templates for each class to be recognized are defined. There are several methods for forming the templates. One of these consists in randomly collecting a fixed amount of examples for each class in order to capture the acoustic variability of that specific class. The template of a class is defined as the set of collected examples for that class.
2. Select a method to deal with utterance time warpings. A commonly used technique is a procedure called Dynamic Time Warping [7]. It consists of an adaptation of dynamic programming optimization procedures for time warped utterances.
3. Select some distance measure for comparing an unknown example against the template of each class after time warping correction. This distance measure can be

based on geometrical measures, like Euclidean distance, or perceptual ones, like Mel or Bark scales [3].

4. Compare unknown patterns against the collected templates using the selected time warping correction and the chosen distance measurement. The unknown pattern is classified according to the values obtained from the distance measurements.

One drawback of the template procedure is that nothing guarantees that the chosen examples would really capture the class variability. The main problem with this method is that several utterances must be collected in order to capture the word's variability. For small vocabularies this may not be a problem, but for large ones it is something unthinkable: no user would willingly utter thousands and thousands of examples.

Another approach is based on HMM, and solves the temporal and acoustic problems at once using statistical considerations. It is described by the following steps [8]:

1. Like the Template Approach, something that stores the knowledge about the possible variations of a class must be defined. The difference is that in this case it is an HMM instead of a template. Consider a system that may be described at any time by means of a set of M distinct states, such as a word which can be represented by a collection of phonemes. At regularly spaced, discrete times, the system undergoes a change of state according to a set of probabilities associated with each state. Let us assume that each time the system moves to another state, an output selected from a set of N distinct outputs is produced. The outputs are selected according to a probability mass function. The states are hidden from the outside world, which only observes the outputs. In other words, a HMM is an embedded stochastic process with an underlying stochastic process that is not directly observable but can be observed only through another set of stochastic processes that produces the sequence of observations. In order to obtain the HMM that represents a class, the number of hidden states, state transition probabilities, and output generation probabilities must be defined. Usually, the number of states is defined by trial and error. The probabilities are instead obtained through iterative methods that work with sets of training examples. An example of an HMM is shown in Fig.1.2, where the solid circles represent the possible states, and the arrows represent the transitions between

states as an utterance progresses. In the most basic level, a HMM is used to model a word and each state represents a phoneme. The number of states used to model a word depends on the number of phonemes needed to describe that word.

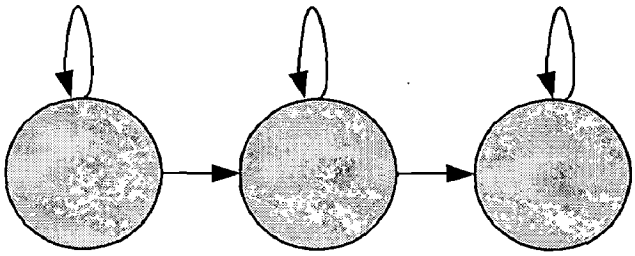


Fig.1.2 Hidden markov model example.

2. Once the HMMs have been determined, whenever an unknown utterance is presented to the system, each model evaluates the probability of producing the output sequence associated to that utterance. The word associated to the model with the highest probability is then assigned to the unknown utterance.

The HMM method does not have some of the drawbacks exhibited by the Template Approach. Because it models the utterances as stochastic processes, it can capture the variability within a class of words. In terms of scalability, it offers much more flexibility. Instead of defining one HMM for each word, a hierarchy of HMMs is built, where the bottommost layer of HMMs models phonemes, a second layer which models short sequences of connected phonemes and uses the output of the first layer as input, and a final layer which models each word and uses the output of the second layer as input. This hierarchical approach constitutes the most successful approach ever devised for speech recognition.

The limitations of the HMMs are that they rely on certain *a priori* assumptions that do not necessarily hold in a speech recognition problem [9] [10]:

- The first order assumption, that all probabilities depend solely on the current state, is false for speech applications. This is the reason why HMMs have strong problems modeling coarticulations, where utterances are in fact strongly affected by recent state history.
- The independence assumption, that there is no correlation between adjacent time frames, is also false. At a certain instant of time, both words can be described by

the same state but the only thing that differentiates them is their behavior before that state, which is something completely ignored by the HMM model.

A characteristic common to all speech recognizers is that they are composed of layered hierarchies of sub-recognizers. Normally, the first layer is composed of a set of sub-recognizers whose output is integrated by the following layer of sub-recognizers, and so on, until the desired output is obtained. An example is the architecture used in a HMM based recognizer: the first layer is composed of a set of HMMs, each of them specialized in recognizing phonemes. The second layer, again composed of a set of HMMs, uses as input the output of the first layer and specializes in recognizing collections of phonemes. The third layer, based on HMMs too, uses the output of the second layer to recognize words. Finally, the fourth layer integrates the words into sentences using built-in knowledge about the grammar of the target language.

1.6 Organization of the thesis

The dissertation has been composed of eight chapters. The details of the contents of each chapter are given below.

In **chapter 1,** introduction of speech recognition and its brief history and applications.

In **chapter 2,** basics of spoken and written languages have been presented.

In **chapter 3,** basics of speech signal, how the speech is produced and perceived by humans being is discussed along with different representations of speech are discussed.

In **chapter 4,** how the speech is transformed into feature vectors along with the steps that are required to achieve that are given. The need to find out the start and end point of speech, so that it can be effectively transformed into speech vectors, has been presented.

In **chapter 5,** the stochastic approach i.e Neural Network to speech recognition is introduced and the technique to apply them on isolated word speech recognition is discussed.

In **chapter 6,** the scheme for the proposed work has been given. The various steps that are involved and all GUI based menu driven programs have been listed.

In **chapter 7,** the feature vector set of the two words have been presented. Results have been obtained for the given vocabulary size and the reduced one. The results of both the cases (the original vocabulary and the reduced one) have been compared along with the different speakers and a discussion of the result presented.

Lastly, In **chapter 8,** conclusion of present work and some suggestion for future work have been given.

Chapter-2

LANGUAGES

2.1 Introduction

Language is what allows us to communicate, *i.e.* to convey information from one person to another. It is this ability that permits groups of individuals to transform into an information sharing community with formidable powers. Thanks to the existence of language, anyone can benefit from the knowledge of others, even if this knowledge was acquired in a different place or at different times.

Language is a complex phenomenon, and it can appear in very different forms, such as body positions, facial expressions, spoken languages, etc. Not only that, these different types can be used separately or in combination, rendering really complex and powerful methods to communicate information. For example, dance is a method of communication that emphasizes the use of body position and facial expressions. Like dancing, a face to face conversation relies on body gestures and facial expressions, but it also relies on the use of sounds and spoken language. Summing up, when we are speaking about a language, we should not restrict ourselves to think about it as a collection of sounds ordered by some grammar. Any language is far more than that.

Modern language theories state that what we call languages are surface manifestations, or exterior representations, of a common inner core, not directly seen from the outside, where thought processes occur (Fig.2.1). In other words, they state that we do not think using what we would normally call a language, but internal mental representations that condense all sorts of cognitive processes [11]. Under this point of view, the different languages that a person uses are nothing more than a set of different interfaces between the processing results of that inner core and the community.

Even though under normal conditions we normally use all types of languages at the same time, it can be said that spoken language is the one that conveys the most of the information, while the others normally enhance or back up the meanings conveyed by spoken language.

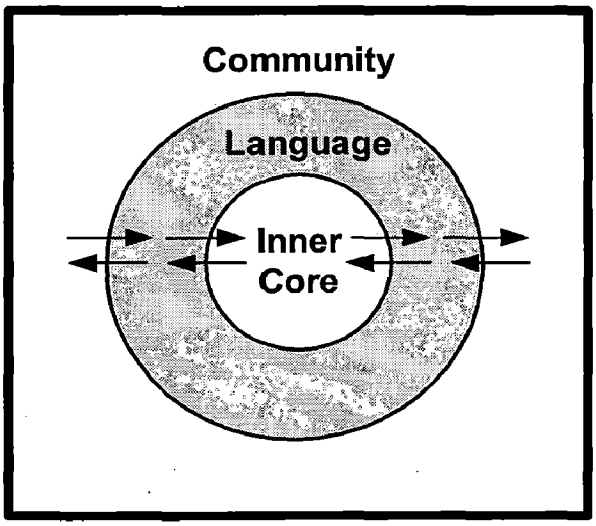


Fig.2.1. Relationship between languages and inner core thought processes.

Language is a complex, specialized skill, which develops in the child spontaneously, without conscious effort or formal instruction, is deployed without awareness of its underlying logic, is qualitatively the same in every individual, and is distinct from more general abilities to process information or behave intelligently." Current research in cognitive sciences states that language corresponds more to an instinct [11] than to a socially developed skill, implying that is something inherently human. More than this, it seems to be inherently biological: the capacity of speaking a language is a consequence of human biology, while its manifestation, *i.e.* the actual way we speak, a consequence of the culture where persons live.

Even more, it seems that only humans possess this skill. All studies done in animals, even the more advanced primates, indicate orders of magnitude of difference between human languages and the animal ones. So big is this difference that if language is defined as something similar to human languages, it can be safely considered that animals do not have the ability to communicate using a language.

2.2 Spoken language

The basic building block of any language is a set of sounds named phonemes. As an example, a condensed list of Hindi phonemes, along with their IPA symbol and ASCII representation, is given in Table 2.1 [12].

Table 2.1: The Hindi alphabet. For a phoneme in a cell, the corresponding IPA symbol and the ASCII representation used in the database are shown in the second and third row of the cell respectively.

अ	आ	इ	ई	उ	ऊ	ए	ऐ	ओ	औ
ə	*aː*	*i*	*iː*	*u*	*uː*	*ɛ*	*ɛː*	*o*	*oː*
ə	A	i	I	u	U	e	E	o	O

क	ख	ग	घ	ङ
k	*kʰ*	*g*	*gʰ*	ŋ
k	kh	g	gh	gñ
च	छ	ज	झ	ञ
ʧ	*ʧʰ*	*ʤ*	*ʤʰ*	ɲ
c	ch	j	jh	jñ
ट	ठ	ड	ढ	ण
ʈ	ʈʰ	ɖ	ɖʰ	ɳ
T	Th	D	Dh	N
त	थ	द	ध	न
t	*tʰ*	*d*	*dʰ*	*n*
t	th	d	dh	n
प	फ	ब	भ	म
p	*pʰ*	*b*	*bʰ*	*m*
p	ph	b	bh	m

य	र	ल	व	श	ष	स	ह
j	*r*	*l*	ʋ	ʃ	ʂ	*s*	*ɦ*
y	r	l	w	s~	S	s	h

The Hindi alphabet is shown in Table 2.1. It has three sections: the first section lists the vowels, the second section lists phonemes whose production involves complete closure of oral tract (plosives, affricates and nasals); the third section lists the semivowels and fricatives. Each cell in the figure represents a phoneme and has 3 rows: the first row is the Devnagri script, the second corresponding to IPA symbol, and the third the roman script used to label the phoneme in a spoken Hindi sentence.

The phonemes of Table 2.1 comprise all the building blocks needed to produce what is called Devnagri script (Hindi). Other versions of Hindi use slightly different sets of phonemes, but similar enough to allow understanding by any Hindi speaker. These differences explain the different accents existing in Indian Hindi. As it is in Hindi, all languages follow this structure: they are built upon a basic set of phonemes. It is interesting to note that the number of phonemes in a set of basic phonemes never exceeds seventy. This fact might indicate a limit on the number of sounds a human can produce or distinguish in order to communicate efficiently.

The second level of building blocks of a spoken language consists of the words. These building blocks are built from sequentially concatenated phonemes extracted from the basic set of phonemes of the language. An utterance of the word 'ZERO' is shown in Fig.2.2. It can be seen in the figure that there are roughly four distinct zones in the utterance, each of them directly related to one of the four phonemes that comprise the utterance of that word.

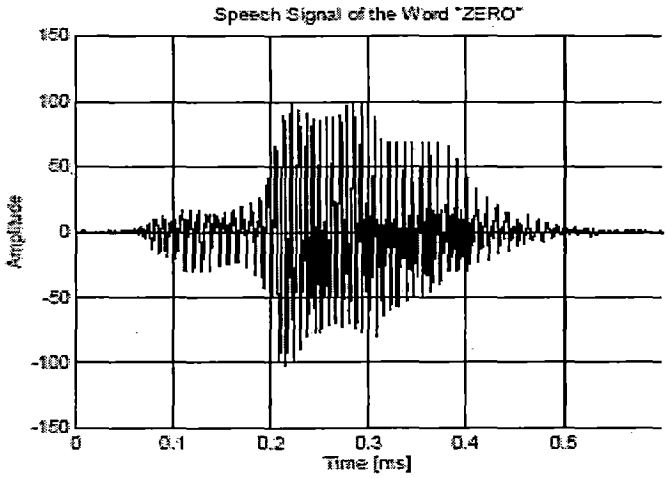


Fig.2.2. Speech signal for the word 'ZERO'.

All the words in a language constitute a set called vocabulary. It is generally agreed among experts that a normal person understands and uses an average of 60,000 words [11] of his native language.

The enormous number of possible utterances within a given language is explained by all possible combinations of phonemes, words, phrases and sentences, degree of interaction with other people, possible manifestations of the stresses, rhythms, and intonations, the continuous effort to optimize performed by the speech organs, and noise. It is important to note that there is a set of possible utterances for a given word. It is not necessarily valid to model all of them as deviations from a "correct" one. On the contrary, each of them is equally valid. Within the language context, utterance variability does not necessarily mean utterance deviation. Utterance variability cannot be explained as an ideal template distorted by some error. The concept of error is only useful to explain the always present noise.

2.3 Learning spoken language

Spoken language acquisition is an unconscious process. During their first year of existence, children learn the basic set of phonemes. During the first two months children produce all kinds of sounds to express themselves. After that, they start playing with them and babbling in syllables like 'BA-BA-BA'. When they are 10 months old, they tune their phoneme hearing capacity such that they start to react more frequently to the sounds spoken by their parents. This transition is done before they understand words, implying that they learn to identify phonemes before they start understanding words.

At the end of the first year they start to understand syllable sequences that resemble speech. By listening to their babbling they receive a feedback that helps them to finish the development of their vocal tract and to learn how to control it.

Around the first year they start understanding words and produce words. The one word stage can last from two months to a year. Around the world, scientists have proven that the content of this vocabulary is similar: half of them are used to designate objects ('JUICE', 'EYE', 'DIAPER', 'CAR', 'DOLL', 'BOTTLE', 'DOG', etc.), the rest are for basic actions ('EAT', 'OPEN', 'UP', etc.), modifiers ('MORE', 'HOT', 'DIRTY', etc.), and social interaction routines ('LOOK AT THAT', 'WHAT IS THAT', 'HI', 'BYEBYE', etc.).

Language starts to develop at an astonishing rate at eighteen months. They already understand sentences according to their grammar. They start to produce two and three word utterances and sentences that are correctly ordered despite some missing words.

After late twos, the language employed by children develops so fast that in the words of Pinker [11]: "it overwhelms the researchers who study it, and no one has worked out the exact sequence." Sentence length and grammar complexity increases exponentially.

Why does it take almost the very first three years of our lives to master a language? Before birth, virtually all nerve cells are formed and are in their proper places, but it is after birth that head size, brain weight, thickness of the cerebral cortex, long distance connections, myelin insulators and synapses grow. It is during these first years that the qualitative aspects of our behavior are developed by adding and chipping away material from our brains. Later, the growth is more devoted to quantity rather than quality.

2.4 Written language

Written language is what permits us to communicate through physical tokens in a completely time independent manner. While speech uses configurations of matter that change through space and time to convey information, written language only uses permanent configurations of matter that do not change over reasonable changes of space and time. It seems to be a small difference, but it turns out to be of transcendental importance. Thanks to this characteristic that it is possible to communicate no matter the distance in space and time. As an example, it is due to this that it is possible to read Homero's Odyssey, which was written in Greece thousands of years ago.

Written language is a cultural phenomenon. Not every culture developed it. Moreover, impressive cultures have existed without needing a written language at all. Good examples are all the cultures that flourished in the Americas before the arrival of Christopher Columbus.

2.5 Learning written language

Written language acquisition is done from learning the correlation between utterances and their symbolic representations. The character and type of this correlation depends on the language. It usually involves learning how to write. This process, as everything related to written language, is a cultural process. While spoken language is normally learned in an unconscious way during the early infancy, written language is an excellent example of a conscious process.

Chapter-3

HUMAN SPEECH PRODUCTION SYSTEM

3.1 Introduction

Undoubtedly, ability to speak is the most important way for humans to communicate between each other. Speech conveys various kind of information, which is essentially the meaning of information speaking person wants to impart, individual information representing speaker and also some emotional filling. Speech production begins with the initial formalization of the idea which speaker wants to impart to the listener. Then speaker converts this idea into the appropriate order of words and phrases according to the language. Finally, his brain produces motor nerve commands, which move the vocal organs in an appropriate way. Understanding of how human produce sounds forms the basis of speaker identification.

3.2 Anatomy of human speech production system

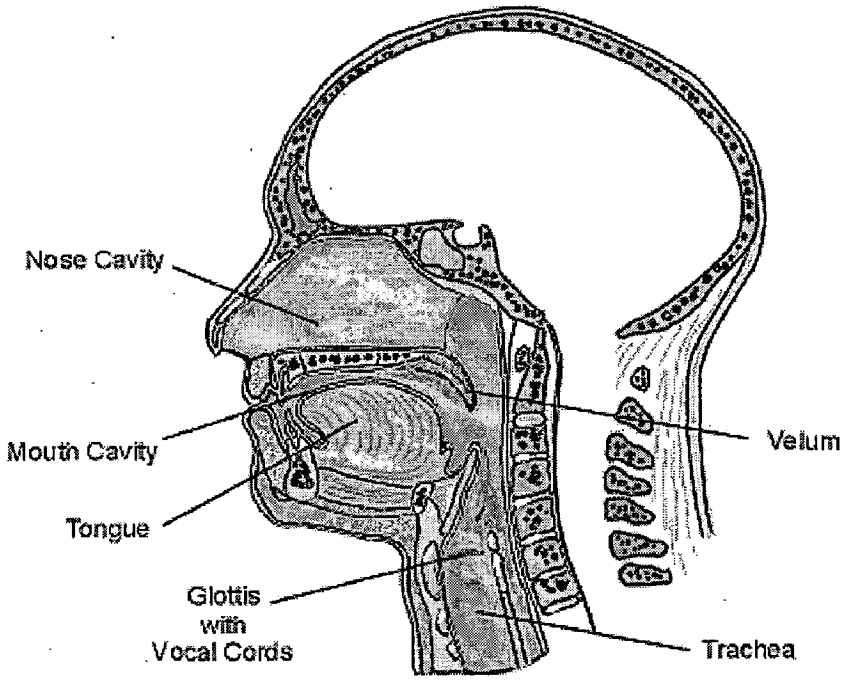


Fig.3.1 Cross sectional view of human vocal tract

The sound is an acoustic pressure formed of compressions and rarefactions of air molecules that originate from movements of human anatomical structures. Most important components of the human speech production system are the *lungs* (source of air during speech), *trachea* (windpipe), *larynx* or its most important part *vocal cords*

(organ of voice production), *nasal cavity* (nose), *soft palate* or *velum* (allows passage of air through the nasal cavity), *hard palate* (enables consonant articulation), *tongue*, *teeth* and *lips*. All these components, called *articulators* by speech scientists, move to different positions to produce various sounds. Based on their production, speech sounds can also be divided into consonants and voiced and unvoiced vowels. From the technical point of view, it is more useful to think about speech production system in terms of acoustic filtering operations that affect the air going from the lungs. There are three main cavities that comprise the main acoustic filter. They are *nasal*, *oral* and *pharyngeal* cavities. The articulators are responsible for changing the properties of the system and form its output. Combination of these cavities and articulators is called *vocal tract*. Its simplified acoustic model is represented in Fig.3.2.

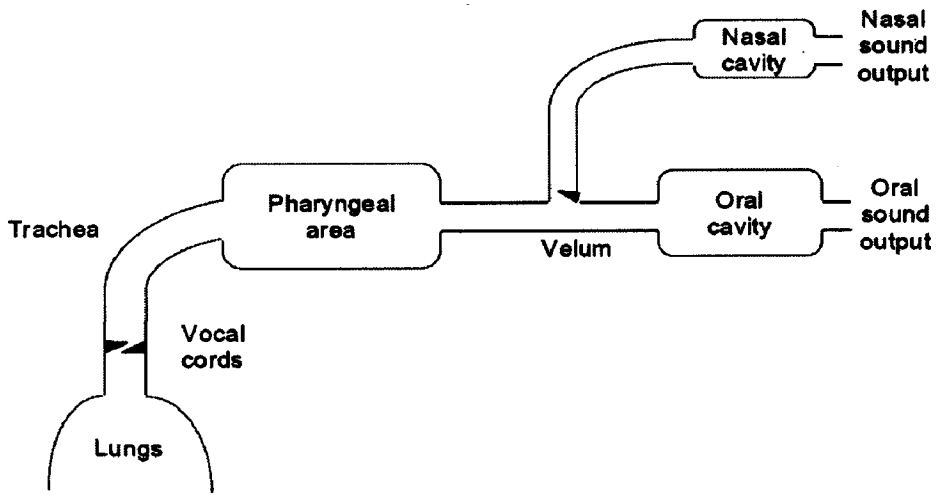


Fig: 3.2 Vocal tract model

Speech production can be divided into three stages: first stage is the sound source production, second stage is the articulation by vocal tract, and the third stage is sound radiation or propagation from the lips and/or nostrils. A *voiced sound* is generated by vibratory motion of the vocal cords powered by the airflow generated by expiration. The frequency of oscillation of vocal cords is called the *fundamental frequency*. Another type of sounds - *unvoiced sound* is produced by turbulent airflow passing through a narrow constriction in the vocal tract.

In a speech recognition task, the knowledge of the physical properties of human vocal tract is defined in general, it is assumed that vocal tract carries most of the speaker related information. However, all parts of human vocal tract described above can serve as

speaker dependent characteristics. Starting from the size and power of lungs, length and flexibility of trachea and ending by the size, shape and other physical characteristics of tongue, teeth and lips. Such characteristics are called *physical distinguishing factors*. Another aspect of speech production that could be useful in discriminating between speakers are called *learned factors*, which include speaking rate, dialect, and *prosodic* effects.

3.3 Vocal tract model

In order to develop an automatic speech recognition system, it is desired to construct reasonable model of human speech production system. Having such a model, one can extract its properties from the signal and, using them, one can decide whether or not two signals belong to the same model and as a result to the same word. Modeling process is usually divided into two parts: the excitation (or source) modeling and the vocal tract modeling. This approach is based on the assumption of independence of the source and the vocal tract models. In the *continuous-time* vocal tract model also called as *multitube lossless model,* is based on the fact that production of speech is characterized by changing the vocal tract shape. Because the formalization of such a time-varying vocal-tract shape model is quite complex, in practice, it is simplified to the series of concatenated lossless acoustic tubes with varying cross-sectional areas , as shown in Fig.3.3. This model consists of a sequence of tubes with cross-sectional areas A_k and lengths L_k. In practice, the lengths of tubes assumed to be equal. If a large amount of short tubes is used, then one can approach to the continuously varying cross-sectional area, but at the cost of more complex model. Tract model serves as a transition to the more general *discrete-time* model, also known as *source-filter model*, which is shown in Fig.3.4.

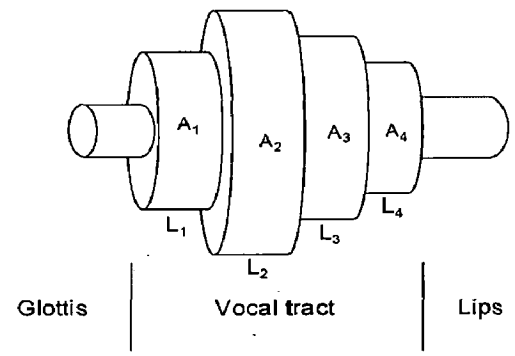


Fig.3.3 Multi tube lossless model

In this model, the voice source is either a periodic pulse stream or uncorrelated white noise, or a combination of these. This assumption is based on the evidence from human anatomy that all types of sounds, which can be produced by humans, are divided into three general categories: voiced, unvoiced and combination of these two. Voiced signals can be modeled as a basic or fundamental frequency signal filtered by the vocal tract and unvoiced as a white noise also filtered by the vocal tract. Here *E(z)* represents the *excitation function*, *H(z)* represents the *transfer function*, and *s(n)* is the output of the whole speech production system. Finally, we can think about vocal tract can be thought of being a digital filter, which affects source signal and produced sound output as a filter output. Then based on the digital filter theory the parameters of the system from Its output can be extracted.

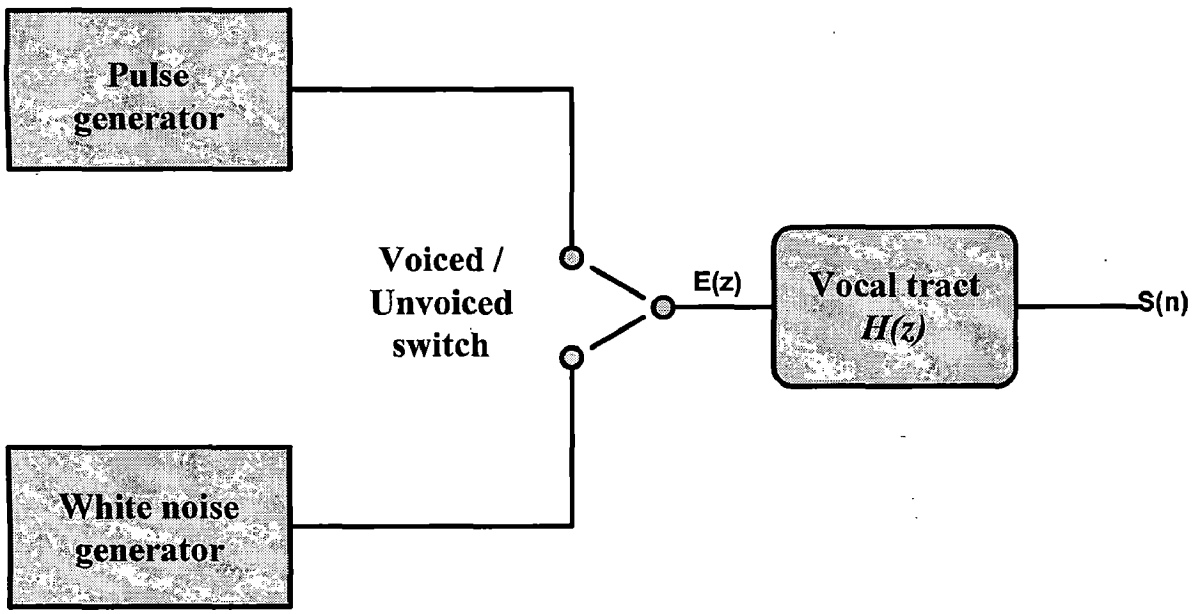


Fig.3.4 Source-filter model

The issues described in this chapter serve as a basis for developing speech recognition techniques described in the next chapter.

Chapter-4

FEATURE EXTRACTION

4.1 Introduction

In this chapter the possible ways of extracting speech discriminative characteristics from speech signal are discussed. A wide range of possibilities exist for parametrically representing a speech signal and its content.

The acoustic speech signal contains different kind of information about speech. This includes "high-level" properties such as dialect, context, speaking style, emotional state of speaker and many others [13]. A great amount of work has been already done in trying to develop Feature extraction algorithms. But these efforts are mostly impractical because of their complexity and difficulty in measuring the speech discriminative properties used by humans [13]. More Useful approach is based on the "low-level" properties of the speech signal such as *pitch* (fundamental frequency of the vocal cord vibrations), *intensity*, *formant frequencies* and their *bandwidths*, *spectral correlations*, *short-time spectrum* and others [14].

From the continuous speech recognition task point of view, it is useful to think about speech signal as a sequence of *features* that characterize both the speaker as well as the speech. It is an important step in recognition process to extract sufficient information for good discrimination in a form and size which is amenable for effective modeling [15]. The amount of data, generated during the speech production, is quite large while the essential characteristics of the speech process change relatively slowly and therefore, they require less data. According to these matters *feature extraction* is a process of reducing data while retaining speaker discriminative information [15, 16].

4.2 Short-Term analysis

Because of its nature, the speech signal is a slowly varying signal or *quasi-stationary*. It means that when speech is examined over a sufficiently short period of time (20-30 milliseconds) it has quite stable acoustic characteristics. It leads to the useful concept of describing human speech signal, called "*short-term analysis*", where only a portion of the signal is used to extract signal features at one time. It works in the following way: predefined length window (usually 20-30 milliseconds) is moved along the signal with an

overlapping (usually 30-50% of the window length) between the adjacent frames. Overlapping is needed to avoid losing of information. Parts of the signal formed in such a way are called *frames*.

4.3 Windowing

In order to prevent an abrupt change at the end points of the frame, it is usually multiplied by a *window function*. The operation of dividing signal into short intervals is called *windowing* and such segments are called *windowed frames* (or sometime just *frames*). There are several window functions used in speech recognition area, but the most popular is *Hamming window function*, which is described by the following equation:

$$w(n) = 0.54 - 0.46\cos\left(\frac{2n\pi}{N-1}\right) \quad \text{..... (4.1)}$$

where N is the size of the window or frame. A set of features extracted from one frame is called *feature vector*. Overall overview of the short-term analysis approach is represented in Fig.4.1.

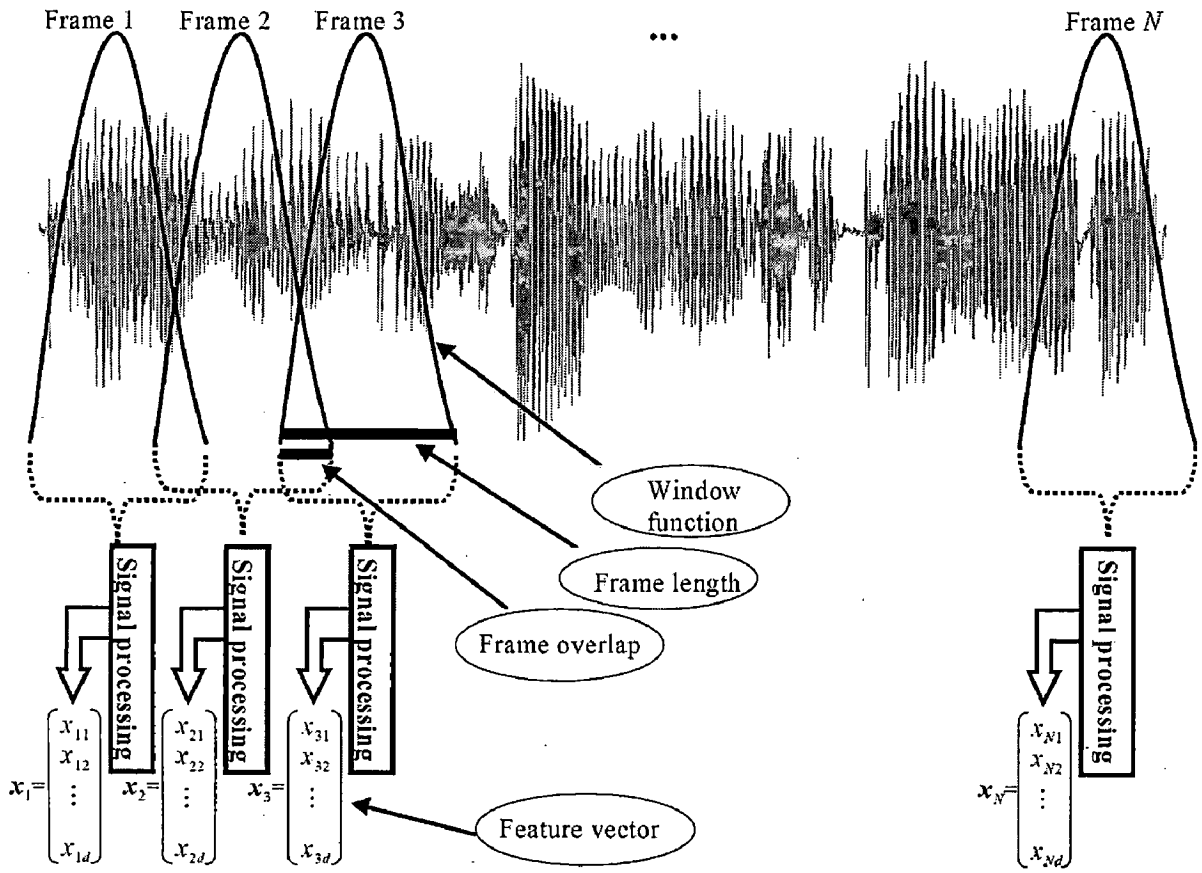


Fig.4.1 Short-Term analysis

4.4 Cepstrum

According to the issues described in the subsection 3.2, the speech signal *s(n)* can be represented as a "quickly varying" source signal *e(n)* convolved with the "slowly varying" impulse response *h(n)* of the vocal tract represented as a linear filter [16]. There is access only to the output (speech signal) and it is often desirable to eliminate one of the components. Separation of the source and the filter parameters from the mixed output is in general difficult problem when these components are combined using non linear operation, but there are various techniques appropriate for components combined linearly. The *cepstrum* is representation of the signal where these two components are resolved into two additive parts [16]. It is computed by taking the inverse DFT of the logarithm of the magnitude spectrum of the frame. This is represented in the following equation:

$$Cepstrum(frame) = IDFT(\log(|DFT(frame)|)) \quad \text{..... (4.2)}$$

Some explanation of the algorithm is therefore needed. By moving to the frequency domain convolution is changed to multiplication. Then by taking logarithm moving from the multiplication to the addition. That is desired division into additive components. Then linear operator inverse DFT is applied, knowing that the transform will operate individually on these two parts and knowing what Fourier transform will do with quickly varying and slowly varying parts. Namely it will put them into different, hopefully separate parts in new, also called *frequency* axis [20]. One example of speech magnitude spectrum is shown in Fig 4.2 [20].

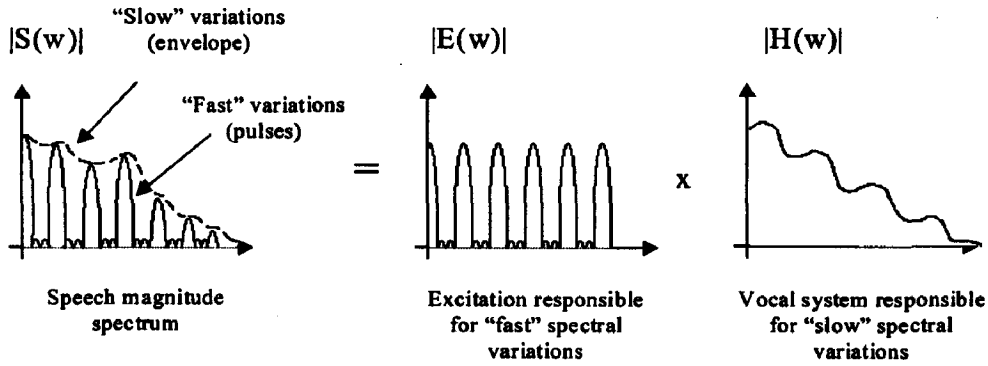


Fig.4.2 Speech magnitude spectrum

From the Fig.4.2 it can be seen that the speech magnitude spectrum is combined from slow and quickly varying parts. But there is still one problem: multiplication is not a linear operation. This can be solve it by taking logarithm from the multiplication as described earlier. Finally, The result of the inverse DFT is shown in Fig.4.3 [16].

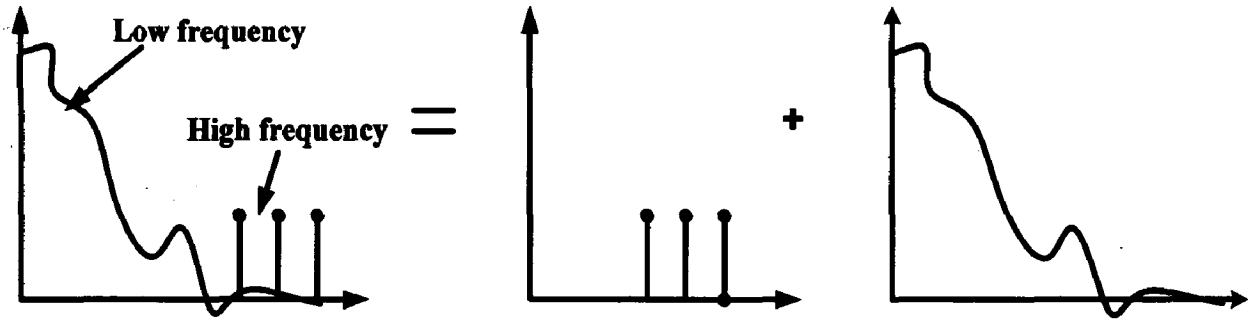


Fig.4.3 Cepstrum

From this figure it can seen that two components are clearly distinctive now. The details of cepstrum can be seen in references [15, 16, 17].

4.5 Mel-frequency cepstrum analysis

4.5.1 Introduction

Mel-frequency cepstrum coefficients (MFCC) are well known features used to describe speech signal. Mel-scale grouping is a commonly employed dimensionality reduction technique in automatic speech recognition systems. The grouping follows a filtering operation that converts the time series representation of speech into a frequency representation. They are based on the known evidence that the information carried by low-frequency components of the speech signal is phonetically more important for humans than carried by high-frequency components [16]. Technique of computing MFCC is based on the short-term analysis, and thus from each frame a MFCC vector is computed.

4.5.2 Background

To understand how the warping was incrementally modified in this experiment, the reader must first understand in general how cepstral coefficients and mel-warped cepstral coefficients (MFCCs) are calculated for use in typical speech recognition systems. This paper will specifically discuss how the mel-warped cepstrum is calculated.

Given a frame of speech, the following steps compute the cepstral coefficients:

1) Window the speech frame, using a Hamming (or other) window.

2) Do zero padding to achieve a frame length suitable for an FFT.

3) Do the FFT.

4) Find the Power Spectrum.

5) Take the Log of the Power Spectrum. (This is the cepstrum.)

6) Inverse FFT. (The results are the cepstral coefficients.)

The "Mel" is a unit of measure of perceived pitch or frequency of a tone. The Mel scale was developed by Stevens and Volkman (1940) as a result of a study of human auditory perception (the field of psychoacoustics) using the following procedure:

1) Choose the reference frequency as 1000 Hz and designate it "1000 Mels".

2) Listeners were then presented a signal and asked to change it's frequency until the pitch they perceived was twice the reference, then 10 times the reference, etc. and then half the reference, 1/10 the reference, etc.

3) From this data, the mel scale was constructed.

The mel scale is approximately linear below 1 kHz and logarithmic above. This is approximated using the following formula, and shown in Fig.4.4:

$$Mel(f) = 1127 \log\left(1 + \frac{f}{700}\right)$$

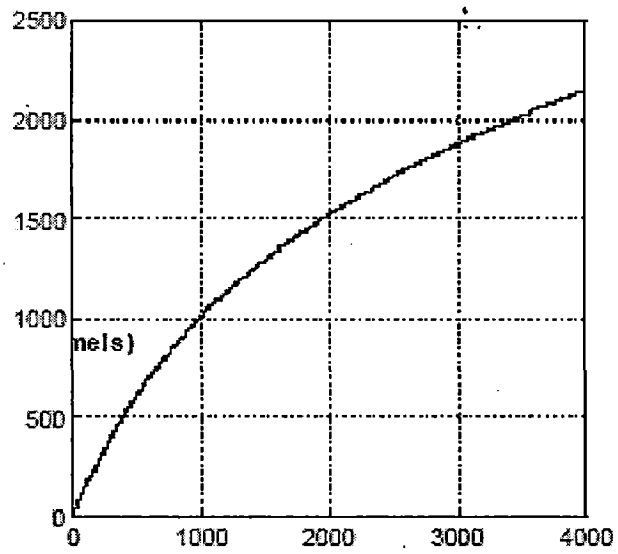


Fig.4.4 The mel scale

The process of extracting MFCC from continuous speech is illustrated in Figure 4.5.

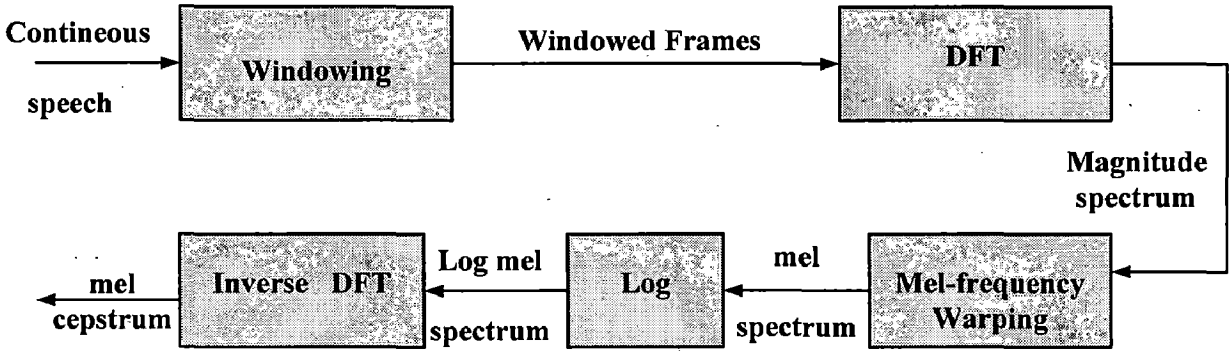


Fig.4.5 Computational figures for mel-cepstrum

As described above, to place more emphasize on the low frequencies one special step before inverse DFT in calculation of cepstrum is inserted, namely mel-scaling. A "*mel*" is a unit of special measure or scale of *perceived pitch* of a tone [16]. It does not correspond linearly to the normal frequency, indeed it is approximately linear below 1 kHz and logarithmic above [16]. This approach is based on the psychophysical studies of human perception of the frequency content of sounds [16, 17]. One useful way to create mel-spectrum is to use a filter bank, one filter for each desired mel-frequency component. Every filter in this bank has triangular bandpass frequency response. Such filters compute the average spectrum around each center frequency with increasing bandwidths, as displayed in Fig.4.6.

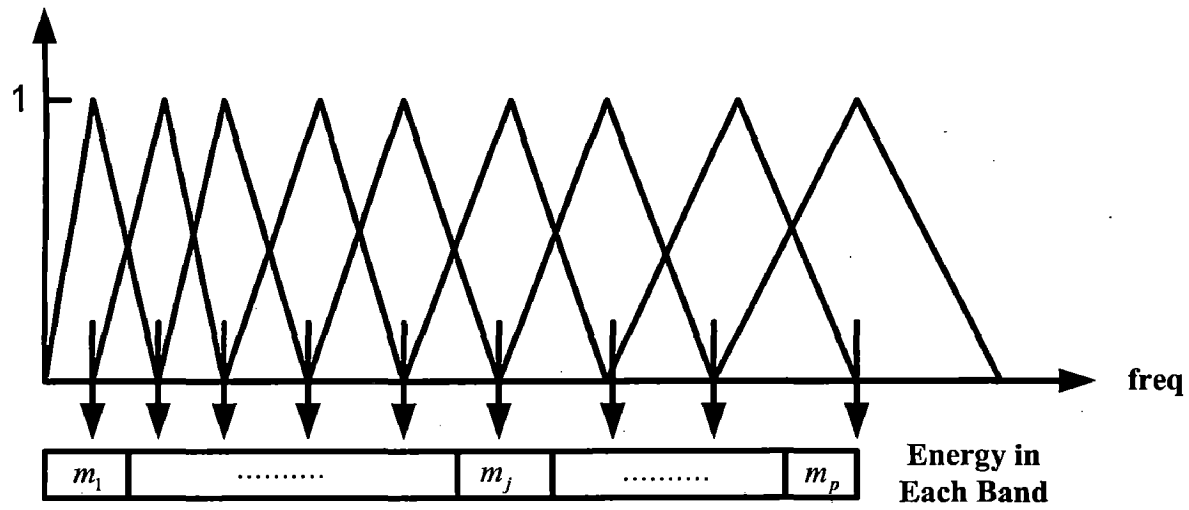


Fig.4.6 Mel-scale filter bank

This filter bank is applied in frequency domain and therefore, it simply amounts to taking these triangular filters on the spectrum. In practice the last step of taking inverse DFT is replaced by taking *discrete cosine transform (DCT)* for computational efficiency.

The number of resulting mel-frequency cepstrum coefficients is practically chosen relatively low, in the order of 12 to 26 coefficients. The zeroth coefficient is usually dropped out because it represents the average log energy of the frame and carries only a little speaker specific information. However, MFCC are not equally important in speech recognition [11] and thus some coefficients weighting might by applied to acquire more precise result. A different approach for computation of MFCC other than described in this work cab be found in [18] that is simplified by omitting filter bank analysis. More details about MFCC can be found in [4, 15, 16, 17, 19, 24].

Chapter-5

FEATURE MATCHING

5.1 Introduction

In this chapter we are going to discuss all possible ways for matching the test signal with the reference signal which is stored in database by using the Artificial Neural Network. A wide range of possibilities exists for recognition of speech signal.

Speech is a most natural and efficient way to exchange information for human beings. To make a real "intelligent computer," it is important that the machine can "hear," "understand," and "act upon" spoken information, and also "speak" to complete the information exchange. Therefore, speech recognition is essential for a computer to reach the goal of natural human-computer communication [16].

Many algorithms are currently used for searching a best matching path between two signals. Some of the more successful techniques include dynamic programming; hidden Markov models and neural networks applied at both the phoneme and at the word level. However, neural networks remain the most widely used algorithm for real-time recognition systems. It is considered sufficiently mature to solve sequential decision problems.

5.2 NEURAL NETWORKS

5.2.1 Biological Inspiration

The brain consists of a large number (approximately 10^{11}) of highly connected elements (approximately 10^{4} connections per element) called neurons. For our purposes these neurons have three components: the dendrites, the cell body and the axon. The dendrites are tree-like receptive networks of nerve fibers that carry electrical signal into the cell body. The cell body effectively sums and thresholds these incoming signals. The axon is a single long fiber that carries the signal from the cell body out to other neurons. The point of contact between an axon of one cell and a dendrite of another cell is called a synapse. It is the arrangements of neurons and the strengths of the individual synapses, determined by a complex chemical process, that establishes the function of the neural network. Fig.5.1 is a simplified schematic diagram of two biological neurons [20].

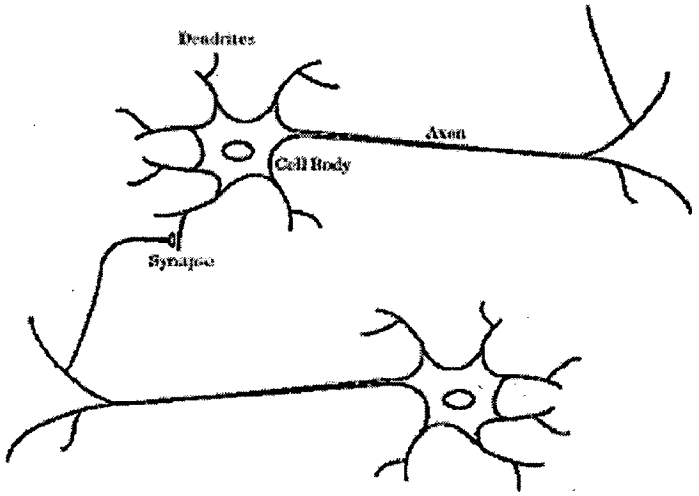


Fig.5.1 Schematic diagram of biological neurons [21]

Some of the neural structure is defined at birth. Other parts are developed through learning, as new connections are made and others waste away. This development is most noticeable in the early stages of life. For example, it has been shown that if a young cat is denied use of one eye during a critical window of time, it will never develop normal vision in that eye.

Neural structures continue to change throughout life. These later changes tend to consist mainly of strengthening or weakening of synaptic junctions. For instance, it is believed that new memories are formed by modification of these synaptic strengths. Thus, the process of learning a new friend's face consists of altering various synapses.

Artificial neural networks do not approach the complexity of the brain. There are, however, two key similarities between biological and artificial neural networks. First, the building blocks of both networks are simple computational devices (although artificial neurons are much simpler than biological neurons) that are highly interconnected. Second, the connections between neurons determine the function of the network.

It is worth noting that even though biological neurons are very slow when compared to electrical circuits (10^{-3} s compared to 10^{-9} s), the brain is able to perform many tasks much faster than any conventional computer. This is in part because of the massively parallel structure of biological neural networks share this parallel structure. Even though most artificial neural networks are currently implemented on conventional digital computers, their parallel structure makes them ideally suited to implementation using VLSI, optical devices and parallel processors.

5.2.2 Artificial neurons

An artificial neuron is an information processing unit that is fundamental to the operation of a neural network. Fig.5.2 shows the schematic representation of an artificial neuron. Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the network function is determined largely by the connections between elements. A neural network can be trained to perform a particular function by adjusting the values of the connections (weights) between elements.

Commonly neural networks are adjusted, or trained, so that a particular input leads to a specific target output. Such a situation is shown below. There, the network is adjusted, based on a comparison of the output and the target, until the network output matches the target. Typically many such input/target pairs are used, in this *supervised learning*, to train a network.

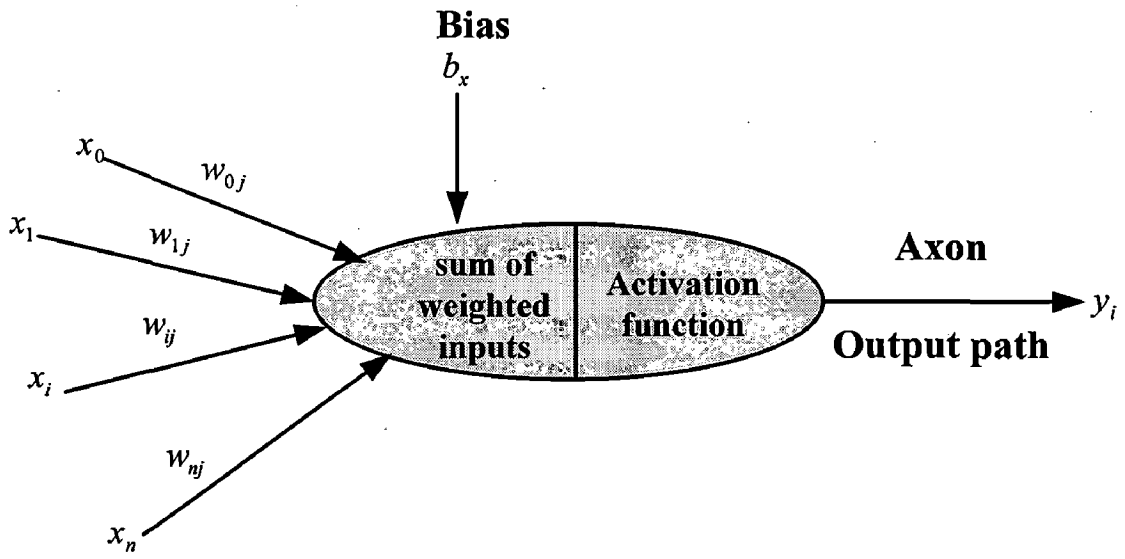


Fig.5.2 An artificial neuron

5.2.3 Neuron modeling

A neuron with a single scalar input with no bias appears on the left below.

Summer output n = WP +b

Neuron output a = f (wp+b)

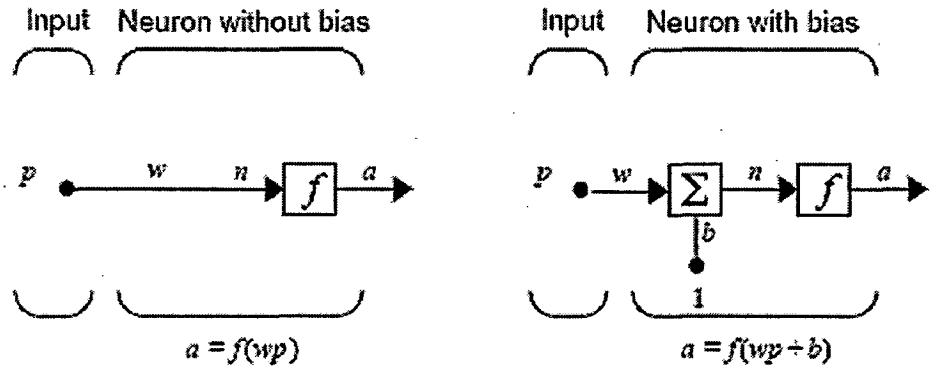


Fig.5.3 Single input neuron [21]

The scalar input *p* is transmitted through a connection that multiplies its strength by the scalar weight *w*, to form the product *wp*, again a scalar. Here the weighted input *wp* is the only argument of the transfer function *f*, which produces the scalar output *a*. The neuron on the right has a scalar bias, *b*. The bias may be viewed as simply being added to the product *wp* as shown by the summing junction or as shifting the function *f* to the left by an amount *b*. It has the effect of raising or lowering the net input of the activation function, depending on whether it is positive or negative, respectively. The bias is much like a weight, except that it has a constant input of 1.

The transfer function net input *n*, again a scalar, is the sum of the weighted input *wp* and the bias *b*. This sum is the argument of the transfer function *f*. Here *f* is a transfer function, typically a step function or a sigmoid function, which takes the argument *n* and produces the output *a*. It is to be noted that *w* and *b* are both *adjustable* scalar parameters of the neuron. The central idea of neural networks is that such parameters can be adjusted so that the network exhibits some desired or interesting behavior. Thus, the network can be trained to do a particular job by adjusting the weight or bias parameters, or perhaps the network itself will adjust these parameters to achieve some desired end.

5.3 TRANSFER FUNCTIONS

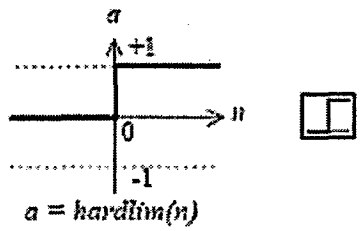
Referring to Fig.5.2, the processing elements consist of two parts. The first part consists of an adder for summing up the input signals, weighted by the respective synapses of the neuron, and the second part consists of an activation function for limiting the amplitude of the out put of the neuron. The activation function is also referred to as a squashing function, in that it squashes the permissible amplitude range of the output

signal to some finite value. The normalized amplitude of the output of a neuron lies within the closed unit interval [0, 1] or [-1, 1].

The activation function, denoted by $\Phi(I)$, defines the output of the neuron in terms of the individual local field *I*. The threshold function passes information (usually a +1 signal) only when the output of the first part of the artificial neuron exceeds the threshold value T. A single neuron with a threshold activation function is known as a single-layer perceptron, whereas the same neuron with a signum activation function is known as an adaline.

$$\Phi(I), \begin{cases} 1 \quad \text{if} \quad I >= 1 \\ 0 \quad \text{if} \quad I <= 0 \end{cases} \qquad (\text{assume } \Phi(I) = a)$$

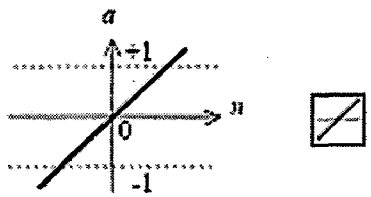
5.3.1 Hard-Limit Transfer Function



a = hardlim(n)

The hard-limit transfer function shown above limits the output of the neuron to either 0, if the net input argument *n* is less than 0; or 1, if *n* is greater than or equal to 0.

5.3.2 Linear transfer function

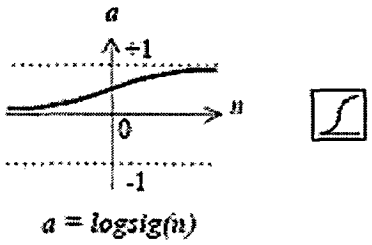


a = purelin(n)

5.3.3 Sigmoid transfer function

The sigmoid function, whose graph is s-shaped, is the common form of activation function used in the construction of ANNs. It is defined as a strictly increasing function that exhibits a graceful balance between linear and non-linear behaviour. The sigmoid function passes negative information when the output is less than the threshold value T and positive information when the output is greater than the threshold value T. It is a

continuous function that varies that gradually between two asymptotic values, typically 0 and 1, or -1 and +1. The sigmoid transfer function shown below takes the input, which may have any value between plus and minus infinity, and squashes the output into the range 0 to 1.



Log-Sigmoid Transfer Function

$$\Phi(I) = \frac{1}{1+e^{-\alpha I}}$$

Where α is a slope parameter of the sigmoid function which adjusts the abruptness of this function as it changes between the two asymptotic values. By varying the parameter α, one can obtain sigmoid function of different slopes.

This transfer function is commonly used in backpropagation networks, in part because it is differentiable. The symbol in the square to the right of each transfer function graph shown above represents the associated transfer function. These icons will replace the general *f* in the boxes of network diagrams to show the particular transfer function being used.

5.4 ARTIFICIAL NEURAL NETWORK

The interconnection of artificial neurons results in an ANN, and its objective is to emulate the function of a human brain to solve scientific, engineering and many other real life problems. As mentioned before, the interconnection of biological neurons is not well understood, but scientists have come up with neural network models and many more are yet to come. These network can generally be classified as feedforward and feedback (or recurrent) types. In a feedforward network, signals from neuron to neuron flow only in the forward direction, whereas in the recurrent network, the signals can flow in a forward as well as backward or lateral direction. A few network models can be listed as follows:

Feedforward:

- Perceptron
- Adaline and Madaline
- Backpropagation Network
- Radial Basis Function Network (RBFN)
- General Regression Network
- Modular Neural Network (MNN)
- Learning Vector Quantization(LVQ) Network
- Probabilistic Neural Network (PNN)
- Fuzzy Neural Network (FNN)

Recurrent:

- Hopfield Network
- Boltzmann Machine
- Kohonen's Self-Organizing Feature Map (SOFM)
- Recirculation Network
- Brain-State-in-a-Box (BSB)
- Adaptive Resonance Theory (ART) Network
- Bi-directional Associative Memory (BAM)

A Network can be defined as static or dynamic, depending on weather it is a simulating static or dynamic system. It has been claimed [21] that any problem that can be solved by a recurrent network can also be solved by a feedforward network with the proper external connections.

5.4.1 Backpropagation training algorithms

Backpropagation is a systematic method for training multiple-layer(three or more) artificial neural networks. it was created by generalizing the Widrow-Hoff learning rule to multiple-layer networks and nonlinear differentiable transfer functions. The standard backpropagation algorithm is summarised as.

1) Build a network with the chosen number of input, hidden and output units.
2) Initialize all the weights to low random values.
3) Choose a single training pair at random.
4) Copy the input pattern to the input layer.

5) Cycle the network so that the activations from the inputs generate the activations in the hidden and output layers.
6) Calculate the error derivative between the output activation and target output.
7) Backpropagate the summed products of the weights and errors in the output layer in order to calculate the error in the hidden units.
8) Update the weights attached to each unit according to the error in that unit, the output from the unit below it, and the learning parameters, until the error is sufficiently low or the network settles.

The propagation algorithm has been critical to advances in neural networks, because of the limitations of the one-and two-layer networks. Today it is estimated that 80% of all applications utilize the backpropagation algorithm in one form or another. Prior to the development of backpropagation, attempts to use perceptrons with more than one layer of weights were frustrated by what was called the 'weight assignment problem'.

Let us consider a typical neuron as shown in fig.5.2 with inputs x_i, weights W_i, a summation function in the left half of the neuron, and a non-linear activation function n the right half. The summation of the weighted inputs designated I is given as follows:

$$I = x_1 W_1 + x_2 W_2 + \ldots\ldots\ldots + x_n W_n = \sum_{i=1}^{n} x_i W_i \quad \ldots\ldots\ldots\ldots (5.1)$$

We have used a typical sigmoid function as the non-linear activation function, give by

$$\Phi(I) = \frac{1}{(1+e^{-\alpha I})} = (1+e^{-\alpha I})^{-1} \quad \ldots\ldots\ldots\ldots (5.2)$$

This sigmoid function is a logistic function, which monotonically increases from a lower limit (0 or -1) to an upper limit (+1) as I increase; the values vary between 0 and 1, with a value of 0.5 when I is zero. An examination of Fig.5.5 shows that the derivative (slope) of the curve asymptotically approaches zero as the input I approaches $\pm\infty$, and it reaches a maximum value of $\alpha/4$ when I=0, as shown in Fig.5.6. Since this derivative function will be utilized in backpropagation, let us reduce it to its most simple form. If we take the derivative of eqn.5.2, we get

$$\frac{\partial \Phi(I)}{\partial I} = (-1)(1+e^{-\alpha I})^{-2} e^{-\alpha I}(-\alpha)$$

$$= \alpha e^{-\alpha I}(1+e^{-\alpha I})^{-2} = \alpha e^{-\alpha I}\Phi^2(I) \quad \ldots\ldots\ldots\ldots (5.3)$$

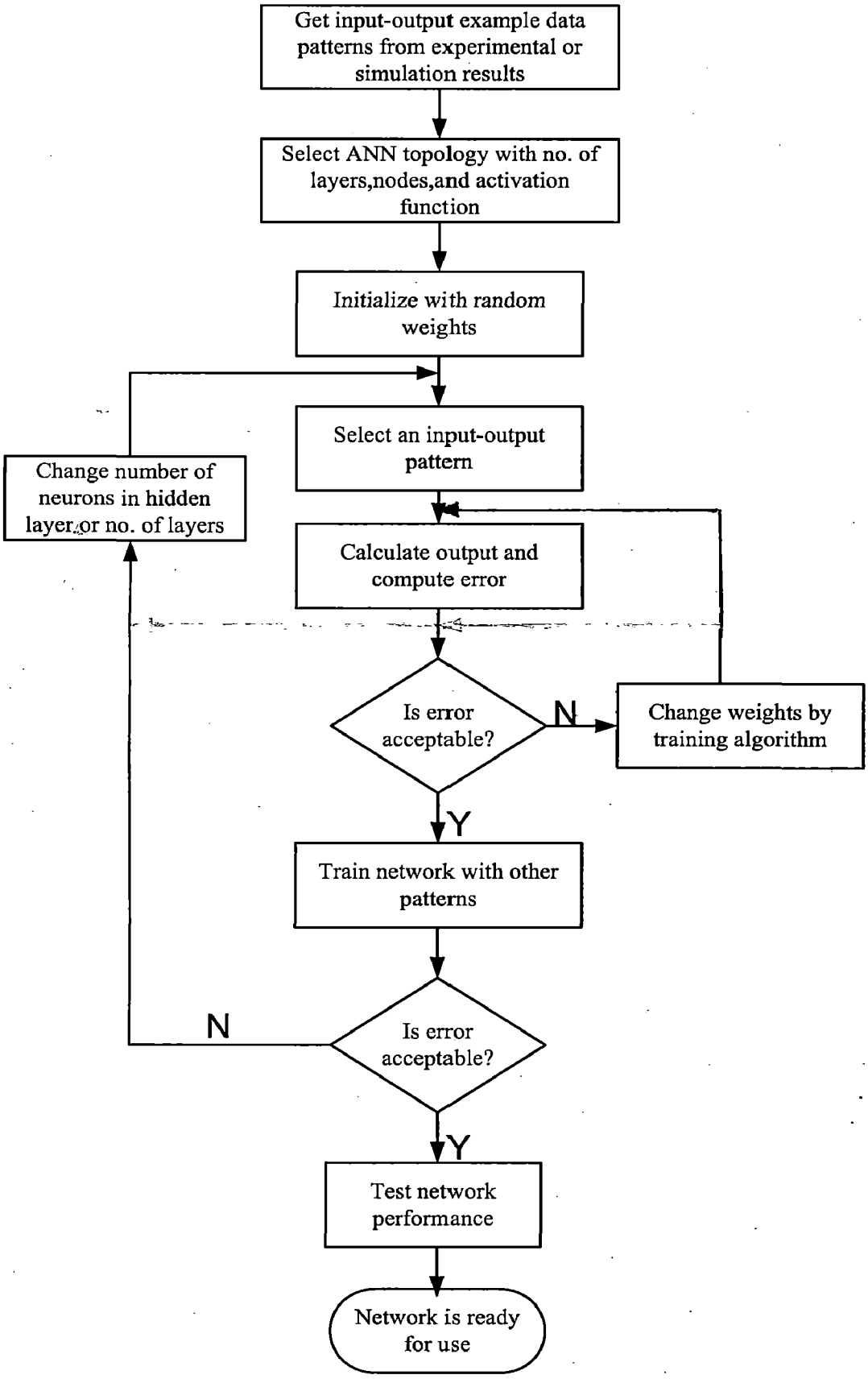


Fig. 5.4 Flowchart for backpropagation training of feedforward neural network

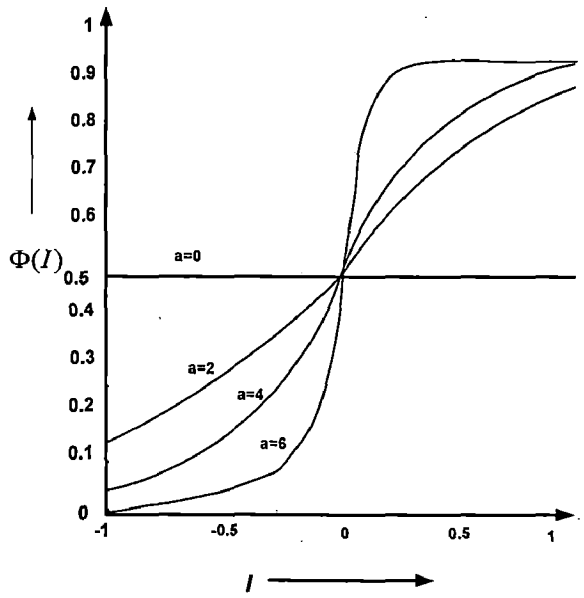


Fig. 5.5 The logistic activation function

If we solve eqn.5.2 for $e^{-\alpha I}$, substitute it in eqn.5.3, and simplify, we get

$$\frac{\partial\Phi(I)}{\partial I} = \alpha\frac{1-\Phi(I)}{\Phi(I)}\Phi^2(I) = \{\alpha[1-\Phi(I)]\Phi(I)\} = \alpha(1-\Phi)\Phi \quad \text{..... (5.4)}$$

where Φ(I) has been simplified to Φ by dropping (I).

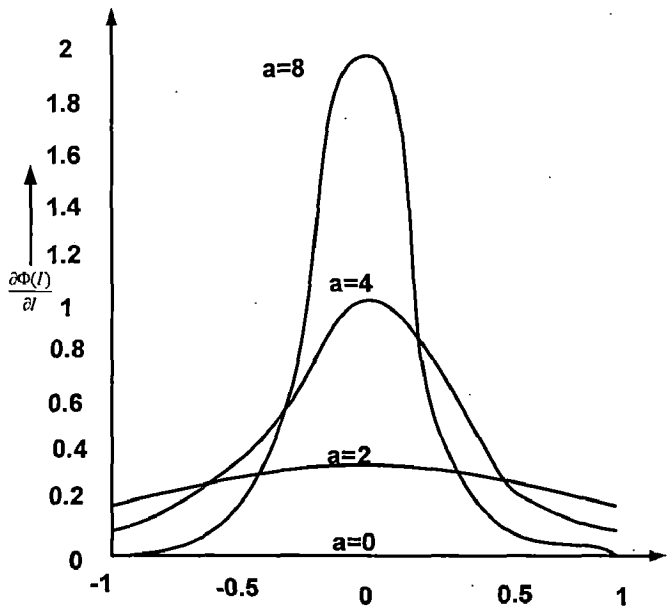


Fig. 5.6 First derivative of the logistic activation function

Multilayer networks have greater representational power than single-layer networks only if non-linearities are introduced. The logistic function provides the

necessary non-linearity. While applying the backpropagation algorithm, any non-linear function can be used if it is differentiable everywhere and monotonically increasing with I. Sigmoidal functions, including logistic, hyperbolic tangent, and arctangent functions, meet these requirements. The arctangent function, denoted as $\tan^{-1}$, has the form

$$\Phi(I) = \frac{2}{\pi}\tan^{-1}(\alpha I) \quad \text{..... (5.5)}$$

where the factor 2/Π reduces the amplitude of the arctangent function so that it is restricted to the range -1 to +1. The constant α determines the rate at which the function changes between the limits -1 and +1; the slope of the function at the origin is 2α /Π . The α- value influences the shape the arctangent function in the same way that α influences the logistic function in Fig.5.5. The arctangent function has a sigmodial shape as shown in Fig.5.7. The derivative is

$$\frac{\partial\Phi(I)}{\partial I} = \frac{2}{\pi}\left[\frac{\alpha}{1+\alpha^2 I^2}\right] \quad \text{..... (5.6)}$$

which would be used in place of eqn.5.4 if the arctangent function replaced the logistic activation function.

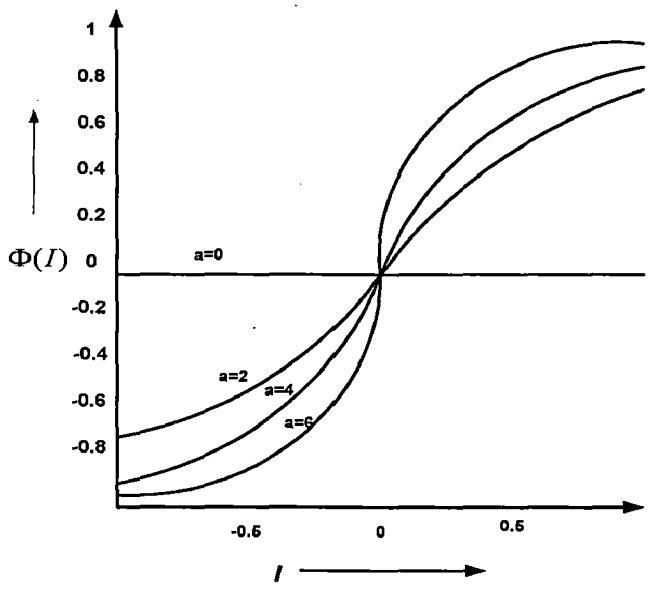


Fig.5.7 Arctangent activation function

The hyperbolic tangent function has the form

$$\Phi(I) = \tanh(\alpha I) = \frac{e^{\alpha I} - e^{-\alpha I}}{e^{\alpha I} + e^{-\alpha I}} \qquad \text{..... (5.7)}$$

and its shape is shown in Fig.5.8. Its derivative is

$$\frac{\partial \Phi(I)}{\partial I} = \alpha \operatorname{sec} h^2(\alpha I) \qquad \text{..... (5.8)}$$

The slope of Φ(I) at the origin is 4α; it determines the rate at which the function changes between the limits -1 and +1 in the same general way that α influences the shape of the logistic function in Fig.5.5.

The use of a sigmoidal function provides a form of 'automatic gain control'; that is, for small values of I near zero, the slope of the input-output curve is steep producing a high gain, since all sigmoid activation functions have derivatives with bell shapes of the type shown in Fig.5.6. As the magnitude of I become greater in a positive or negative direction, the gain decreases. Hence, large signals can be accommodated without saturation, as given in Fig.5.7.

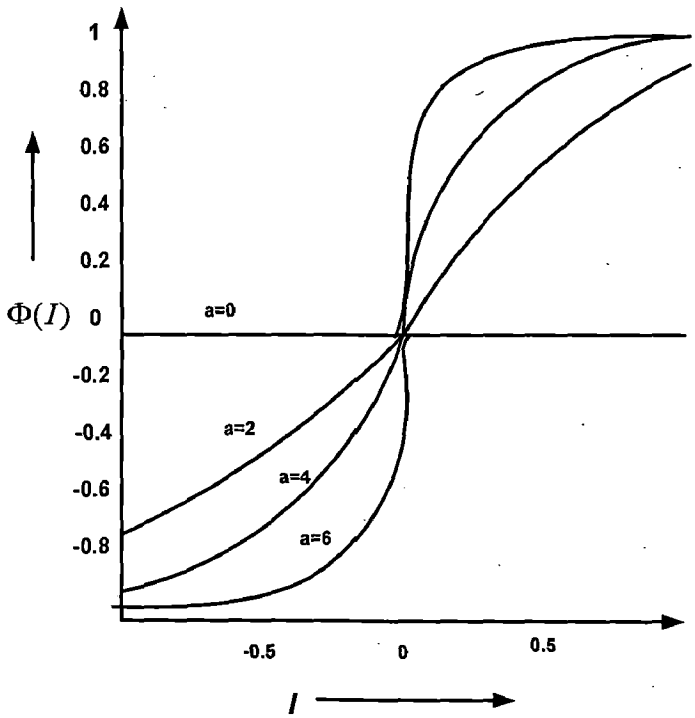
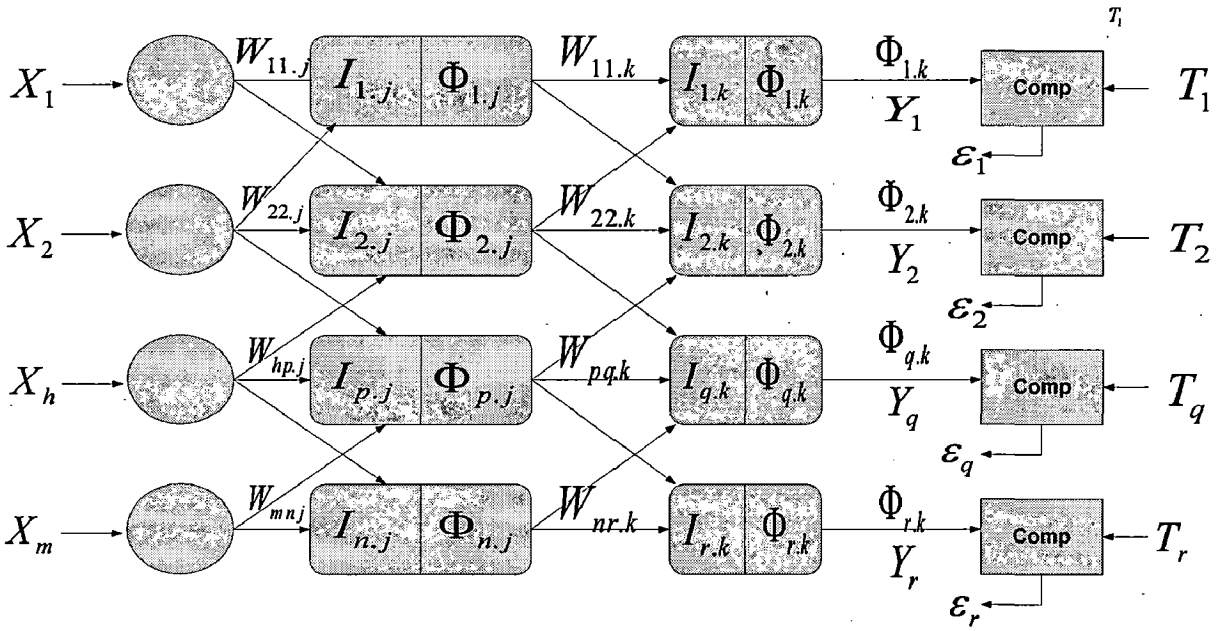


Fig.5.8 Hyperbolic tangent activation function

5.5. BACK PROPAGATION TRAINING FOR A MULTILAYER NEURAL NETWORK

Before discussing the details of the back propagation process, let us consider the benefits of the middle layer(s) in an artificial neural network. A network with only two layers (input ad output) can only represent the input with whatever representation already exists in the input data. Hence, if the data are discontinuous or non-linearly separable, the innate representation is inconsistent and the mapping cannot be learned.

The network is composed of a hierarchy of processing units, organized in a series of two or more mutually exclusive sets of neurons or layers. This mainly consists of the input layer, the output layer, and a hidden layer between these two layers. Weights connect each unit in one layer only to those in the next higher layer. The output of the unit is scaled by the value of the connecting weight and is fed forward to provide a portion of the activation for the units in the next higher layer.



i th layer	*j* th layer	*k* th layer
index h	index p	index q
1 – m	1 – n	1 - r
m nodes	n nodes	r nodes

Fig.5.9 A Multilayer neural network showing the symbols and indices used in deriving the backpropagation training algorithm

Let us consider the three-layer network shown in Fig.5.9, where all activation functions are logistic functions. It is important to note that backpropagation can be applied to an

artificial neural network with any number of hidden layers. The training objective is to adjust the weights so that the application of a set of inputs produces the desired outputs. To accomplish this, the network is usually trained with a large number of input-output pairs.

The training procedure is the following.

1. Generate small random values (both positive and negative) for the weights to ensure that the network is not saturated by large values of weights. (If all weights start at equal values, and the desired performance requires unequal weights, the network will not train at all).
2. Choose a training pair from the training set.
3. Apply the input vector to the network input.
4. Calculate the network output.
5. Calculate the error, i.e., the difference between the network output and the desired output.
6. Adjust the weights of the network in a way that minimizes this error.
7. Repeat steps (2)-(6) for each input-output pair in the training set until the error for the entire system is acceptably low.

The training of an artificial neural network involves two passes. In the forward pass, the input signals move forward from the network input to the output. In the backward pass, the calculated error signals propagate backward through the network, where they are used to adjust the weights. The calculation of the output is carried out, layer by layer, in the forward direction. The output of one layer is the input to the next layer as in feedback. In the reverse pass, the weights of the output neuron layer are adjusted first, since the target value of each output neuron is available to guide the adjustment of the associated weights, using the delta rule. Next, the weights of the middle layers are adjusted. The fact that the middle-layer neurons have no target values makes this process complex. Hence, the training is more complicated, because the error must be propagated back through the network, including the non-linear functions, layer by layer. The number of hidden units depends on the number of input units. Kolomogorov's theorem states that any function of n variables may be represented by a superposition of a set of 2n+1 univariate functions to derive the upper bound for the required number of

hidden units as one greater than twice the number of hidden units as one has inputs. When choosing the number of hidden units h, the following points must be kept in mind.

- The value of h should never be more than twice the number of input units.
- *p* patterns of I elements can be loaded into $\log_2 P$ hidden units. Therefore, if good generalization is required, considerably fewer patterns should be used.
- It should be ensured that there are at least 1/e times as many training examples as there are weights in the network.
- Feature extraction requires fewer hidden units than inputs.
- Learning many examples of disjointed inputs requires more hidden units than inputs.
- The number of hidden units required for a classification task increases with the number of classes in the task. Large networks require longer training times.

5.5.1 Calculation of weights for output-layer neurons

Let us consider the details of the backpropagation learning process for the weights of the output layer. When neuron j is located in the output layer of the network, it is supplied with a desired response of its own. Fig.5.10 suggests training of neurons leading to the output layer designated by the subscript k with neurons p and q, outputs $\Phi_{p,j}(I)$ and $\Phi_{q,j}(I)$, input weights $W_{hp,j}$ and $W_{pq,k}$, and a target value T_q. Henceforth, the notation (I) in $\Phi_{q,kj}(I)$ will be dropped for convenience.

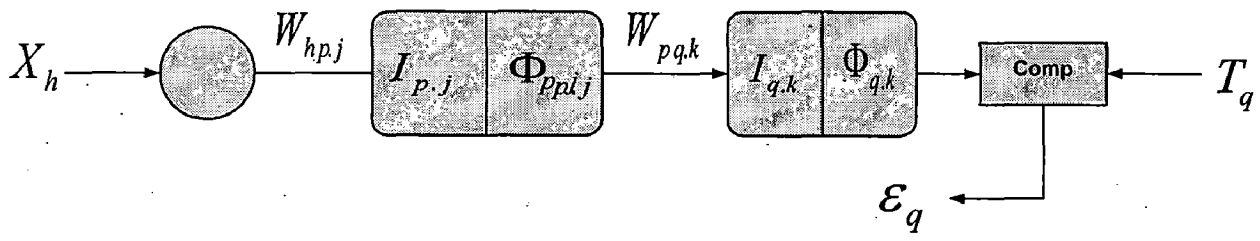


Fig.5.10 Representation of neurons for the calculation of the output-layer neuron's weight

The output of the neuron in layer k is subtracted from its target value, shown in Fig.5.11, and squared to produce the square error signal, which for a layer-k neuron is

$$\varepsilon = \varepsilon_q = [T_q - \Phi_{q,k}] \qquad \text{..... (5.9)}$$

Since only one output error is involved, hence

$$\varepsilon^2 = \varepsilon_q{}^2 = [T_q - \Phi_{q.k}]^2 \qquad \text{..... (5.10)}$$

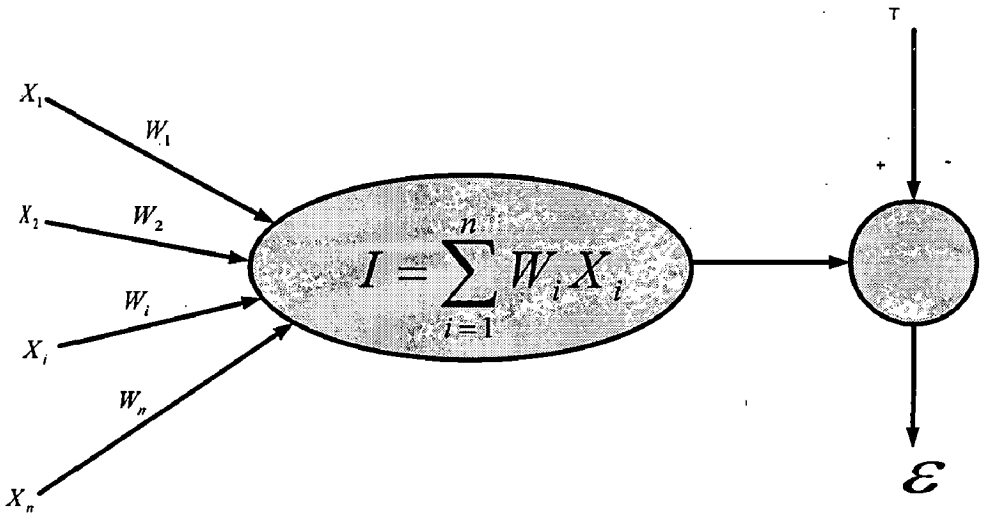


Fig. 5.11 A neuron with the target and error

According to the delta rule, the change in a weight is proportional to the rate of change of the squared error with respect to that weight. That is,

$$\Delta W_{pq.k} = -\eta_{p.q} \frac{\partial \varepsilon_q{}^2}{\partial W_{pq.k}} \qquad \text{..... (5.11)}$$

Where $\eta_{p.q}$ is a constant of proportionality called the learning rate. To evaluate this partial derivative, we use the chain rule of differentiation:

$$\frac{\partial \varepsilon_q{}^2}{\partial W_{pq.k}} = \frac{\partial \varepsilon_q{}^2}{\partial \Phi_{q.k}} \frac{\partial \Phi_{q.k}}{\partial I_{q.k}} \frac{\partial I_{q.k}}{\partial W_{pq.k}} \qquad \text{..... (5.12)}$$

Each of these terms is evaluated in turn. The partial derivative of eqn.5.10 with respect to $\Phi_{q.k}(I)$ gives

$$\frac{\partial \varepsilon_q{}^2}{\partial \Phi_{q.k}} = -2[T_q - \Phi_{q.k}] \qquad \text{..... (5.13)}$$

We know that

$$\frac{\partial \Phi_{q.k}}{\partial I_{q.k}} = \alpha \Phi_{q.k}[1 - \Phi_{q.k}] \qquad \text{..... (5.14)}$$

Observe that $I_{q.k}$ is the sum of the weighted inputs from the middle layer. That is,

$$I_{q.k} = \sum_{p=1}^{n} W_{pq.k}\Phi_{p.j} \qquad \text{..... (5.15)}$$

Taking the partial derivative with respect to $W_{pq.k}$ gives

$$\frac{\partial I_{q.k}}{\partial W_{pq.k}} = \Phi_{p.j} \qquad \text{..... (5.16)}$$

Since we are dealing with one weight, only one term of the summation of eqn.5.15 survives. Substituting eqns.(5.13)-(5.16) in eqn.5.12 gives

$$\frac{\partial \varepsilon_q^{\,2}}{\partial W_{pq.k}} = -2\alpha[T - \Phi_{q.k}]\Phi_{q.k}[1 - \Phi_{q.k}]\Phi_{q.k}$$

$$= -\delta_{pq.k}\Phi_{p.j} \qquad \text{..... (5.17)}$$

Where $\delta_{pq.k}$ is defined as

$$\delta_{pq.k} = 2\alpha[T - \Phi_{q.k}]\Phi_{q.k}[1 - \Phi_{q.k}]$$

$$= 2\varepsilon_q \frac{\partial \Phi_{q.k}}{\partial I_{q.k}} \qquad \text{..... (5.18)}$$

Substituting eqn.5.17 in eqn.5.11 gives

$$\Delta W_{pq.k} = -\eta_{p.q} \frac{\partial \varepsilon_q^{\,2}}{\partial W_{pq.k}} = \eta_{p.q}\delta_{pq.k}\Phi_{p.j} \qquad \text{..... (5.19)}$$

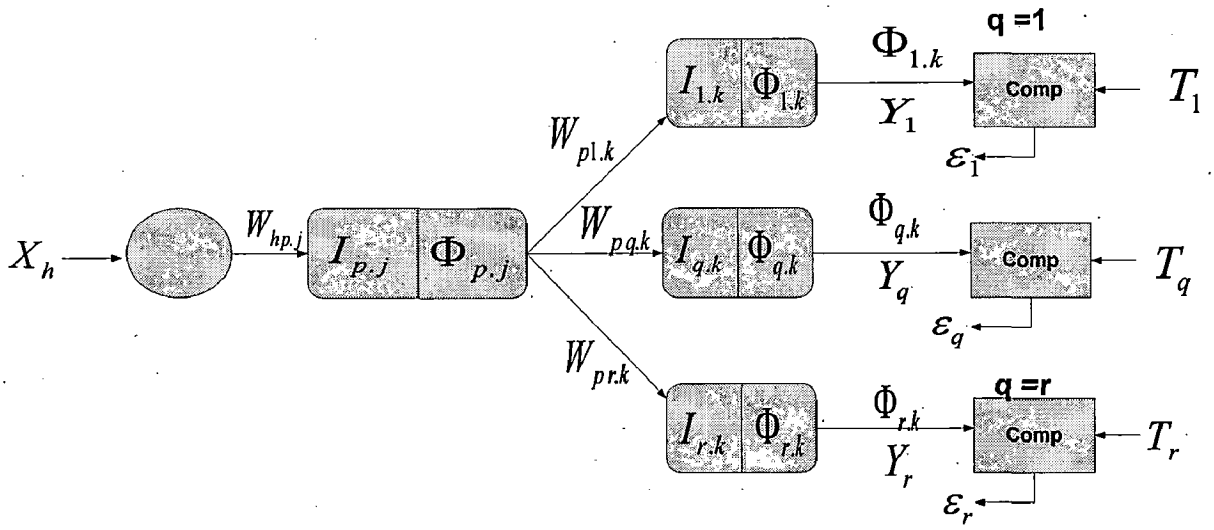
$$W_{pq.k}(N+1) = W_{pq.k}(N) + \eta_{p.q}\delta_{pq.k}\Phi_{p.j} \qquad \text{..... (5.20)}$$

where N is the iteration number. An identical process is performed for each weight of the output layer to give the adjusted values of the weights. The error term $\delta_{pq,k}$ is used to adjust the weights of the output-layer neurons using eqns.5.19 and 5.20. In eqn .5.19, we have calculated the error that has to be propagated back through the network. This error exists because of the wrong outputs generated by the output-layer neurons. This is due to their own incorrect weights and the fact that the middle-layer neurons generate the wrong output. To overcome this situation, we back propagate the errors for each output-layer neuron, using the same interconnections and weights the middle layer used to transmit its outputs to the output layer. When a weight between a middle-layer neuron and an output-layer neuron is large and the output-layer neuron has a very large

error, the weight of the middle-layer neuron may be assigned a very large error, even if that neuron has a very small output and, thus, could not have contributed much to the output error. By applying the derivative of the squashing function, this error is moderated, and only small-to-moderate changes are made to the middle-layer weights because of the bell-shaped curve of the derivative function.

5.5.2 Calculation of weights of hidden-layer neurons

When neuron j is located in the hidden layer of the network, there is no specified desired response for that neuron. The error signal for a hidden neuron has to be determined recursively in terms of the error signals of all the neurons to which the hidden neuron is directly connected. Since the hidden layers have no target vectors, the problem of adjusting the weights of the hidden layers has troubled researchers in this field for years until backpropagation was put forth. Back propagation trains hidden layers by propagating the adjusted error back through the network, layer by layer, adjusting the weight of each layer as it goes. The equations for the hidden layer are the same as for the output layer, except that the error term $\delta_{hp,j}$ must be generated without a target vector. We must compute $\delta_{hp,j}$ for each middle-layer neuron that includes contributions from the errors in each neuron in the output layer to which it is connected.



i th layer	*j* th layer	*k* th layer
index h	index p	index q
m nodes	n nodes	r nodes

Fig.5.12 Representation of a train of neurons for calculating the change of weight for a middle-(hidden) layer neuron in backpropagation

Let us consider a single neuron in the hidden layer just before the output layer, designated the subscript p. In the forward pass, this neuron propagates its output values to the q neurons in the output layer through the interconnecting weights $W_{pq,k}$. During training, these weights operate in reverse order, passing the value of $\delta_{pq,k}$ from the output layer back to the hidden layer. Each of these weights is multiplied by the value of the neuron which is connected to the output layer. Summing all such products the value of $\delta_{hp,j}$ needed for the hidden-layer neuron.

The arrangement in Fig.5.12 shows the errors that are backpropagated to produce the change in $W_{hp,j}$. Since all error terms of the output layer are involved, the partial derivative involves a summation over all the r outputs. The procedure for calculating $\delta_{hp,j}$ is substantially the same as that for calculating $\delta_{pq,k}$. Let us start with the derivative of the squared error with respect to the weight for the middle layer that is to be adjusted. Then, in a manner analogous to eqn.5.11, delta rule training gives

$$\Delta W_{hp.j} = -\eta_{h.p} \frac{\partial \varepsilon^2}{\partial W_{hp.j}} = -\eta_{h.p} \sum_{q=1}^{r} \frac{\partial \varepsilon_q^{\,2}}{\partial W_{hp.j}} \qquad \text{..... (5.21)}$$

where the total mean squared error ε^2 is now defined as

$$\varepsilon^2 = \sum_{q=1}^{r} \varepsilon_q^{\,2} = \sum [T_q - \Phi_{q.k}]^2 \qquad \text{..... (5.22)}$$

since several output errors may be involved. The learning rate $\eta_{h,p}$ is usually, but not necessarily, equal to $\eta_{p.p}$.

Again, we can evaluate the last term of eqn.5.21 using the chain rule of differentiation, which gives

$$\frac{\partial \varepsilon^2}{\partial W_{hp.j}} = \sum \frac{\partial \varepsilon_q^{\,2}}{\partial \Phi_{q.k}} \frac{\partial \Phi_{q.k}}{\partial I_{q.k}} \frac{\partial I_{q.k}}{\partial \Phi_{p.j}} \frac{\partial \Phi_{p.j}}{\partial I_{p.j}} \frac{\partial I_{p.j}}{\partial W_{hp.j}} \qquad \text{..... (5.23)}$$

where

$$\frac{\partial \varepsilon_q^{\,2}}{\partial \Phi_{q.k}} = -2(T_1 - \Phi_{q.k}) = -2\varepsilon_q \qquad \text{..... (5.24)}$$

$$\frac{\partial \Phi_{q.k}}{\partial I_{q.k}} = \alpha \Phi_{q.k}(1 - \Phi_{q.k}) \qquad \text{..... (5.25)}$$

and

$$I_{q.k} = \sum_{p=1}^{n} W_{pq.k} \Phi_{p.j} \qquad \text{..... (5.26)}$$

Taking the partial derivative of eqn.5.26 with respect to $\Phi_{p.j}$ gives

$$\frac{\partial I_{q.k}}{\partial \Phi_{q.k}} = W_{pq.k} \qquad \text{..... (5.27)}$$

The summation over p disappears because only one connection is involved. Changing the subscripts in eqn.5.14 to correspond to the middle layer gives.

$$\frac{\partial \Phi_{p.j}}{\partial I_{p.j}} = \alpha \Phi_{p.j}(1 - \Phi_{p.j}) \qquad \text{..... (5.28)}$$

Changing the subscripts in eqn.5.26 and substituting the i^{th} –layer input x_h for the j^{th} – layer input $\Phi_{p.j}$ gives

$$I_{p.j} = \sum_{h=1}^{m} W_{hp.j} X_h \qquad \text{..... (5.29)}$$

Taking the partial derivative of eqn.5.29 gives

$$\frac{\partial I_{p.j}}{\partial W_{hp.j}} = X_h \qquad \text{..... (5.30)}$$

Again, the summation over h in eqn.5.29 disappears because only one connection is involved. Substituting eqns.(5.24)-(5.30) in eqn.5.23 gives

$$\frac{\partial \varepsilon^2}{\partial W_{hp.j}} = \sum_{q=1}^{r} (-2)\alpha(T_q - \Phi_{q.k})[\Phi_{q.k}(1 - \Phi_{q.k})] W_{pq.k} \alpha[\Phi_{p.j}(1 - \Phi_{p.j})] X_h$$

$$= \sum_{q=1}^{r} \delta_{pq.k} W_{pq.k} \frac{\partial \Phi_{p.j}}{\partial I_{p.j}} X_h \qquad \text{..... (5.31)}$$

$\delta_{hp.j}$ can be defined as

$$\delta_{hp.j} = \delta_{pq.k} W_{pq.k} \frac{\partial \Phi_{p.j}}{\partial I_{p.j}} \qquad \text{..... (5.32)}$$

eqn.5.32 becomes

$$\frac{\partial \varepsilon^2}{\partial W_{hp.j}} = -\sum_{q=1}^{r} \delta_{hp.j} X_h \qquad \text{..... (5.33)}$$

Since the change in weight given in eqn.5.21 is proportional to the negative of the rate of change of the squared error with respect to that weight, the substitution of eqns.5.32 and 5.33 in eqn.5.21 gives

$$\Delta W_{hp.j} = -\eta_{h.p} \frac{\partial \varepsilon^2}{\partial W_{hp.j}} = \eta_{h.p} \sum_{q=1}^{r} \delta_{pq.k} W_{pq.k} \frac{\partial \Phi_{p.j}}{\partial I_{p.j}} X_h$$

$$= \eta_{h.p} X_h \sum_{q=1}^{r} \delta_{hp.j} \qquad \text{..... (5.34)}$$

and hence

$$W_{hp.j}(N+1) = W_{hp.j}(N) + \eta_{h.p} X_h \sum_{q=1}^{r} \delta_{hp.j} \qquad \text{..... (5.35)}$$

If there is more than one middle layer of neurons, this process moves through the network, layer by layer, to the input layer, adjusting the weights as it goes. When finished, a new training input is applied and the whole process starts again. It continues until an acceptable error is obtained. At that point the network is said to be trained.

5.6. TRAINING ALGORITHMS

There are two different ways in which this gradient descent algorithm can be implemented: incremental training and batch training.

5.6.1 Incremental training.

In the incremental mode, the gradient is computed and the weights are updated after each input is applied to the network. Incremental training is sometimes referred to as online or adaptive training.

5.6.2 Batch training.

In batch mode the weights and biases of the network are updated only after the entire training set has been applied to the network. The gradients calculated at each training example are added together to determine the change in the weights and biases. Neural network have been trained to perform complex functions in various fields of application including pattern recognition, identification, classification, speech, vision and control systems.

Chapter – 6

SOFTWARE IMPLEMENTATION

6.1 Introduction

This chapter explains the design and operation of the speech recognition program. The program is divided into three main processes, namely the extraction, coding and the classification process. The scheme which is used for the speaker dependent isolated spoken Hindi word recognition is given below in Fig.6.1.

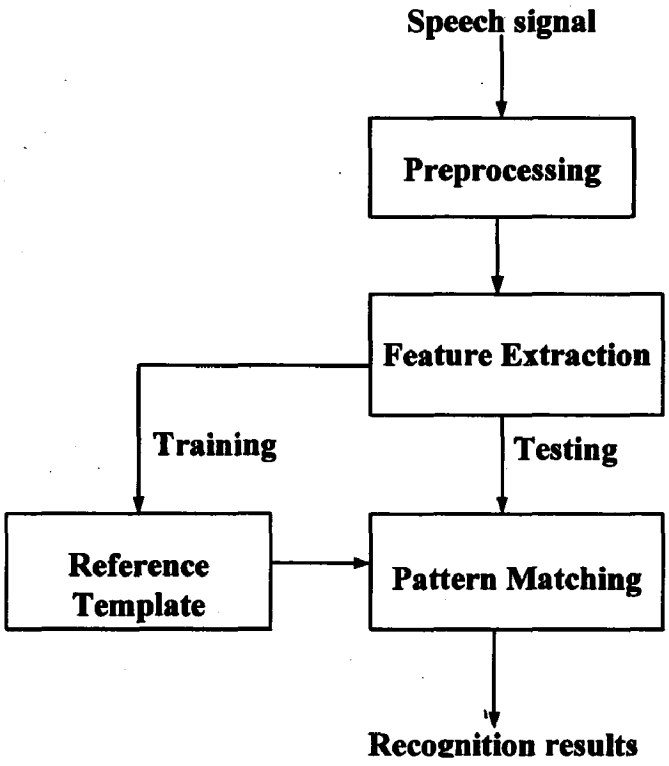


Fig.6.1 Block diagram of the scheme

These speech samples are stored as *.mat files using the Matlab wavrecord function. In order to extract the features of voices, first the voice is isolated using word isolation algorithm this is necessary because the extraction of features is wholly dependent on the samples we are working on. Once the speech is isolated the weighted Mel-frequency cepstrum analysis is done for extracting features. These features act as training set for network. During the training phase network was trained finally, the trained network has been saved in database.. For each word in the vocabulary sixteen samples are taken from one single speaker. Thus, corresponding to 64 words there are 1024 samples are recorded. Once the features are extracted they can be used for matching

with the features of the other words, which are to be recognized. In the recognition phase word has been recorded and features are extracted and given as input to the network, the network output is the word to be recognized. How the above scheme is implemented in this dissertation is shown in the following paragraphs.

The software for speaker dependent isolated word speech recognition was developed in Matlab 6.5. The software is menu driven. As the program runs the menu which appears is shown in Fig.6.2.

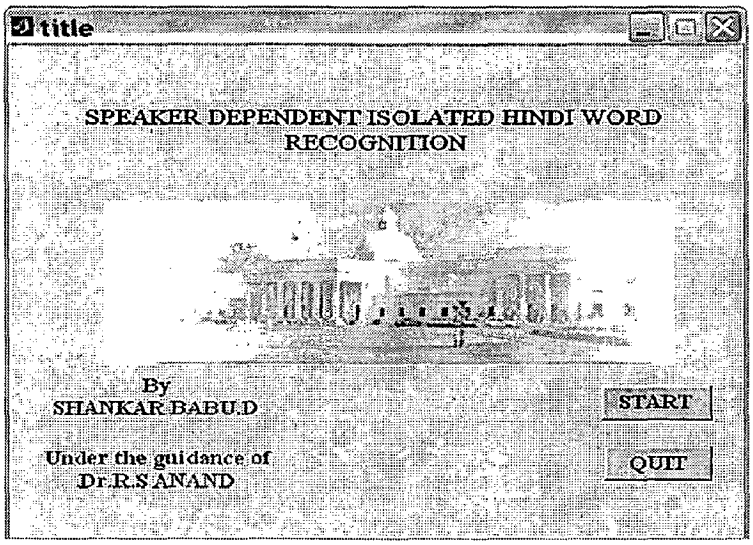


Fig.6.2 Menu of the title page

It consists of two buttons: start button, Testing quit button, if we click start button the next menu will appear It consists of two main modes: Training mode, Testing mode.

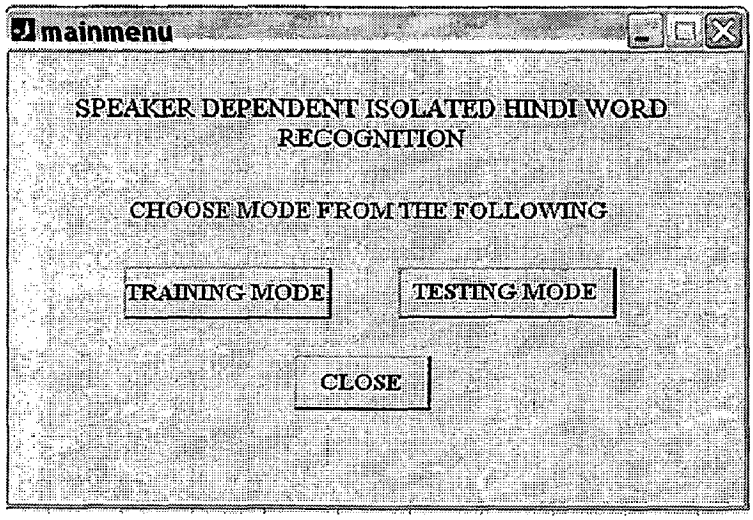


Fig.6.3 Menu of the main program

6.2 Training mode: The data consists of those words which are in the vocabulary and these words are spoken 16 times to take into account the dynamic aspects of speech. It consists of three functions; one is for recording the speech signal and second one is for extracting the features from signal, third one is for training he network shown in Fig.6.4.

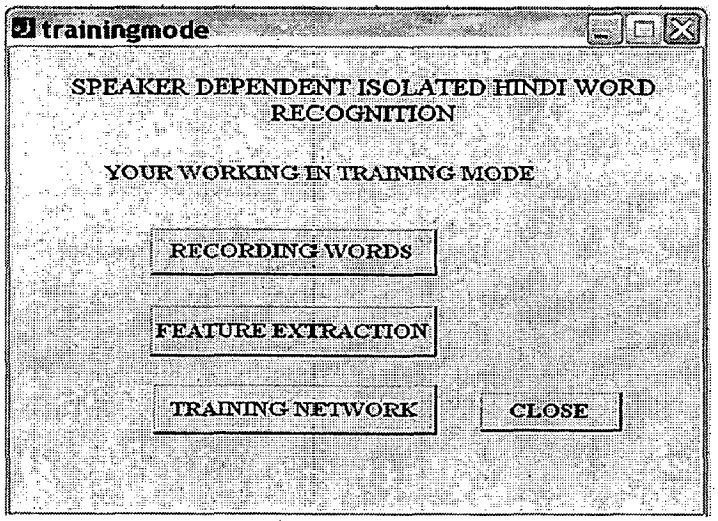


Fig.6.4 Menu of training mode

When the recording words button is pressed it invokes another menu as shown in Fig.6.5. A record button is used to record the word. The word should be spoken when a beep sound is heard from the microphone. Thus the numbers of words that are in the vocabulary are recorded by this means.

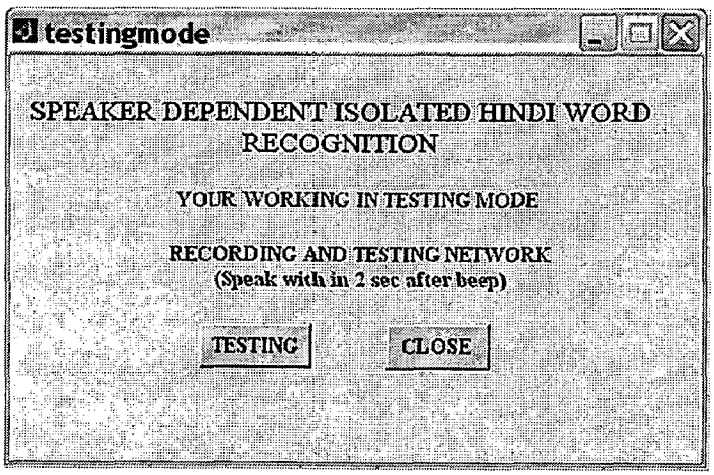


Fig.6.5 Menu for recording the words

When the feature extraction button is pressed it goes into feature extraction process, when training network button is pressed, the network is trained as shown in Fig.6.6 and produces a message of wait for the user to wait till the training is over.

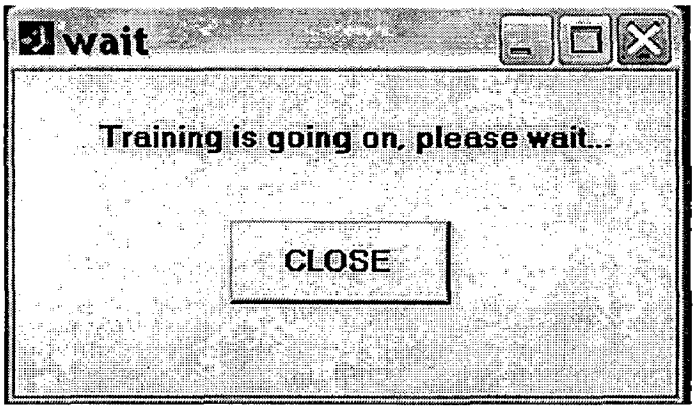


Fig.6.6 Wait message while training is going on

6.3 Testing mode: Testing data is the data i.e. the words which are to be recognized for testing. Once the training is over the testing button can be invoked for recognizing words.

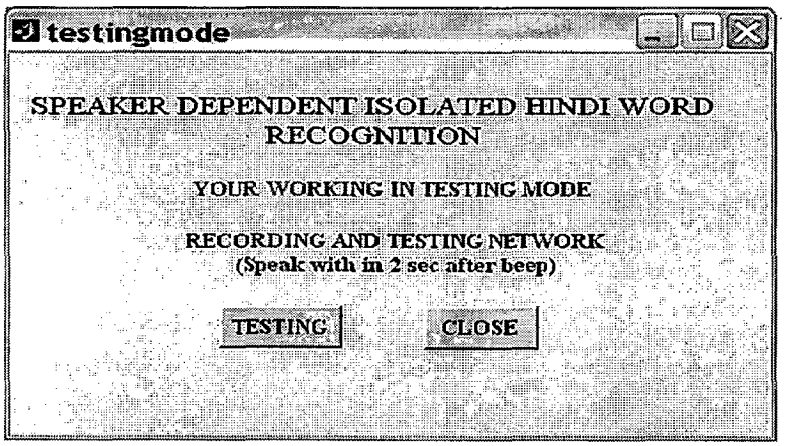


Fig.6.7 Menu of Testing mode

Fig.6.7 shows the Testing function menu. It has two buttons: first, record a new word for recognizing, the word should be present in the vocabulary, and if it is present in the vocabulary then it will display that word in the command window of the Matlab. Second, pick an already existing word from the vocabulary, so that it will display it.

Chapter – 7

RESULTS AND DISCUSSIONS

7.1 Obtaining speech waveform

The work that has been done is carried out on a Pentium IV PC 2.0 GHz with a 8-bit sound card. The software for speaker dependent isolated Hindi word recognition is developed using Matlab 6.5. All the digital signal processing [20] and implementation were carried using Matlab signal processing toolbox. A microphone and a sound card including a PC are used for collecting these samples. These speech samples are approximately of 2 sec and stored as *.mat files using the Matlab wavrecord function. The wavrecord function is given as

Y = wavrecord (n*fs, fs, 'double').

where, n is the number of seconds

fs is the sampling frequency

and 'double' is the number of bits per sample

waveform of recorded speech are shown as below.

The training set was obtained by recording utterances of a set of Hindi words. The recordings were done for a single male speaker. The recognition vocabulary consists of 64 Hindi words taken randomly from a Hindi dictionary. These 16 different words were spoken 16 times each. Thus, in total there are 1024 samples. The waveform of the word is shown below in Fig.7.1.

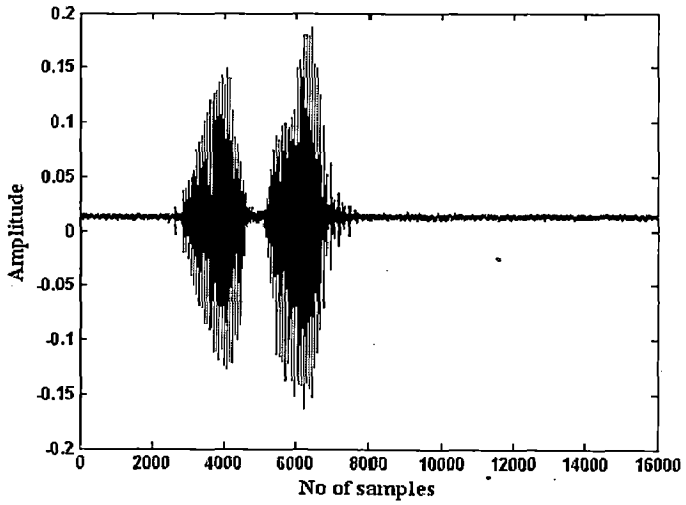


Fig.7.1 Wave form of word "Raja"

7.2 Pre-extraction process

The speech wave filed is loaded into the Matlab program by using a "wavrecord" function which limits the amplitude of the speech signal to a magnitude of 1. The signal is then saved as an M x 1 vector where M refers to the total number of samples in the speech signal. Each element in the M vector contains the amplitude of the speech signal at a particular sampling instant.

7.2.1 Quality Process

Before the actual extraction process takes place, the wave file is subjected to a series of processes to ensure the compatibility and quality of the signal. When the speech signal is being loaded into the Matlab program, the signal is not centered at the y = 0 axis. In order to bring the whole signal to centre on the zero-line, special program code was written. This code is used to find the mean of the signal and then subtract this mean from each of the sample values of the signal. This is shown in Fig.7.2 (a) and 7.2(b) below.

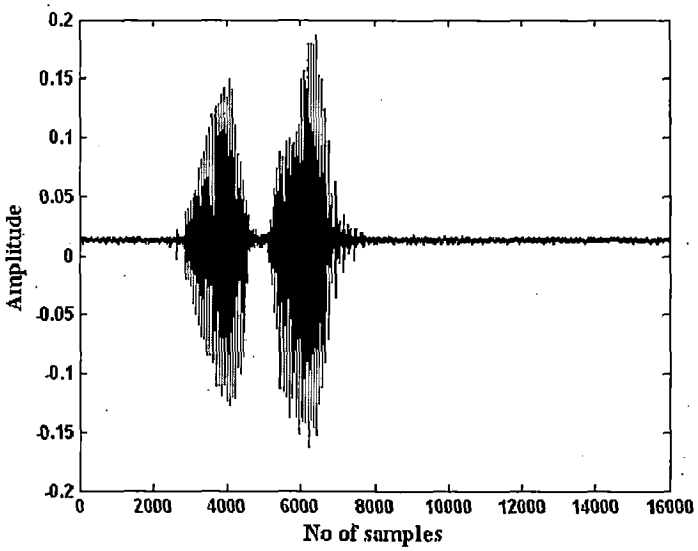
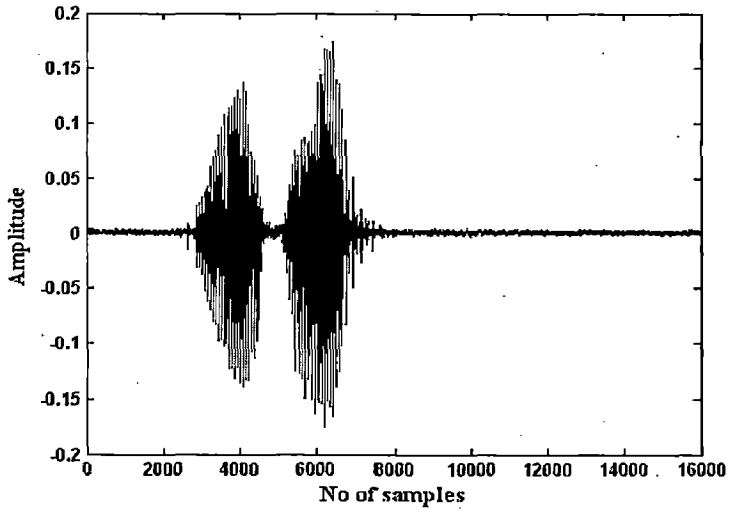
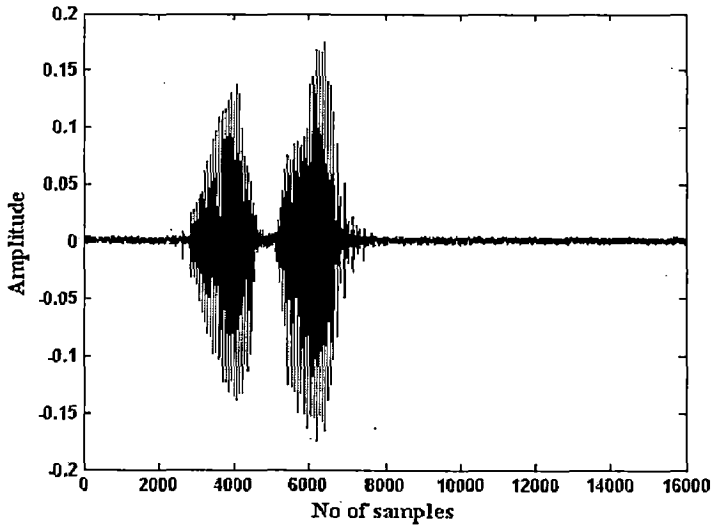


Fig.7.2 (a) Speech waveform before entering process



7.2(b) Speech waveform after entering process

The next process is to suppress the noise present in the speech waveform. Although the sound recorder program has performed the initial filtering, some noise is still present in the speech waveform. Another section of the Matlab program code is used to set a threshold value on the speech signal. Any value of the speech signal that falls below this threshold value will be set to zero. This will greatly suppress the unwanted noise and in the meantime preserve the content of the main speech signal. This is illustrated in Fig. 7.2(c) and Fig.7.2 (d) below. After some testing, it was found that a threshold value of 0.02 is most suitable.



7.2(c) Speech waveform before filtering

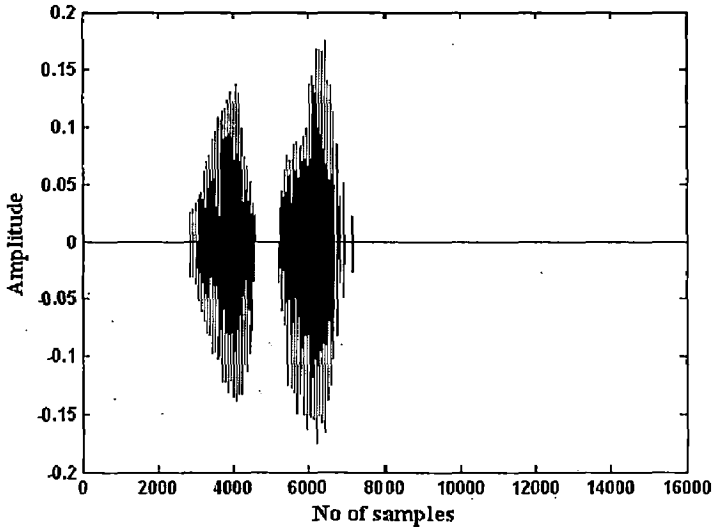


Fig.7.2 (d) **Speech waveform after filtering**

The final compatibility and quality process is to determine the area of interest of the speech signal. This is done by detecting the first rise point and the final drop point of the speech waveform. This can be done easily since the speech signal is cleared of unwanted noise. Hence area of interest of the speech lies between the first rise and final drop point of the speech waveform. This area of interest is later used for the extraction and coding processes.

In order to extract the features of voices, first the voice is isolated using word isolation algorithm this is necessary because the extraction of features is wholly dependent on the samples we are working on.

7.2.2 Endpoint detection

An important problem in speech processing is to detect the presence of speech in a background of noise. This problem is often referred to as the endpoint location problem [11]. The accurate detection of a word's start and end points means that subsequent processing of the data can be kept to a minimum. In many cases the accuracy of alignment depends on the accuracy of the endpoint detections. In order to perform well, the algorithm must take a number of special situations into account such as:

- Words which begin or end with low-energy phonemes (weak fricatives).
- Words which end with an unvoiced plosive.
- Words which end with a nasal.

- Speakers ending words with a trailing off in intensity or a short breath (noise).

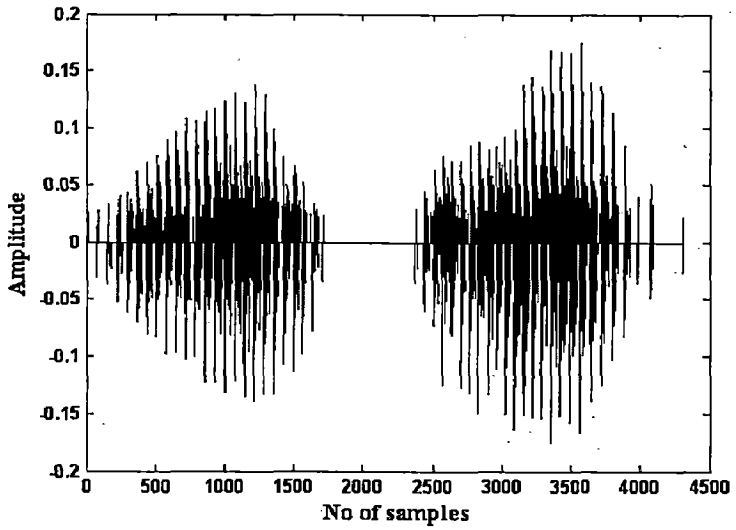


Fig.7.2 (e) Wave form after end point detection

7.3 Blocking and windowing

To extract the short-time features of a speech signal, the speech signal should be blocked into short segments called frame. The duration of each frame varies from 10 to 30 ms. Hence, the speech data was processed in 23.2 msec (corresponding to 256 points for a sampling frequency of 8000) frames. The speech belonging to each frame is assumed to be stationary. To reduce the edge effect of each segment, a smoothing window (e.g. Hamming window) is applied to each frame. A Hamming window is used because the side lobes of this window are much lower than the rectangular window (i.e. the leakage effect is decreased) although resolution is appreciably reduced. To obtain a more smooth feature set over time the successive frames are overlapped over each other by 11.6msec (i.e. 156 points). The waveform of a single frame is shown in Fig.7.4.

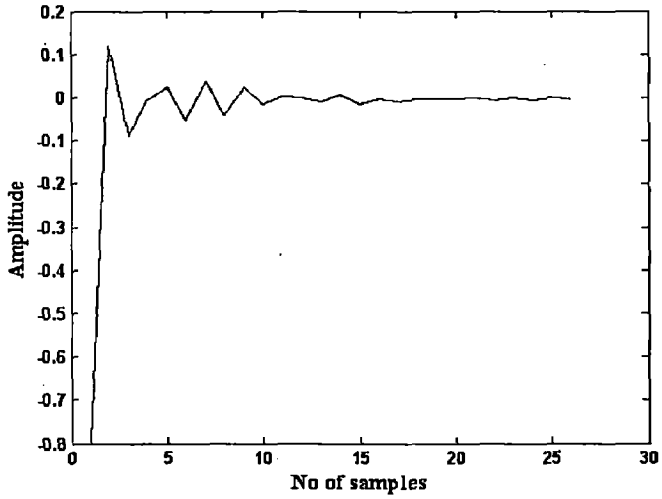


Fig.7.2 (f) The waveform of a single frame

Fig.7.2 Waveforms of word "Raja"

7.4 Feature extraction process

After blocking and windowing the speech features are extracted from them. The approach used here is Mel-frequency cepstral analysis.

The following steps compute the Mel-frequency cepstral coefficients:

1) Window the speech frame, using a Hamming (or other) window.

2) Do zero padding to achieve a frame length suitable for an FFT.

3) Do the FFT.

4) Find the Power Spectrum.

5) From this data, the mel scale was constructed.

The mel scale is approximately linear below 1 kHz and logarithmic above. This is approximated using the following formula.

$$Mel(f) = 1127 \log\left(1 + \frac{f}{700}\right)$$

5) Take the Log of the Power Spectrum. (This is the cepstrum.)

6) Inverse FFT. (The results are the cepstral coefficients.)

In practice the last step of taking inverse DFT is replaced by taking *discrete cosine transform (DCT)* for computational efficiency. Finally, the mel-weighted cepstral

coefficients are calculated from the log filter bank amplitudes, denoted m_j, using the Discrete Cosine Transform

$$c_i = \sqrt{\frac{2}{N}} \sum_{j=1}^{N} m_j \cos\left(\frac{\prod i}{N}(j-0.5)\right), i = 1..$$ number of cepstral coefficients

where N is the number of filter bank channels

After finding out the starting and end point of the words there corresponding waveform is shown in Fig.7.2 (e). These isolated words are then used for generating feature vectors. The normalized feature vectors of this waveform are shown in the Table7.1.

Table 7.1: Feature vector of word "Raja"

Columns 1 through 8

-0.7561	-0.2932	-0.2456	0.0626	0.114	0.2014	0.2922	0.3416
0.1645	0.0922	0.0739	-0.0386	-0.1167	-0.1292	-0.2204	-0.2627
0.0241	-0.0568	-0.1004	-0.1842	-0.2873	-0.276	-0.266	-0.2898
-0.0052	-0.0131	-0.0706	-0.0042	0.06	0.1348	0.1445	0.089
0.0387	0.0178	0.0004	0.0047	-0.029	-0.0181	-0.0056	-0.0364
0.0891	0.0559	0.0405	0.0395	0.0261	-0.0012	-0.0328	-0.0566
0.0198	0.0055	0.0491	-0.0207	-0.0747	-0.074	-0.0727	-0.0667
-0.0882	-0.1259	-0.1427	-0.1039	-0.0572	-0.0164	-0.0125	0.0144
0.0222	0.0439	0.0578	0.0674	0.0542	0.0679	0.0554	0.0591
-0.0304	0.0049	-0.0042	0.0397	0.0328	-0.0092	-0.0385	-0.0691
0.0059	0.0245	-0.0145	-0.0355	-0.057	-0.05	-0.0088	0.0214
0.0324	0.0241	0.0085	-0.0009	0.0139	0.0385	0.0482	0.0355
0.0167	-0.0272	-0.019	-0.025	-0.0044	-0.045	-0.0408	-0.0308
0.0619	0.0629	0.0355	0.0357	0.0447	0.0115	-0.0085	0.0138
0.0244	0.0136	0.0359	-0.0161	0.0338	0.0436	0.0619	0.0563
0.0215	0.0105	0.0291	0.009	0.0466	0.0046	-0.0021	0.0163
-0.015	-0.0055	0.041	0.0622	0.1056	0.0726	0.0915	0.1068
0.0174	0.0412	0.0745	0.058	0.0842	0.0577	0.0758	0.0872
0.0123	0.0354	0.0558	0.0594	0.119	0.0983	0.0716	0.0672
0.0105	0.0339	0.0615	0.0532	0.0946	0.0684	0.0879	0.1111
-0.0325	0.0163	0.0479	0.0544	0.0622	0.0281	0.0155	0.0333
-0.0079	0.025	0.0462	-0.0046	-0.0028	-0.0178	-0.0186	-0.017
0.0104	0.0208	0.0307	0.003	0.0082	-0.0338	-0.036	-0.0239
0.0002	-0.0015	0.0072	0.0143	0.0068	0	0.01	0.0025
0.0052	0.0049	-0.0197	0.0018	0.0039	0.0175	0.0401	0.0272
0.0112	0.014	-0.0362	-0.0371	-0.0083	-0.0035	0.0174	0.0306

Columns 9 through 16

-0.0376	-0.0139	-0.0993	-0.2626	-0.729	-0.9208	-1	-0.8842
-0.156	-0.1417	-0.1107	0.0299	0.1022	0.1283	0.1615	0.0667
-0.2431	-0.2151	-0.1816	-0.1217	-0.0743	-0.0714	-0.0432	0.0138
0.064	0.0227	-0.0248	-0.1206	-0.0277	0.0596	0.0683	-0.0004
0.0659	0.0799	0.0903	-0.0015	0.0147	0.0691	0.0873	0.0245
-0.0516	0.0049	0.11	0.1413	0.0809	0.0505	0.0308	0.0725
-0.0119	0.0094	-0.0388	-0.0291	0.013	0.0038	-0.0076	0.0363
-0.0187	-0.0588	-0.0802	-0.021	0.0065	-0.0053	0.0118	0.0635
0.128	0.1315	0.0957	0.0525	0.0828	0.1085	0.1087	0.0829
-0.0135	0.0038	-0.0065	-0.019	-0.0002	0.0107	0.0454	0.021
-0.0302	-0.0084	-0.0007	0.034	0.0271	0.0474	0.0614	0.0366
0.0499	0.0129	0.0276	0.0299	0.0643	0.0614	0.0133	0.0161
-0.0028	-0.0023	0.0098	0.004	0.0477	0.0116	0.0244	0.0151
0.0212	0.0243	0.0401	0.04	0.0104	-0.0241	0.0182	0.0138
0.0273	-0.0244	-0.0037	0.0253	-0.0036	-0.014	-0.0097	0.0262
0.0329	0.0485	0.0658	0.057	0.072	0.0781	0.0353	0.0411
0.0494	0.0736	0.0665	0.0711	0.068	0.0864	0.0608	0.0706
0.0854	0.0406	0.0369	0.0569	0.0357	0.0515	0.0408	0.0489
0.071	0.0445	0.0684	0.0387	0.0321	0.0617	0.0654	0.0732
0.0738	0.0501	0.0596	0.0456	0.0692	0.0623	0.0313	0.0555
0.0373	0.0219	0.0135	0.0262	0.0525	0.0282	0.036	0.0334
0.0158	-0.0161	-0.0137	0.0186	0.0151	0.0126	-0.0029	-0.0216
0.008	-0.0099	-0.0133	0.0052	0.011	-0.0073	-0.0211	-0.0353
-0.009	0.0125	0.0188	-0.0179	-0.0214	-0.0394	0.0154	0.0105
-0.0082	-0.0187	0.0008	-0.0068	0.0066	0.0066	-0.0059	-0.0034
-0.0119	-0.0289	0.0168	0.0345	-0.0024	0.0124	-0.0135	-0.0008

Columns 9 through 16

-0.5996	-0.2437	-0.0329	0.0211	0.0763	0.008	-0.0261	-0.0496
0.1028	0.0574	-0.022	-0.0581	-0.1206	-0.1804	-0.2003	-0.1843
0.0387	-0.0346	-0.0759	-0.1341	-0.2168	-0.2385	-0.2637	-0.2634
-0.1278	-0.1607	-0.1402	-0.0932	-0.0613	0.008	0.0427	0.0603
-0.0381	-0.0211	-0.0019	0.0127	0.03	0.1032	0.1051	0.11
0.0641	0.0869	0.1335	0.1226	0.0532	0.0458	0.0025	-0.0172
0.0304	0.0384	0.0191	-0.0344	-0.0531	-0.0563	-0.0339	-0.016
0.0897	0.071	-0.0161	-0.0186	-0.0525	-0.0564	-0.0367	-0.0765
0.0324	-0.01	-0.0334	0.0201	0.0938	0.1303	0.1407	0.143
-0.0234	-0.0293	-0.03	0.0005	-0.0196	-0.0429	-0.0419	-0.0257
0.0451	0.0467	0.0561	0.0678	0.0234	-0.0118	-0.0599	-0.0751
0.0023	0.006	-0.0072	0.0024	0.0053	0.0217	0.0029	0.0289
0.0648	0.0649	0.0258	0.0215	0.0168	0.0185	0.0254	0.0377
0.0162	-0.0185	-0.0216	0.0514	0.0852	0.1176	0.1349	0.1054
0.0133	0.0362	0.0468	0.0286	0.0433	0.0964	0.1254	0.1376
0.0211	0.0641	0.0701	0.0452	0.0646	0.0926	0.1176	0.129
0.0976	0.097	0.0642	0.0552	0.0448	0.07	0.0854	0.0715

0.0677	0.0415	0.0069	0.0399	0.061	0.0364	0.0292	0.0312
0.051	-0.0238	-0.0801	0.0009	0.0308	0.0067	-0.0068	-0.0091
0.0008	-0.0122	-0.055	-0.032	-0.019	-0.0711	-0.0674	-0.0382
0.0021	-0.0201	-0.0578	-0.0372	-0.0164	-0.0487	-0.0705	-0.0715
-0.0238	-0.0135	-0.0106	0.0046	0.0119	0.0074	-0.0086	0.008
0.007	0.0143	0.0212	0.036	0.0395	0.0523	0.0818	0.0854
0.0119	0.0247	0.0264	0.0306	0.0459	0.0947	0.1085	0.0809
0.0107	0.005	-0.0066	-0.01	-0.0001	-0.0021	0.0026	0.0229
0.0106	-0.0059	-0.0047	-0.0313	-0.0589	-0.051	-0.034	-0.0351

Columns 25 through 30

-0.0602	-0.1436	-0.2631	-0.4563	-0.6827	-0.7754
-0.17	-0.1255	-0.075	-0.0238	0.0629	0.1716
-0.3005	-0.2801	-0.2599	-0.2141	-0.1197	-0.079
0.0975	0.1298	0.1249	0.0959	0.0888	0.0749
0.096	0.07	0.0569	0.0447	0.0421	0.069
-0.0672	-0.0848	-0.0831	-0.0565	-0.0169	-0.0209
0.0588	0.0865	0.1011	0.0841	0.059	0.0445
-0.0691	-0.0643	-0.0663	-0.0567	-0.0368	-0.0143
0.1113	0.0687	0.0627	0.0376	0.0465	0.0475
-0.0051	-0.005	0.011	0.0369	0.0224	-0.0159
-0.0283	0.0021	0.025	0.0139	-0.0059	-0.0243
0.0349	0.021	0.0199	0.0237	0.0537	0.0324
0.051	0.052	0.0465	0.0522	0.0588	0.0282
0.0946	0.094	0.0812	0.081	0.053	0.0418
0.1114	0.1043	0.1088	0.1341	0.1108	0.047
0.0837	0.0876	0.1026	0.1252	0.1374	0.063
0.0591	0.0533	0.0553	0.0788	0.0748	0.0362
0.0223	0.0421	0.049	0.0494	0.0139	0.037
-0.0334	-0.0373	-0.0548	-0.0551	-0.0445	0.0546
-0.0323	-0.0417	-0.0565	-0.0636	-0.0524	0.0302
-0.0761	-0.032	-0.0171	-0.0165	-0.0224	0.0282
-0.0095	-0.0056	-0.0028	0.0111	0.0089	0.007
0.0573	0.0486	0.0405	0.059	0.0598	0.028
0.0434	0.0533	0.0593	0.0677	0.053	0.0122
0.0136	0.0082	-0.0074	-0.0083	-0.0179	-0.0145
-0.0447	-0.041	-0.0346	-0.0319	-0.0256	-0.0079

7.5 Classification using MFCC and neural network

After extracting the features, these features act as templates. During the training phase templates of the words are given to the neural network as input to train the network. The templates generated are through the scheme depicted above in Fig.6.1. For each word in the vocabulary sixteen samples are taken and the number of frames in each

word depends upon the length of word. Thus, corresponding to single word there are seven hundred and eighty samples are extracted.

The MFCC matrices are ideally matched to the processing requirements of Neural Networks for which the use of fixed sized training and interrogation vectors is typically essential [22]. In order to bring the matrix to fixed size, special program code was written. This code is used to set a threshold value on the no of samples of the speech signal. If the speech samples that falls below this threshold value will be set to add some samples to reach threshold this is known as up-sampling, or if the no of samples falls above this threshold will be set to subtract some samples, this is known as down-sampling. Fig.7.4 shows an overview of the classification process using a MFCC S-Matrix in a Neural Network.

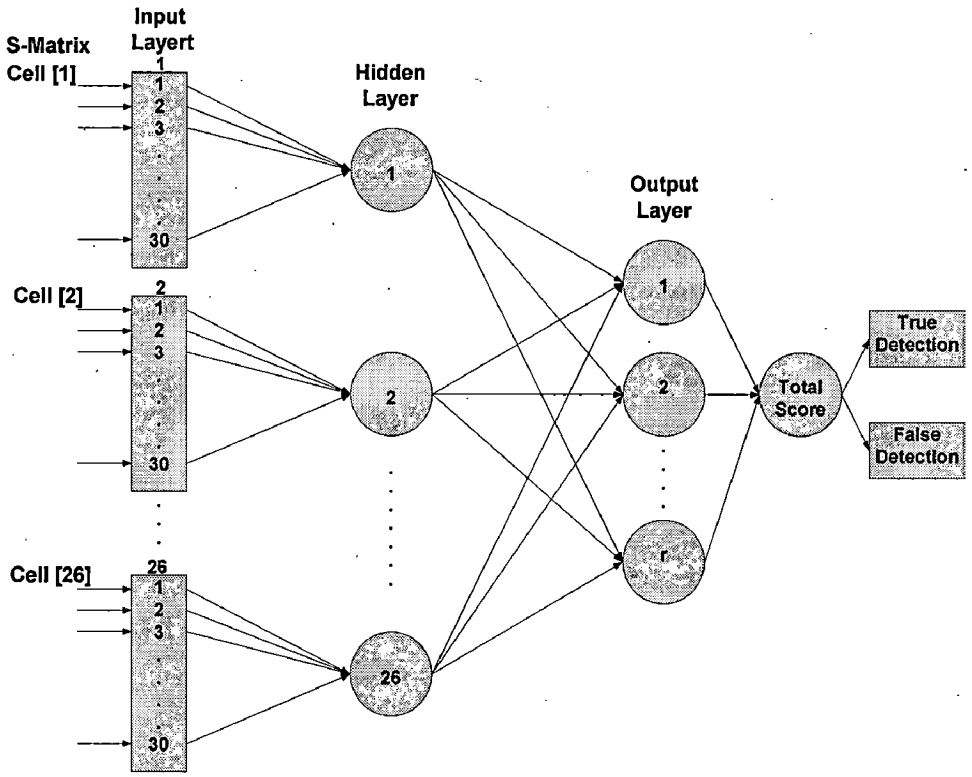


Fig.7.4 Multi-Layer perceptron

7.5.1 Training phase

During training phase, the words are recorded as wave files then loaded in to the Matlab program and goes through feature extraction process and then, features are converted to an S-Matrix (Reference Matrix) and stored in to the database. For each word

there are 16 speech samples are taken. Similarly the same process has been done for 64 words. This process is necessary to increases the accuracy of recognition.

After the S-matrix of each individual word is obtained, the elements in the matrix are then passed through the Neural Network to construct the weight and bias for each individual word. Finally the network was stored in database. The network is shown in Fig.7.4 and the values weights are in table 7.2 and 7.3.

Table:7.2 The weights of input hidden layer Whp=Columns 1 through 9

1.4226	-10.18	9.4651	-3.6589	-8.406	3.8622	2.0387	16.671	-5.4754
8.4432	9.6794	-2.2093	-5.1366	9.0023	-11.19	-1.6677	8.0495	-4.4972
-9.1212	4.4324	-4.0314	-0.4966	13.86	12.817	8.6948	-7.913	-3.4661
-1.5266	15.314	14.752	1.9454	1.6964	9.4697	-0.4735	-10.234	0.32061
1.6925	-9.0035	-1.208	-0.978	-4.1805	3.802	-1.7772	4.1812	-2.0892
-0.7778	1.7696	1.7073	-0.1198	3.5273	9.1109	4.3016	-3.7905	4.3379
5.4275	5.0852	-3.5432	-3.9175	-7.8564	18.601	3.3316	9.5018	7.999
6.3099	-22.268	-3.3512	-3.3069	-10.44	2.077	-2.7553	7.1467	7.8277
5.3741	13.003	-2.7395	6.1174	9.8785	0.85807	-6.6719	-3.3742	8.3092
5.7291	14.11	5.623	-18.954	13.67	7.3448	8.3289	-10.255	11.829
2.0324	-8.3889	-2.548	4.6662	5.8542	-4.3796	7.6256	1.2391	-3.0104
1.0015	7.5478	-1.8563	3.5202	5.1106	-6.1463	3.747	11.204	-6.0284
3.1971	5.5402	0.9486	14.389	-6.308	-2.1413	-4.3302	16.791	-6.2528
3.319	-7.5299	-4.3662	-4.0664	-11.039	-5.2492	1.9644	-3.954	6.2281
2.0297	3.852	9.2181	-3.7768	-9.1067	-10.489	6.9685	-3.9018	3.0286
-1.5156	4.85	-10.176	7.5239	5.0796	-0.4886	3.7938	-0.5992	-1.9832
1.6492	-16.14	1.1182	5.5684	3.308	6.3621	14.965	4.9084	5.9783
4.4993	-1.3762	-10.85	15.552	7.0078	1.7011	7.3741	7.8168	-6.9735
2.1171	-1.9403	4.2247	-0.749	5.036	4.4284	3.0279	5.6917	2.4581
-5.0666	2.9286	16.821	2.6557	11.623	6.7462	6.2418	-6.6886	8.7417
1.7626	8.8148	-7.3073	-1.7654	-3.7	-2.4266	-4.0933	-9.0851	7.2346
3.116	-3.4548	-5.6722	4.9301	7.0054	8.7751	-12.033	1.9183	5.0654
3.4745	-11.129	-5.983	-2.2427	8.4575	-4.4192	-15.304	-7.4047	-3.0491
3.3567	2.5705	-6.5813	-10.364	4.2512	-4.4442	-3.6009	7.2661	-2.1989
-4.4338	5.2714	-10.5	7.4119	-3.4804	1.5058	-7.5342	-8.6945	3.04
-5.6149	1.5765	6.9352	-5.1277	-15.852	11.776	-11.339	-9.5267	6.148
-4.9991	1.1892	16.343	13.94	-8.6383	7.2817	-0.5103	3.3783	8.3684
-4.8868	-7.3517	14.586	7.4963	-1.7585	5.2447	6.2771	2.5789	-1.8123
-2.988	9.2875	4.7054	-1.2346	1.1473	1.6076	-3.3296	8.1075	-3.5438
-4.3079	5.2685	-14.185	-8.6209	9.0562	3.4429	2.3202	5.3093	2.2586

Columns 10 through 19

4.2843	2.898	3.3862	-5.425	-1.4282	2.7785	-2.7484	-3.388	-4.8284
14.509	4.1695	0.18437	0.56129	1.056	5.9237	1.8281	-2.1086	0.55084
8.449	2.8216	-3.03	-3.1294	1.7482	-1.1849	0.98948	2.0605	-0.3961
4.2625	-0.1962	0.67436	-0.0238	5.0684	-0.7945	0.10835	-3.6549	8.6287
4.4523	-3.0934	-3.566	-0.3357	8.2734	1.7766	1.5149	-10.498	5.3066
-16.556	-3.797	-4.5579	-1.0088	-2.9007	6.6919	4.2366	-8.07	13.006
-13.54	-0.2003	-3.968	-8.6766	-10.231	5.5577	6.0805	1.576	10.467
-1.9076	-4.2271	-6.2033	-13.222	-13.292	3.2541	7.8152	-5.1712	7.0578
7.2864	2.4375	-7.2919	-1.9805	-2.4561	-3.8235	1.9452	-12.62	3.2299
11.195	-4.264	-4.7404	-3.167	-3.6191	0.86269	2.9042	-16.325	4.2961
-4.345	1.2562	-7.1958	0.16974	1.7029	-1.492	1.5283	-6.5331	3.3345
3.9356	9.3716	-6.7978	-1.698	7.623	-2.4949	2.2951	-5.3172	-5.7764
5.9097	3.1093	-3.3932	-2.3631	-2.4159	-8.9882	3.4082	-10.551	-1.1789
10.78	7.6115	-7.384	0.58734	-7.8081	-8.5315	1.1517	-8.0359	1.6416
-0.639	-1.4485	-6.7499	-3.488	6.9749	-6.6533	2.3406	-2.3361	8.6737
-5.5942	6.4403	-4.6084	-4.9012	3.1312	-5.2757	0.67955	9.8431	6.1156
-10.053	7.2812	-4.136	-1.0664	-9.1545	-5.0305	0.65191	4.0867	3.9759
-4.7351	2.7049	-3.9395	0.5658	-2.5297	-3.9538	5.3171	-2.3236	1.8495
-1.6502	-0.7518	-2.9642	-3.3848	1.6832	-5.9036	3.7541	0.40041	1.2256
2.8685	-4.697	-0.4224	-2.5723	-0.2206	-1.6183	1.1668	8.7748	9.3765
3.5003	-7.7509	2.0177	0.68895	0.2057	-1.0204	1.3217	8.1578	12.657
-4.6961	-10.491	-1.7416	-0.2389	7.4505	-3.157	1.3403	9.4271	4.1858
1.2058	-3.916	-1.2636	-0.2399	7.6521	-5.3506	1.5133	7.7551	14.525
-6.4382	-8.6754	-0.9236	-3.7929	15.042	-3.5077	-3.0452	1.7676	2.8771
-4.9181	-6.9876	0.19402	1.7984	8.7768	-3.7392	-3.4357	1.7324	-2.3692
-3.9715	-7.9785	-3.1878	-0.5543	3.795	-4.4571	-2.4703	4.2852	-8.0093
-3.8115	-11.304	-2.5078	2.4578	5.8057	-5.4683	-3.3518	1.8309	-12.024
6.7457	-7.8578	-1.7076	5.5628	9.481	-0.9326	-3.2076	-1.0953	-7.7491
3.0949	2.2654	-4.1952	5.1846	8.7066	-0.861	-3.3452	4.6737	3.8519
-11.544	-2.2973	-0.9309	-0.4837	8.0785	0.46757	-4.0432	1.7158	6.5965

Columns 20 through 19

1.6365	-1.131	0.95218	-1.4901	-0.5844	0.85618	7.4173
1.7636	1.2923	-2.6191	-3.228	4.3926	1.0593	8.9483
1.8369	0.085185	-5.2711	-3.0347	6.0177	0.65688	3.763
-2.0977	0.14586	-3.059	-1.2238	4.1612	0.37762	0.62584
-0.5124	0.9595	2.7653	1.5435	-0.6253	0.23636	3.2775
4.8504	4.71	-0.1317	-0.4092	1.261	1.3094	0.2803
3.7319	6.4031	0.81543	0.53357	3.802	2.8623	1.8251
0.22635	2.687	3.8721	2.157	5.713	2.573	-5.7396
-1.4897	2.9339	7.5895	5.8807	5.2834	1.8677	-0.0407
-2.7028	0.687	7.0072	3.9533	-0.52	2.4701	0.10096
-4.4736	0.42817	9.8717	2.4321	-1.3253	0.46035	1.915
-1.8861	-3.5134	2.5109	-0.0674	-0.3114	1.3534	9.3563

0.34578	0.22365	6.2212	0.14612	-6.979	1.6068	5.8097
-0.1404	-0.46953	3.6474	3.5714	-4.8454	2.45	8.2733
-1.08	0.3652	1.7047	2.8233	-2.9715	1.9939	7.4173
-0.2956	0.84346	-0.0446	2.0068	-1.7268	2.8866	4.5941
-2.1298	1.0659	1.2568	-0.6611	-3.7913	1.4253	1.8139
-1.9467	-1.1128	1.3531	-0.4339	-3.9757	-0.8175	3.2005
-1.1043	-0.89402	1.5812	3.7911	-0.2027	0.5193	3.0453
-1.3221	3.4257	-3.5949	4.4182	0.59613	-0.278	0.52548
-1.8657	2.1811	0.70214	4.3112	-1.3426	-0.6878	1.045
-0.6389	1.9278	-0.7993	7.7367	-0.9877	1.7034	-2.4885
-1.3479	3.7417	-1.6852	5.5255	0.42886	0.97252	-1.4757
-0.7232	0.76607	-2.8013	3.9854	0.38476	-0.291	2.3663

Table:7.3 The weights of output hidden layer Wpq= Columns 1 through 6

3.5209	-7.1428	26.305	15.263	4.7571	-14.145
-42.306	-10.36	-2.0748	-26.457	-2.6515	7.816
8.4899	16.097	-14.044	21.864	-21.917	34.193
11.643	-10.888	13.138	32.444	42.79	34.189
24.855	-23.32	5.0913	-20.466	13.417	29.951
-2.3158	9.5372	42.607	25.355	-30.143	-26.286
-29.147	6.4883	-33.692	16.256	8.2189	2.8307
23.12	23.612	-29.338	-25.417	11.87	-1.3095
11.061	-19.962	24.731	-14.606	2.5366	10.15
22.609	-35.718	-24.607	21.101	-17.802	-9.8699
4.3905	26.561	21.636	-24.573	17.478	-12.807
17.991	-12.061	14.329	-15.6	24.833	-4.0849
14.131	-2.37	2.9005	7.7991	-11.773	-16.645
5.6197	18.838	5.8294	2.9158	-1.8076	-11.173
-13.65	33.786	4.5649	12.201	1.0307	10.751
5.5279	6.1662	-2.1198	2.7799	-2.7174	-23.526
-4.253	1.0515	-2.1606	-13.045	-6.4614	1.5897
12.737	-21.595	-19.607	-19.178	2.0731	-15.159
-4.4401	-16.582	6.9463	9.1655	-39.281	12.204
2.0852	-2.478	-7.6464	1.2094	6.7848	-7.4115
6.5414	-6.7843	-4.7554	2.3912	11.51	0.94323
-7.9592	-14.221	4.4906	-4.5619	9.8272	-10.128
-2.7796	-5.4454	4.5053	3.8655	10.641	-13.014
-13.218	4.7918	1.2072	8.6155	10.228	0.37052
-7.9988	0.69064	-0.3798	2.6206	1.5841	4.8211
2.6156	20.64	-8.2857	-3.0794	-13.686	-4.9576

7.5.2 Testing phase

During testing phase the word has been recorded as a wave file, and then loaded in to Matlab program and goes through feature extraction process. Features are converted to an S-Matrix and given as inputs to the network; the program will use the network which is in database for recognizing the words.

6.5.2 Classification process

Once test word has been recorded the speech into a wave file, the wave file is then loaded into the Matlab program and goes through feature extraction process. A targeted S-Matrix is generated and it goes through the Neural Network for classification. Inside the Neural Network, the targeted S-Matrix is being compared against the reference S-Matrices in the database. An individual score is generated for each individual word in the database. The first decision applied is that the word with the highest score is deemed the word. This is followed by an inspection on the highest score obtained. A second classification decision involves checking on whether the highest score exceeds a threshold. If the threshold is exceeded, then the test word is classified as the word corresponding to the highest score.

Table.7.4 Vocabulary of Hindi words

S.NO	WORD NAME	S.NO	WORD NAME	S.NO	WORD NAME
1	Krodh	41	Tavu	81	Kaksha
2	Jnan	42	Shatru	82	Kathor
3	Daan	43	Vahan	83	Kadam
4	Satya	44	Haati	84	Kanya
5	Daya	45	Shir	85	Kapat
6	Buddhi	46	Chuha	86	Kamar
7	Vidhya	47	Dus	87	Karma
8	Ersha	48	Bees	88	Kalian
9	Raja	49	Assi	89	Kavita
10	Bhakti	50	Rakhi	90	Kasam
11	Shakti	51	Dava	91	Kagaz
12	Ram	52	Rassi	92	Kazal
13	Ravi	53	Shidi	93	Kaatna
14	Shashi	54	Chat	94	Koyala
15	Chand	55	Chata	95	Ganga
16	Paani	56	Peti	96	Ghabir
17	Hava	57	Khat	97	Ghatak
18	Bhali	58	Haya	98	Chanchal
19	Aam	59	Pashu	99	Chikna

20	Hosh	60	Gay	100	Ghatna
21	Jeevan	61	Paxi	101	Jutha
22	Jal	62	Vrux	102	Taaya
23	Sthal	63	Baaz	103	Taarik
24	Mabh	64	Naath	104	Tiranga
25	Gendh	65	Ambar	105	Todna
26	Balla	66	Anda	106	Dakal
27	Aakhir	67	Anjam	107	Dada
28	Akbar	68	Abhay	108	Damad
29	Aaph	69	Amal	109	Darji
30	Ham	70	Amma	110	Dana
31	Tum	71	Indr	111	Dixa
32	Mitr	72	Ichha	112	Dulha
33	Dost	73	Indhan	113	Doodh
34	Guru	74	Uday	114	Doosra
35	Saathi	75	Udyan	115	Nambar
36	Uva	76	Rugvedh	116	Najar
37	Bhai	77	Rushu	117	Naya
38	Papa	78	Ainak	118	Nasib
39	Mama	79	Kangha	119	Nisha
40	Chacha	80	Kanjus	120	patha

Table.7.5 Recognition results obtained for a vocabulary of 4 Hindi words

S.NO	WORD NAME	ACCURACY IN (%)
1	Krodh	100
2	Jnan	100
3	Daan	100
4	Saty	100

Table.7.6 Recognition results obtained for a vocabulary of 8 Hindi words

S.NO	WORD NAME	ACCURACY IN (%)
1	Krodh	100
2	Jnan	100
3	Daan	100
4	Saty	100
5	Daya	100
6	Buddhi	100
7	Vidhya	100
8	Ersha	100

Table:7.7 Recognition results obtained for a vocabulary of 16 Hindi words

S.NO	WORD NAME	ACCURACY IN (%)
1	Krodh	80
2	Jnan	60
3	Daan	60
4	Satya	70
5	Daya	70
6	Buddhi	90
7	Vidhya	90
8	Ersha	90
9	Raja	100
10	Bhakti	80
11	shakti	80
12	Ram	70
13	Ravi	70
14	Shashi	60
15	Chand	60
16	Paani	80

From the Tables 7.5, 7.6 and 7.7 it is seen that the recognition accuracy in case of reduced vocabulary is far more than the larger vocabulary. By recognition accuracy we mean how many times that word is recognized accurately out of the number of times it was spoken for recognition. It is due to the fact that the approach used is whole word. However, if phoneme-based approach is used the recognition accuracy will be much larger and the execution time is much smaller than the case which is discussed. However, other factors also influence the recognition rate like the effect of noise, effect of room acoustics, and appearance of similar words.

Chapter - 8

CONCLUSIONS AND SCOPE FOR FUTURE WORK

8.1 Conclusion

An experimental speaker dependent isolated word recognition system for Hindi language was implemented. The choice of Hindi language, on its part, has not been arbitrary. A major part of the motivation for choosing Hindi as the language for recognition system comes from its local relevance. Whereas the English speaking community forms a very small percentage of India's population, Hindi being the national language of India is much more widely accepted. Hindi also offers several advantages as the language for speech recognition. It does not have a separate phoneme-set and a separate Alphabet. In other words, the alphabet itself is the phoneme-set. Besides, the Hindi Alphabet is very well categorized on the basis of similarities in articulation methods of its letters. This property of Hindi makes it free of homonyms, thus obviating the complexity of handling them in design of speech recognition systems. The advantages are even more in the case of phoneme-based recognizers, where a phonetic dictionary is not required for speech to text conversion. The results were found to be satisfactory.

With the positive results collected, Speech recognition using Neural Network has proven to be excellent in classifying speech signals. Unlike traditional speech recognition techniques which involve complex Fourier transformations, the method used in coding the signal is simple and accurate.

From the results, it is obvious that single syllable words are more reliable in terms of training. This is probably because humans' pronunciations of single syllable words are more consistent. There are still a few "False" acceptances and "False" rejections being detected. This may be considered a serious issue when it is applied in a high security room. The main reason behind these errors is due to the inconsistency in the human speech. This dissertation has addressed the question of whether neural networks can serve as a useful foundation for a large vocabulary, speaker dependent speech recognition system.

This work has been studied and analyzed for different techniques of speech recognition. The first part was started from the recognition background, which is based

on the digital signal theory and modeling of the speaker vocal tract. Then various techniques for reducing amount of test data or feature extraction is discussed. Further, we studied most popular speech recognition methods, which are commonly used in the speech recognition. Finally, the new approach developed for training the neural network's architecture proved to be simple and very efficient. It reduced considerably the amount of calculations needed for finding the correct set of parameters. If the traditional approach had been used instead, the amount of calculations would have been higher.

8.2 Future scope

In this dissertation work speaker dependent isolated word speech recognition of Hindi words is implemented. However using the phoneme-based approach instead of whole word approach can further increase the accuracy. The work can be further extended to any one of the directions:

1. Phoneme-based approach to speaker dependent isolated word speech recognition can be implemented because the accuracy in this case will be far better than that in the whole word.
2. From speaker dependent to speaker independent type of speech recognition. Such, that the user and it do not affect the recognition accuracy is able to recognize the words spoken by different speakers.
3. From isolated word to continuous word speech recognition. So, that it will be able to recognize the continuous speech, such as lectures, debates, politician's speech etc., which find more applications in the commercial sector.
4. Using variable input length neural networks can further increase the accuracy of recognition.
5. The vocabulary size can be increased from small vocabulary system to large vocabulary.

REFERENCES

[1] S. Yang, M.J. Er, and Y. Gao, "A High Performance Neural-Network-Based Speech Recognition System", Proceeding of International Joint Conference on Neural Networks, Vol2, 2001, pp1527.

[2] Pablo zegers, "Speech recognition using neural network", Master degree thesis Arizona university, 1998.

[3] Whee kian ryan lee," Speech recognition using TESPAR and neural networks", bachelor science degree thesis, Queenland university, May 2003.

[4] C. R. Jankowski Jr., H. H. Vo, and R. P. Lippmann, "A Comparison of Signal Processing Front Ends for Automatic Word Recognition," IEEE Transactions on Speech and Audio processing, vol. 3, no. 4, July 1995.

[5] Joe Tebelskis, "speech recognition using neural networks", PhD thesis, Carnegie Mellon University, May 1995

[6] K. Torkkola and M. Kokkonen, "Using the Topology-Preserving Properties of SOMs in Speech Recognition," Proceedings of the IEEE ICASSP, 1991.

[7] T. Zeppenfeld and A. Waibel, "A Hybrid Neural Network, Dynamic Programming Word Spotter," IEEE Proceedings ICASSP, 1992.

[8] K-F Lee, H-W. Hon, and R. Reddy, "An Overview of the SPHINX Speech Recognition System," IEEE Transactions on Acoustic, Speech, and Signal Processing, vol. 38, no. 1, January 1990.

[9] J. Tebelskis, "Speech Recognition Using Neural Networks," PhD Dissertation, Carnegie Mellon University, 1995.

[10] Y. Gong, "Stochastic Trajectory Modeling and Sentence Searching for Continuous Speech Recognition," IEEE Transactions on Speech and Audio Processing, vol. 5, no. 1, January 1997.

[11] Evgeny Karpov, "Real-Time Speaker Identification", Master's Thesis University of Joensuu, 15.01.2003.

[12] Tarun Pruthi, Sameer Saksena and Pradip K Das, "Swaranjali: Isolated Word Recognition for Hindi Language using VQ and HMM",TU www.ece.umd.edu/~tpruthi/papers/MPS012-ICMPS2000.pdfUH

[13] J. M.Naik, "Speaker Verification: A Tutorial", *IEEE Communications Magazine*, January 1990, pp.42-48.

[14] B. S. Atal, "Automatic Recognition of Speakers from their Voices", Proceedings *of the IEEE*, vol 64, 1976, pp 460 – 475.

[15] H. Gish and M. Schmidt, "Text Independent Speaker Identification", *IEEE signal Processing Magazine*, Vol. 11, No. 4, 1994, pp. 18-32.

[16] J. R. Deller, J. H. L. Hansen, J. G. Proakis, *Discrete-Time Processing of Speech Signals*, Piscataway (N.J.), IEEE Press, 2000.

[17] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Englewood Cliffs (N.J.), Prentice Hall Signal Processing Series, 1993.

[18] S. Molau, M. Pitz, R. Schluter, H. Ney, "Computing Mel-Frequency Cepstral Coefficients on the Power Spectrum", *Acoustics, Speech, and Signal Processing*, 2001 IEEE International Conference, Volume: 1, 2001, pp. 73-76

[19] X. Huang, A. Acero and H.-W. Hon, *Spoken language processing*, Upper Saddle River, New Jersey, Prentice Hall PTR, 2001.

[20] N.P.Padhy.,"Artificial Intelligence and Intelligent Systems", published in India by Oxford university press 2005.

[21] M. T. Hagan and H. B. Demuth and M. Beale, "Neural Network Design" International Thomson Publishing, 1995.

[22] R.A. King and T.C Phipps, "Shannon, TESPAR and Approximation Strategies", ICSPAT 98, Vol. 18, pp 445-453, Great Britain 1999.

[23] Gold B. and Morgan N., Speech and Audio Signal Processing, New York, NY: John Wiley & Sons, Inc., 2000.

[24] John R. Deller, Jr. and John H.L.Hansen and John G.proakis., Discrete-Time Processing of Speech Signals, IEEE Signal Processing Society, sponsor, IEEE PRESS.

[25] J.R. Deller, J.G. Proakis, J.H.L. Hansen, *Discrete Time Processing of Speech Signals*, MacMillian Publishing Co., New York, New York, USA, 1993.

[26] D. Richard, C. Miall, and G. Mitchison, "The Computing Neuron," Addison-Wesley, 1989.

[27] S. Furui, *Digital Speech Processing, Synthesis and Recognition,* New York, Marcel Dekker, 2001.

[28] B.Gold and N. Morgan, Speech and Audio Signal Processing, New York, NY: John Wiley & Sons, Inc., 2000.

[29] C. Becchetti and L. P. Ricotti, Speech Recognition Theory and C++ Implementation, New York, NY: John Wiley & Sons, Inc.,2000.

[30] K.S. Narendra nadParthasarathy,"Identification and control of dynamic systemsusing neural networks", IEEE Trans. Neural Networks, vol. 1, pp. 4-27, Mar. 1990.

[31] N. Negroponte, "Being Digital," Vintage Books, 1995.

[32] L. Rabiner and G. Juang, "Fundamentals of Speech Recognition," Prentice-Hall, 1993.

[33] S. Furui, "Digital Speech Processing, Synthesis and Recognition," Marcel Dekker Inc., 1989.

[34] S. Pinker, "The Language Instinct," HarperPerennial, 1995.