SOFT COMPUTING APPROACH IN SPEECH RECOGNITION

A DISSERTATION

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

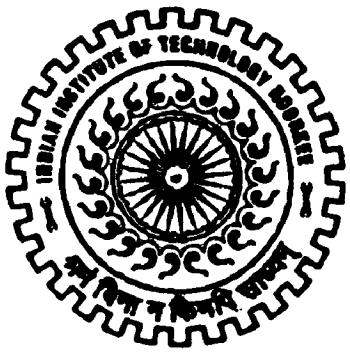
MASTER OF TECHNOLOGY

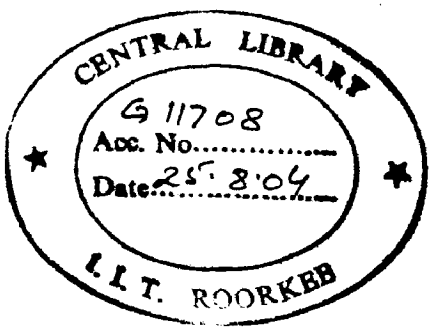
in

INFORMATION TECHNOLOGY

By

SHYAM SUNDAR POLICETTI





IIT Roorkee - CDAC, NOIDA,
c-56/1, "Anusandhan Bhawan"
Sector 62, Noida-201 307

JUNE, 2004

CANDIDATE'S DECLARATION

I hereby declare that the work which is being presented in this dissertation titled **"SOFT COMPUTING APPROACH IN SPEECH RECOGNITION"**, in partial fulfillment of the requirement for the award of the degree of **MASTER OF TECHNOLOGY** with specialization in **INFORMATION TECHNOLOGY**, submitted in the Department of Electronics & Computer Engineering, Indian Institute of Technology Roorkee, Roorkee, is an authentic record of my own original work carried out from August 2003 to June 2004, under the guidance and supervision of **Dr. Poonam Rani Gupta** , Associate Professor C-DAC, NOIDA.

I have not submitted the matter embodied in this dissertation for the award of any other degree.

Date: 29/6/04
Place: Noida

(SHYAM SUNDAR POLICETTI)

CERTIFICATE

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

Date: 29/6/04
Place: Noida

Dr. POONAM RANI GUPTA
Associate Professor
C-DAC,NOIDA

ACKNOWLEDGEMENTS

I take this opportunity to thank my esteemed guide , Dr. **Poonam Rani Gupta**, for providing the right impetus, freedom and guidance.

I express my gratitude to **Prof. PremVrat** Director IIT Roorkee, **Mr. R.K.Verma** Executive Director CDAC NOIDA and **Prof. Sarje** Head of Electronics and Computer Departmant of IIT Roorkee for enabling the conducive atmosphere for this course. I graciously acknowledge the role played by **Mr.V.N.Shukla**, course coordinator M.Tech(I.T) program CDAC NOIDA, **Prof. R.P.Agarwal,** course coordinator M.Tech(I.T) IIT Roorkee in making this program a success.

I am indebted to all my teachers who shaped and moulded me. A special thanks to Dr.S.N.Singh who demystified Neural Networks. Dear Prof T.S "Lion" Gururaj it is a great honour and inspiration to work under you. A special mention to "Buddy" Rao for bringing the best out of me with his constant encouragement .It took me a long time to realize the profoundness of his selfless friendship. Without him I would not have made it to IITR.

There are no words to describe "Beta" Rama's contribution , you complete me. Je t'aime beaucoup.

If ever I wanted to emulate some one in my life it would be my dearest brother MAD.I simply do not have any adjectives to describe his brilliance, poise and motivation.

Two years of one's life is a long time but it was made short by Swap, Lat, Mani, Sis, Kid, Deepak, Sarika, Vicky, Sumanth, Sekar, Vissu, Nulu, Suresh, "Item", Prem, Yamuna, Murali ,"Neta", Kiran, Satish and Dharmu. To completely list all the people is an NP complete problem.

Last but not the least I would like to say that the cherished values inculcated in me by Amma and Nanna are a bulwark by my side.

ABSTRACT

In this dissertation, application of soft computing techniques in speech recognition has been explored. At present the prevailing technology for speech recognition is predominantly Hidden Markov Model based. a statistical framework that supports both acoustic and temporal modeling. Despite their state-of-the-art performance, HMMs make a number of sub optimal 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.

Neural networks can be used for situations where speech feature vectors are non-linearly distorted, such as in noisy reverberant speech or telephone speech. By using a neural network, the adaptation process requires a small amount of training data. First, a neural network is applied to the computation of an inverse distortion function. This type of network requires simultaneously recorded input and target pairs for training. Traditionally, neural networks are trained to minimize the mean squared error between the network output and the corresponding target value. However, minimizing the mean squared error does not guarantee maximum recognition accuracy. Therefore, a new objective function for the neural network is proposed, which makes use of fuzzy rules.

Speech recognition throws up a myriad of problems, these problems are generally of pattern recognition, approximation and optimization. These problems have imprecise and distorted data. As a result conventional techniques are grossly inadequate in this domain. Fuzzy logic is the most appropriate in these circumstances. Optimization is necessary while designing Neural Network and handling the input features. This dissertation has explored the application of this paradigms under various conditions.

CONTENTS

Chapter 1

INTRODUCTION

1.1 MOTIVATION

Speech is a natural mode of communication for people. We learn all the relevant skills during early childhood, without instruction, and we continue to rely on speech communication throughout our lives. It comes so naturally to us that we don't realize how complex a phenomenon speech is. The human vocal tract and articulators are biological organs with nonlinear properties, whose operation are 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 [11], even more than speech generation, a very complex problem.

Yet people are so comfortable with speech that we would also like to interact with our computers via speech, rather than having to resort to primitive interfaces such as keyboards and pointing devices. A speech interface would support many valuable applications for example, telephone directory assistance, spoken database querying for novice users, "hands busy" applications in medicine or fieldwork, office dictation devices, or even automatic voice translation into foreign languages. Such tantalizing applications have motivated research in automatic speech recognition since the 1950's. Great progress has been made so far, especially since the 1970's, using a series of engineered approaches that include template matching, knowledge engineering, and statistical modeling. Yet computers are still nowhere near the level of human performance at speech recognition, and it appears that further significant advances will require some new insights. What makes people so good at recognizing speech?

Intriguingly, the human brain is known to be wired differently than a conventional computer; in fact it operates under a radically different computational paradigm. While conventional computers use a very fast & complex central processor

with explicit program instructions and locally addressable memory, by contrast the human brain uses a massively parallel collection of slow & simple processing elements (neurons)[3], densely connected by weights (synapses) whose strengths are modified with experience, directly supporting the integration of multiple constraints, and providing a distributed form of associative memory. The brain's impressive superiority at a wide range of cognitive skills, including speech recognition, has motivated research into its novel computational paradigm since the 1940's, on the assumption that brainlike models [12] may ultimately lead to brainlike performance on many complex tasks. This fascinating research area is now known as *connectionism*, or the study of *artificial neural networks*. The history of this field has been erratic, but by the mid-1980's, the field had matured to a point where it became realistic to begin applying connectionist models to difficult tasks like speech recognition. By 2000, many researchers had demonstrated the value of neural networks for important subtasks like phoneme recognition and spoken digit recognition, but it was still unclear whether connectionist techniques would scale up to large speech recognition tasks.

This dissertation demonstrates that softcomputing techniques can indeed form the basis for a general purpose speech recognition system, and that softcomputing techniques offer some clear advantages over conventional techniques.

1.2 SPEECH RECOGNITION

What is the current state of the art in speech recognition? This is a complex question, because a system's accuracy depends on the conditions under which it is evaluated under sufficiently narrow conditions almost any system can attain human-like accuracy, but it's much harder to achieve good accuracy under general conditions [26]. The conditions of evaluation and hence the accuracy of any system can vary along the following dimensions:

Vocabulary size and confusability. As a general rule, it is easy to discriminate among a small set of words, but error rates naturally increase as the vocabulary [3] size grows. For example, the 10 digits "zero" to "nine" can be recognized essentially perfectly, but vocabulary large size have higher error rate. On the other hand, even a small vocabulary can be hard to recognize if it contains confusable words. For example, the 26 letters of the English alphabet (treated as 26 "words") are very

difficult to discriminate because they contain so many confusable words (most notoriously, the E-set: "B, C, D, E, G, P, T, V, Z").

Speaker dependence vs. independence. By definition, a *speaker dependent* system is intended for use by a single speaker, but a *speaker independent* system is intended for use by any speaker. Speaker independence [4] is difficult to achieve because a system's parameters become tuned to the speaker(s) that it was trained on, and these parameters tend to be highly speaker-specific. Error rates are typically 3 to 5 times higher for speaker independent systems than for speaker dependent ones. Intermediate between speaker dependent and independent systems, there are also *multi-speaker* systems intended for use by a small group of people, and *speaker-adaptive* systems which tune themselves to any speaker given a small amount of their speech as enrollment data.

Isolated, discontinuous, or continuous speech. *Isolated speech* means single words; *discontinuous speech* means full sentences in which words are artificially separated by silence; and *continuous speech* means naturally spoken sentences. Isolated and dis continuous speech recognition is relatively easy because word boundaries are detectable and the words tend to be cleanly pronounced. Continuous speech[4] is more difficult, however, because word boundaries are unclear and their pronunciations are more corrupted by *coarticulation*, or the slurring of speech sounds, which for example causes a phrase like "could you" to sound like "could jou".

Task and language constraints. Even with a fixed vocabulary, performance will vary with the nature of constraints on the word sequences[5] that are allowed during recognition. Some constraints may be *task-dependent* (for example, an airline querying application may dismiss the hypothesis "The apple is red"); other constraints may be *semantic* (rejecting "The apple is angry"), or *syntactic* (rejecting "Red is apple the"). Constraints are often represented by a *grammar*, which ideally filters out unreasonable sentences so that the speech recognizer evaluates only plausible sentences. Grammars[5] are usually rated by their *perplexity*, a number that indicates the grammar's average branching factor (i.e., the number of words that can follow any given word). The difficulty of a task is more reliably measured by its perplexity than by its vocabulary size.

Read vs. spontaneous speech. Systems can be evaluated on speech that is either read from prepared scripts, or speech that is uttered spontaneously. Spontaneous speech is vastly more difficult, because it tends to be peppered with disfluencies [26] like "uh" and "um", false starts, incomplete sentences, stuttering, coughing, and laughter; and moreover, the vocabulary is essentially unlimited, so the system must be able to deal intelligently with unknown words (e.g., detecting and flagging their presence, and adding them to the vocabulary, which may require some interaction with the user).

Adverse conditions. A system's performance can also be degraded by a range of adverse conditions. These include environmental noise (e.g., noise in a car or a factory); acoustical distortions (e.g, echoes, room acoustics); different microphones (e.g., close-speaking, omni-directional, or telephone); limited frequency bandwidth[5] (in telephone transmission); and altered speaking manner (shouting, whining, speaking quickly, etc.). In order to evaluate and compare different systems under well-defined conditions, a number of standardized databases have been created with particular characteristics.

The central issue in speech recognition is dealing with variability. Currently, speech recognition systems distinguish between two kinds of variability acoustic[13] and temporal. *Acoustic variability* covers different accents, pronunciations, pitches, volumes, and so on, while *temporal variability* covers different speaking rates. These two dimensions are not completely independent when a person speaks quickly, his acoustical patterns become distorted as well but it's a useful simplification to treat them independently. Speech can come from different sources this causes a lot of variability. As speech comes through the medium of transmission a lot of noise is added ,this noise introduces a lot of ambiguity in speech recognition. In addition to this the rate at which the speaker speaks also matters a lot.

Of these two dimensions, temporal variability is easier to handle. An early approach to temporal variability was to linearly stretch or shrink (*"warp"*) an unknown utterance to the duration of a known template. Linear warping proved inadequate, however, because utterances can accelerate or decelerate at any time; instead, nonlinear warping was obviously required. Soon an efficient algorithm[5] known as *Dynamic Time Warping* was proposed as a solution to this problem. This algorithm (in some form) is now used in virtually every speech recognition system, and the problem of temporal variability is considered to be largely solved.

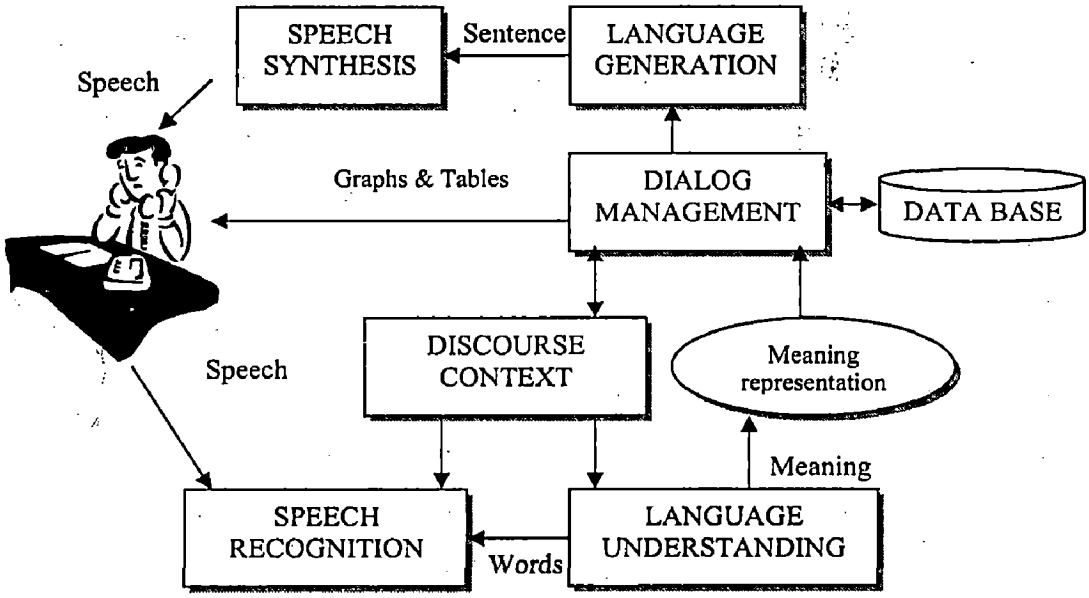


Figure 1.1 Automated Response System

Acoustic variability is more difficult to model, partly because it is so Heterogeneous in nature. Consequently, research in speech recognition has largely focused on efforts to model acoustic variability. Past approaches to speech recognition have fallen into three main categories:

Template-based approaches, in which unknown speech is compared against a set of pre-recorded [11] words (*templates*), in order to find the best match. This has the advantage of using perfectly accurate word models; but it also has the disadvantage that the prerecorded templates are fixed, so variations in speech can only be modeled by using many templates per word, which eventually becomes impractical.

Knowledge-based approaches, in which "expert" knowledge[12] about variations in speech is hand-coded into a system. This has the advantage of explicitly modeling variations in speech; but unfortunately such expert knowledge is difficult to obtain and use successfully, so this approach was judged to be impractical, and automatic learning procedures were sought instead.

Statistical-based approaches, in which variations in speech are modeled statistically (e.g., by *Hidden Markov Models*, or *HMMs*), using automatic learning procedures. This approach represents the current state of the art. The main disadvantage of statistical models[9] is that they must make a priori modeling assumptions, which are

liable to be inaccurate, handicapping the system's performance. We will see that neural networks help to avoid this problem.

1.3 VARIABILITIES OF SPEECH

Automatic speech recognition involves a number of disciplines such as physiology, acoustics, signal processing, pattern recognition, and linguistics. The difficulty of automatic speech recognition is coming from many aspects of these areas.

Variability from speakers: A word may be uttered differently by the same speaker because of illness or emotion. It may be articulated differently depending on whether it is planned read speech or spontaneous conversation. The speech produced in noise is different from the speech produced in a quiet environment because of the change in speech production in an effort to communicate more effectively across a noisy environment. Since no two persons share identical vocal cords[11] and vocal tract, they can not produce the same acoustic signals. Typically, females sound different from males. So do children from adults. Also, there is variability due to dialect and foreign accent.

Variability from environments: The acoustical environment[1] where recognizers are used introduces another layer of corruption in speech signals. This is because of background noise, reverberation, microphones, and transmission channels.

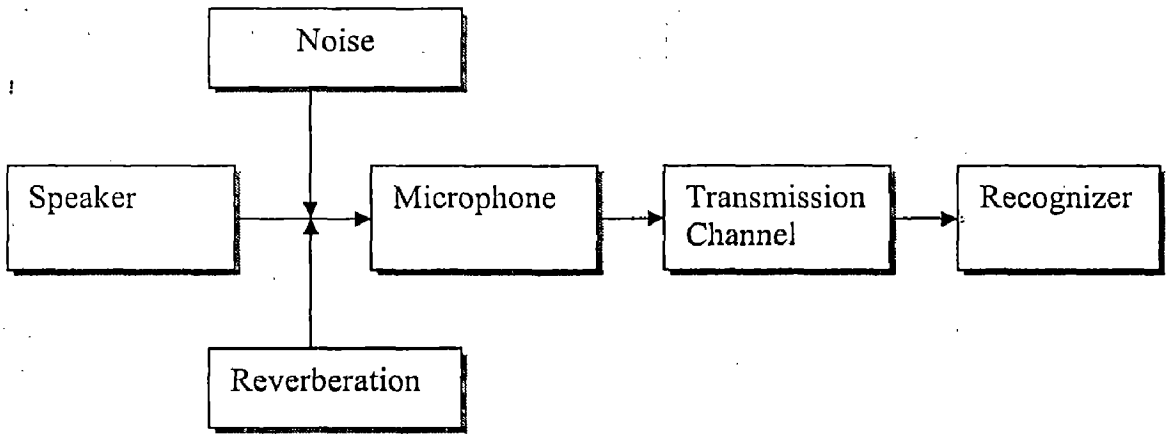


Figure 1.2 Block Diagram of Speech Recognition

One popular method in dealing with variabilities in speech recognition is a statistical approach such as HMM's[13]. In order to deal with variability due to speakers, for example, large vocabulary speaker-independent speech recognizers are usually trained using a large amount of speech data collected from a variety of speakers .If the same amount of training data is used, however, speaker-dependent systems usually perform better than speaker-independent recognizers. In this research, the problem of environmental variability is addressed, and the robust speech recognition methods that do not require a large amount of data are explored

Background noise: When distant-talking speech is to be recognized, not only the intended speaker's voice, but also background noise[8] is picked up by the microphone. The background noise can be white or colored, and continuous or pulsate. Complex noise such as a door slam, cross talk, or music is much more difficult to handle than simple Gaussian noise[9].

Room reverberation: in an enclosed environment such as a room, objects and surfaces reflect the acoustic speech wave, and signals are degraded by multi-path reverberation.

Different microphones: It is usually the case that a recognizer is trained using a high quality close-talking microphone, while it may be used in the real world with unknown microphones that have different frequency response characteristics.

Transmission channels - telephone: A great deal of effort has been put on telephone speech recognition because of its vast range of applications. However, due to the narrow bandwidth (300-3400 Hz) and non-linear distortion in transmission channels, telephone speech recognition is much more difficult than full bandwidth speech recognition. When the training and the testing environments are not matched, it affects speech feature vectors that are used in the recognition process. This typically makes speech recognizers vulnerable to changes in operating environments.

1.4 NEURAL NETWORKS

Connectionism, or the study of artificial neural networks, was initially inspired by neurobiology, but it has since become a very interdisciplinary field, spanning computer science, electrical engineering, mathematics, physics, psychology, and linguistics as well. Some researchers are still studying the neurophysiology of the

human brain, but much attention is now being focused on the general properties of neural computation[1], using simplified neural models. These properties include:

Trainability. Networks can be taught to form associations between any input and output patterns. This can be used, for example, to teach the network to classify speech patterns into phoneme categories.

Generalization. Networks don't just memorize the training data; rather, they learn the underlying patterns, so they can generalize from the training data to new examples. This is essential in speech recognition, because acoustical patterns are never exactly the same.

Nonlinearity. Networks can compute nonlinear, nonparametric functions of their input, enabling them to perform arbitrarily complex transformations of data. This is useful since speech is a highly nonlinear process.

Robustness. Networks are tolerant of both physical damage and noisy data; in fact noisy data can help the networks to form better generalizations. This is a valuable feature, because speech patterns are notoriously noisy.

Uniformity. Networks offer a uniform computational paradigm which can easily integrate constraints from different types of inputs. This makes it easy to use both basic and differential speech inputs, for example, or to combine acoustic and visual cues in a multimodal system.

Parallelism. Networks are highly parallel in nature, so they are well-suited to implementations on massively parallel computers. This will ultimately permit very fast processing of speech or other data.

There are many types of connectionist models, with different architectures, training procedures, and applications, but they are all based on some common principles. An artificial neural network consists of a potentially large number of simple processing elements (called *units*, *nodes*, or *neurons*), which influence each other's behavior via a network of excitatory or inhibitory weights. Each unit simply computes a nonlinear weighted sum of its inputs, and broadcasts the result over its outgoing connections to other units. A training set consists of patterns of values that are assigned to designated input and/or output units. As patterns are presented from the training set, a learning rule modifies the strengths of the weights so that the network gradually learns the training set. This basic paradigm can be fleshed out in

many different ways, so that different types of networks can learn to compute implicit functions from input to output vectors, or automatically cluster input data, or generate compact representations of data, or provide content-addressable memory and perform pattern completion.

Neural networks are usually used to perform static pattern recognition, that is, to statically map complex inputs to simple outputs, such as an N-ary classification[1] of the input patterns. Moreover, the most common way to train a neural network for this task is via a procedure called *backpropagation*, whereby the network's weights are modified in proportion to their contribution to the observed error in the output unit activations (relative to desired outputs). To date, there have been many successful applications of neural networks trained by backpropagation. Speech recognition, of course, has been another proving ground for neural networks.

1.5 GENETIC ALGORITHMS

Algorithms for function optimization are generally limited to convex regular functions. However, many functions are multi-modal, discontinuous, and non-differentiable. Stochastic sampling methods have been used to optimize these functions. Whereas traditional search techniques use characteristics of the problem to determine the next sampling point (e.g., gradients, Hessians, linearity, and continuity), stochastic search techniques make no such assumptions. Instead, the next sampled point(s) is(are) determined based on stochastic sampling/decision rules rather than a set of deterministic decision rules.

Speech recognition is rich with areas where genetic algorithms is an inevitable optimizer Genetic algorithms[7] have been used to solve difficult problems with objective functions that do not possess "nice" properties such as continuity and differentiability. These algorithms maintain and manipulate a family, or population, of solutions and implement a survival of the fittest strategy in their search for better solutions. This provides an implicit as well as explicit parallelism that allows for the exploitation of several promising areas of the solution space at the same time. The implicit parallelism is due to the schema theory, while the explicit parallelism arises from the manipulation of a population of points the evaluation of the fitness of these points is easy to accomplish in parallel.

1.6 FUZZY LOGIC

Fuzzy logic has two different meanings. In a narrow sense, fuzzy logic is a logical system, which is an extension of multivalued logic. But in a wider sense, which is in predominant use today, fuzzy logic (FL)[7] is almost synonymous with the theory of fuzzy sets, a theory which relates to classes of objects with un sharp boundaries in which membership is a matter of degree. In this perspective, fuzzy logic in its narrow sense is a branch of FL. What is important to recognize is that, even in its narrow sense, the agenda of fuzzy logic is very different both in spirit and substance from the agendas of traditional multivalued logical systems.

Fuzzy logic is a convenient way to map an input space to an output space. This is the starting point for everything else, and the great emphasis here is on the word "convenient".

Fuzzy logic is conceptually easy to understand. The mathematical concepts behind fuzzy reasoning are very simple. What makes fuzzy nice is the "naturalness" of its approach and not its far-reaching complexity.

Fuzzy logic is flexible, with any given system, it's easy to massage it or layer more functionality on top of it without starting again from scratch. Fuzzy logic is tolerant of imprecise data everything is imprecise if you look closely enough, but more than that, most things are imprecise even on careful inspection. Fuzzy reasoning builds this understanding into the process rather than tacking it onto the end. Fuzzy logic can model nonlinear functions of arbitrary complexity. You can create a fuzzy system to match any set of input-output data. This process is made particularly easy by adaptive techniques like ANFIS(Adaptive Neuro-Fuzzy Inference Systems)[7]. Fuzzy logic can be built on top of the experience of experts. In direct contrast to neural networks, which take training data and generate opaque, impenetrable models, fuzzy logic lets you rely on the experience of people who already understand your system. Fuzzy logic can be blended with conventional control techniques. Fuzzy systems don't necessarily replace conventional control methods[12]. In many cases fuzzy systems augment them and simplify their implementation. Fuzzy logic is based on natural language. The basis for fuzzy logic is the basis for human communication. This observation underpins many of the other statements about fuzzy logic.

Chapter 2

LITERATURE SURVEY

2.1 INTRODUCTION

In this chapter the fundamental speech sounds are reviewed. All the basic phonetic units are explored. Then the signal processing part is explained .The features that are extracted from the speech signals is defined and elucidated. Finally soft-computing techniques that are to be applied are elaborated.

2.2 SPEECH SOUNDS

There are over 40 speech sounds in English which can be organized by their basic manner of production

Table 2.1 Phoneme Classification

Manner Class	Number
Vowels	18
Fricatives	8
Stops	6
Nasals	3
Semivowels	4
Affricates	2
Aspirant	1

Vowels, glides, and consonants differ in degree of constriction. Sonorant consonants have no pressure build up at constriction Nasal consonants lower the

velum allowing airflow in nasal cavity . Continuant consonants do not block airflow in oral cavity.

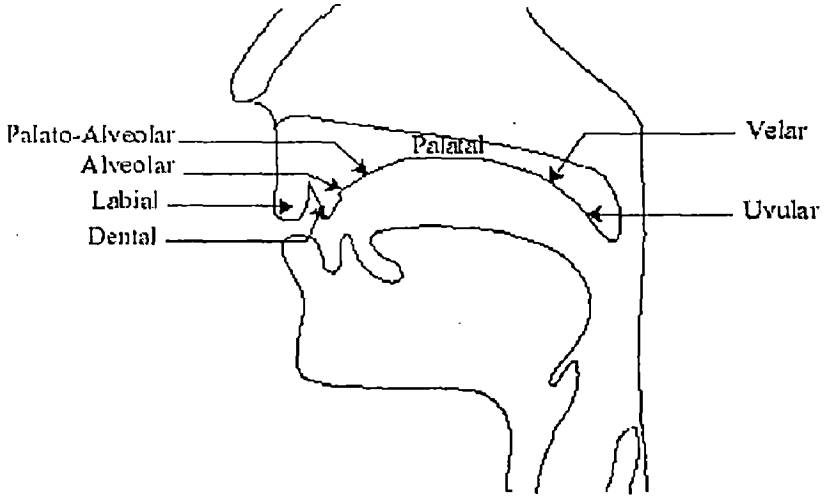


Figure 2.1 Anatomy of Speech Production

2.2.1 VOWEL PRODUCTION

While vowels are uttered there is no significant constriction in the vocal tract. Usually produced with periodic excitation. Acoustic characteristics[11] depend on the position of the jaw, tongue, and lips

Table 2.2 vowels

/i^{y}/ iy beat	/O/ ao bought	/a^{w}/ ay bite
/I/ ih bit	/^/ ah but	/O^{y}/ oy Boyd
/e^{y}/ ey bait	/O^{w}/ ow boat	/a^{w}/ aw bout
/ɛ/ eh bet	/U/ uh book	[ax] ax about
/ae/ ae bat	/u/ uw boot	[i^{x}] ix roses
/ɑ/ aa Bob	/ɝ/ er Bert	[ɦ] axr butter

There are approximately 18 vowels in American English made up of monothongs, diphthongs, and reduced vowels (schwa's).

2.2.2 FRICATIVE PRODUCTION

Fricatives are realized by turbulence produced at narrow constriction position determines acoustic characteristics can be produced with periodic excitation

Table 2.3 Fricatives

Type	Unvoiced	Voiced
Labial	/f/ f fee	/v/ v v
Dental	/T/ th thief	/D/ dh thee
Alveolar	/s/ s see	/z/ z z
Palatal	/S/ sh she	/Z/ zh Gigi

2.2.3 STOP CONSONANTS

There are 6 stop consonants. Three places of articulation: Labial, Alveolar, and Velar each place of articulation has a voiced and unvoiced stop. Unvoiced stops are typically aspirated. Voiced stops usually exhibit a "voice-bar'' during[2] closure Information about formant transitions and release useful for classification

Table 2.4 Stop Consonants

Type	Voiced	Unvoiced
Labial	/b/ b bought	/p/ p pot
Alveolar	/d/ d dot	/t/ t tot
Velar	/g/ g got	/k/ k cot

2.2.4 NASAL PRODUCTION

Velum lowering results in airflow through nasal cavity. Consonants[21] produced with closure in oral cavity. Nasal murmurs have similar spectral characteristics. Three places of articulation: Labial, Alveolar, and Velar.

Table 2.5 Nasal Sounds

Type	Nasal
Labial	/m/ m me
Alveolar	/n/ n knee
Velar	/ŋ/ ng sing

Nasal consonants are always attached to a vowel, though can form an entire syllable in unstressed environments. Place identified by neighboring formant transitions.

2.2.5 SEMIVOWEL PRODUCTION

Constriction in vocal tract, no turbulence. Slower articulatory motion than other consonants. Laterals form complete closure with tongue tip airflow via sides of constriction .

Table 1Semivowel

Type	Semivowel	Nearest Vowel
Glides	/w/ w wet	/u/
	/y/ y yet	/i/
Liquids	/r/ r red	/3/
	/l/ l let	/o/

There are 4 semivowels. Sometimes referred to as Liquids or Glides. Glides are a more extreme articulation of a corresponding vowel. Semivowels are always attached to a vowel, though /l/ can form an entire syllable in unstressed environments.

2.2.6 AFFRICATE PRODUCTION

There are two affricates in English:

Table 2.7 Affricate

Voiced	Unvoiced
/J/ jh judge	/C/ ch church

Affricates are produced by alveolar-stop palatal-frication. It involves sudden release of the constriction, turbulence noise. Can have periodic excitation during closure.

2.2.7 ASPIRANT PRODUCTION

There is one aspirant in American English: /h/ (e.g., "hat").It is produced by generating turbulence excitation at glottis. There is no constriction in the vocal tract, normal formant excitation. Sub-glottal coupling results in little energy in F1 region. Periodic excitation can be present in medial position.

2.3 FUNDAMENTALS OF SPEECH RECOGNITION

Speech recognition is a multileveled pattern recognition task, in which acoustical signals are examined and structured into a hierarchy of sub word units[23] (e.g., phonemes), words, phrases, and sentences. Each level may provide additional temporal constraints, e.g., known word pronunciations or legal word sequences, which can compensate for errors or uncertainties at lower levels. Combining decisions probabilistically at all lower levels, and making discrete decisions only at the highest level can best exploit this hierarchy of constraints. The structure of a standard speech recognition system is illustrated in Figure 2.2. The elements are as follows:

Raw speech. Speech is typically sampled at a high frequency, e.g., 16 KHz over a microphone or 8 KHz over a telephone. This yields a sequence of amplitude values over time.

Signal analysis. Raw speech should be initially transformed and compressed, in order to simplify subsequent processing. Many signal analysis techniques are available which can extract useful features and compress the data by a factor of ten without losing any important information. Among the most popular techniques Fourier analysis (FFT) yields discrete frequencies over time, which can be interpreted

visually. Frequencies are often distributed using a *Mel* scale, which is linear in the low range but logarithmic in the high range, corresponding to physiological characteristics of the human ear.

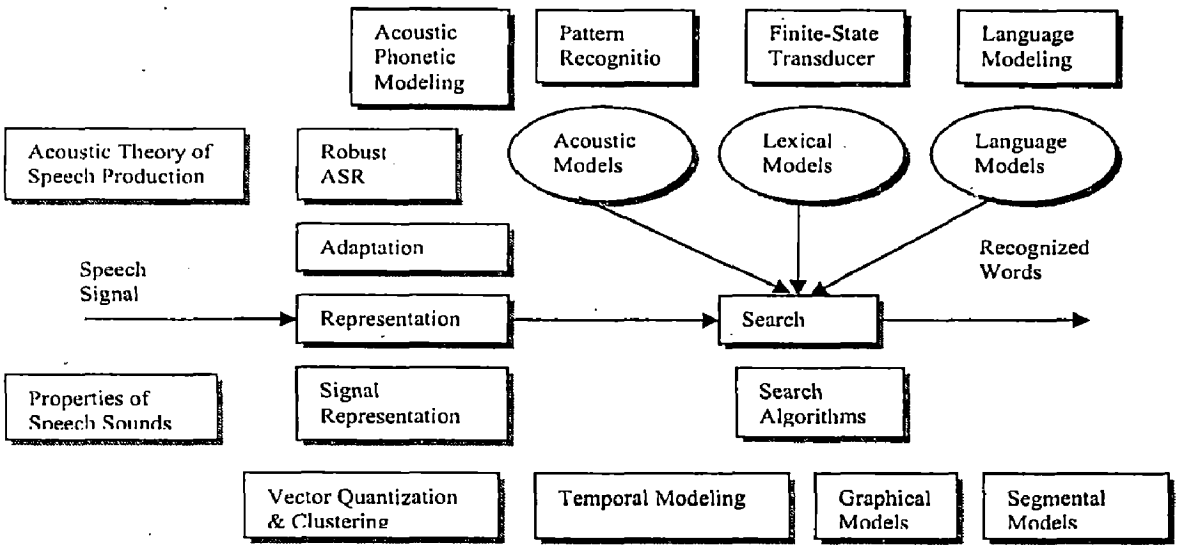


Figure 2.2 Paralinguistic Information Speech Understanding Multi-Modal Interfaces

Perceptual Linear Prediction[5] (PLP) is also physiologically motivated, but yields coefficients that cannot be interpreted visually.

Linear Predictive Coding[2] (LPC) yields coefficients of a linear equation that approximate the recent history of the raw speech values.

Cepstral analysis[2] calculates the inverse Fourier transform of the logarithm of the power spectrum of the signal.

In practice, it makes little difference which technique is used. Afterwards, procedures such as Linear Discriminant Analysis (LDA) may optionally be applied to further reduce the dimensionality of any representation, and to decorrelate the coefficients.

Speech frames. The result of signal analysis is a sequence of *speech frames*, typically at 10 msec intervals, with about 16 coefficients per frame. These frames may be augmented by their own first and/or second derivatives, providing explicit information[2] about speech dynamics; this typically leads to improved performance. The speech frames are used for acoustic analysis.

Acoustic models. In order to analyze the speech frames for their acoustic content, we need a set of *acoustic models*. There are many kinds of acoustic models, varying in their representation, granularity, context dependence, and other properties.

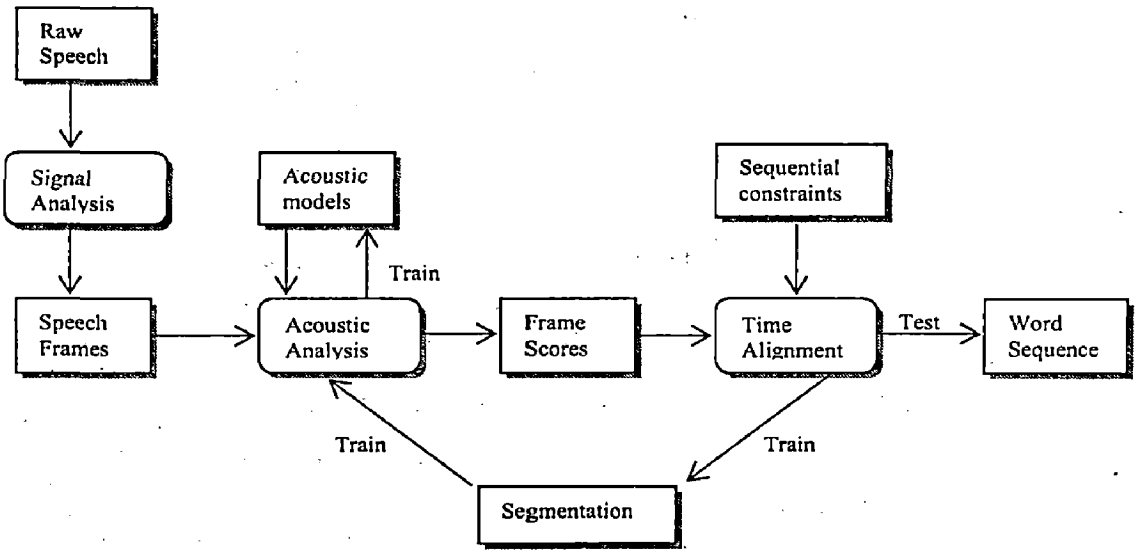


Figure 2.3 Preprocessing Speech

2.4 FEATURE EXTRACTION

As air is expelled from lungs, tensed vocal cords are caused to vibrate by the air flow. These quasi-periodic pulses are then filtered when passing through the vocal tract and the nasal tract, producing voiced sounds [20]. The different positions of articulators, such as jaw, tongue, lips, and velum, produce different sounds. When vocal cords are relaxed, the air flow either passes through a constriction in the vocal tract or builds up pressure behind a closure point and the pressure is suddenly released, causing unvoiced sounds [20]. The positions of constriction or closure decide different sounds. Speech is simply a sequence of these voiced and unvoiced sounds, which vary slowly (5100 ms) because the configuration of the articulators changes slowly.

The purpose of this module is to convert the speech waveform to some type of parametric representation (at a considerably lower information rate) for further analysis and processing. This is often referred as the *signal-processing front end.*

The speech signal is a slowly timed varying signal (it is called *quasi-stationary*). When examined over a sufficiently short period of time (between 5 and 100 msec), its

characteristics are fairly stationary. However, over long periods of time (on the order of 1/5 seconds or more) the signal characteristic change to reflect the different speech[2] sounds being spoken. Therefore, *short-time spectral analysis* is the most common way to characterize the speech signal.

A wide range of possibilities exist for parametrically representing the speech signal for the speaker recognition task, such as Linear Prediction Coding (LPC), Mel-Frequency Cepstral Coefficients (MFCC), and others. MFCC is perhaps the best known and most popular, and these will be used in this project.

MFCC's are based on the known variation of the human ear's critical bandwidths with frequency, filters spaced linearly at low frequencies and logarithmically at high frequencies have been used to capture the phonetically important characteristics of speech. This is expressed in the *mel-frequency* scale, which is a linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz. The process of computing MFCCs is described in more detail next.

2.4.1 MEL-FREQUENCY CEPSTRAL COEFFICIENTS PROCESSOR

The speech input is typically recorded at a sampling rate above 10000 Hz. This sampling frequency was chosen to minimize the effects of *aliasing* in the analog-to-digital conversion. These sampled signals can capture all frequencies up to 5 kHz, which cover most energy of sounds that are generated by humans. As been discussed previously, the main purpose of the MFCC processor is to mimic the behavior of the human ears. In addition, rather than the speech waveforms themselves, MFCC's are shown to be less susceptible to mentioned variations.

2.4.2 FRAME BLOCKING

In this step the continuous speech signal is blocked into frames of N samples, with adjacent frames being separated by M ($M < N$). The first frame consists of the first N samples. The second frame begins M samples after the first frame, and overlaps it by $N - M$ samples. Similarly, the third frame begins $2M$ samples after the first frame (or M samples after the second frame) and overlaps it by $N - 2M$ samples. This process continues until all the speech is accounted for within one or more frames. Typical values for N and M are $N = 256$ (which is equivalent to ~ 30 msec windowing and facilitate the fast radix-2 FFT) and $M = 100$.

2.4.3 WINDOWING

The next step in the processing is to window each individual frame so as to minimize the signal discontinuities at the beginning and end of each frame. The concept here is to minimize the spectral distortion by using the window to taper the signal to zero at the beginning and end of each frame. If we define the window as $w(n), 0 \le n \le N-1$, where N is the number of samples in each frame, then the result of windowing is the signal

$$y_l(n) = x_l(n)w(n), \quad 0 \le n \le N-1 \tag{2.1}$$

Typically the *Hamming* window is used, which has the form:

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right), \quad 0 \le n \le N-1 \tag{2.2}$$

2.4.4 FAST FOURIER TRANSFORM (FFT)

The next processing step is the Fast Fourier Transform, which converts each frame of N samples from the time domain into the frequency domain. The FFT is a fast algorithm to implement the Discrete Fourier Transform (DFT) which is defined on the set of N samples $\{x_n\}$, as follow:

$$X_n = \sum_{k=0}^{N-1} x_k e^{-2\pi jkn/N}, \qquad n = 0,1,2,\ldots,N-1 \tag{2.3}$$

Note that we use j here to denote the imaginary unit, i.e. $j = \sqrt{-1}$. In general X_n's are complex numbers. The resulting sequence $\{X_n\}$ is interpreted as follow: the zero frequency corresponds to n = 0, positive frequencies $0 < f < F_s/2$ correspond to values $1 \le n \le N/2-1$, while negative frequencies $-F_s/2 < f < 0$ correspond to $N/2+1 \le n \le N-1$. Here, F_s denotes the sampling frequency. The result after this step is often referred to as *spectrum* or *periodogram*.

2.4.5 MEL-FREQUENCY WRAPPING

As mentioned above, psychophysical studies have shown that human perception of the frequency contents of sounds for speech signals does not follow a linear scale. Thus for each tone with an actual frequency, f, measured in Hz, a subjective pitch is measured on a scale called the 'mel' scale. The *mel-frequency*

scale is a linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz. As a reference point, the pitch of a 1 kHz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 mels. Therefore we can use the following approximate formula to compute the mels for a given frequency f in Hz:

$$mel(f) = 2595 * \log_{10}(1 + f / 700) \qquad (2.4)$$

One approach to simulating the subjective spectrum is to use a filter bank, spaced uniformly on the mel scale . That filter bank has a triangular band-pass frequency response, and the spacing as well as the bandwidth is determined by a constant mel frequency interval. The modified spectrum of $S(\omega)$ thus consists of the output power of these filters when $S(\omega)$ is the input. The number of Mel spectrum coefficients, K, is typically chosen as 20.This information extracted from the bit of speech is ideal for recognition purposes we train the neural network using these coefficients. These coefficients give accurate results during testing.

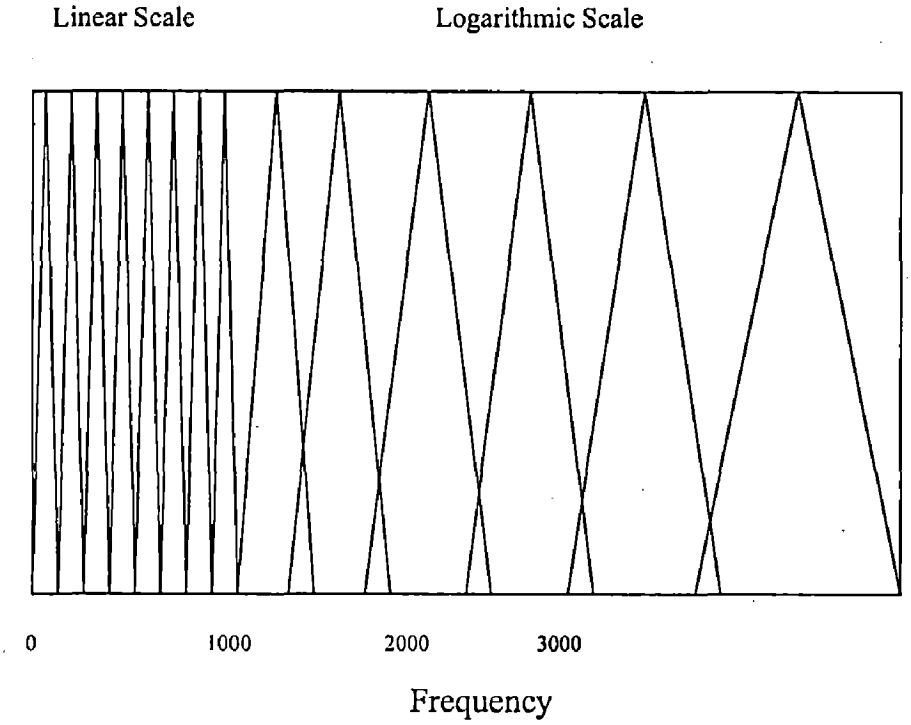


Figure 2.4 Mel –Scale Filter Banks

Note that this filter bank is applied in the frequency domain, therefore it simply amounts to taking those triangle-shape windows in the Figure 2.5 on the spectrum. A useful way of thinking about this mel-wrapping filter bank is to view each filter as an histogram bin (where bins have overlap) in the frequency domain.

2.5 DYNAMIC TIME WARPING

In this section we motivate and explain the *Dynamic Time Warping* algorithm, one of the oldest and most important algorithms in speech recognition. The simplest way to recognize an *isolated* word sample is to compare it against a number of stored word templates and determine which is the "best match". This goal is complicated by a number of factors. First, different samples of a given word will have somewhat different durations. This problem can be eliminated by simply normalizing the templates and the unknown speech so that they all have an equal duration. However, another problem is that the rate of speech may not be constant throughout the word; in other words, the optimal alignment between a template and the speech sample may be nonlinear.

Dynamic Time Warping (DTW)[5] is an efficient method for finding this optimal nonlinear alignment. DTW is an instance of the general class of algorithms known as *dynamic programming*. Its time and space complexity is merely linear in the duration of the speech sample and the vocabulary size. The algorithm makes a single pass through a matrix of frame scores while computing locally optimized segments of the global alignment path.

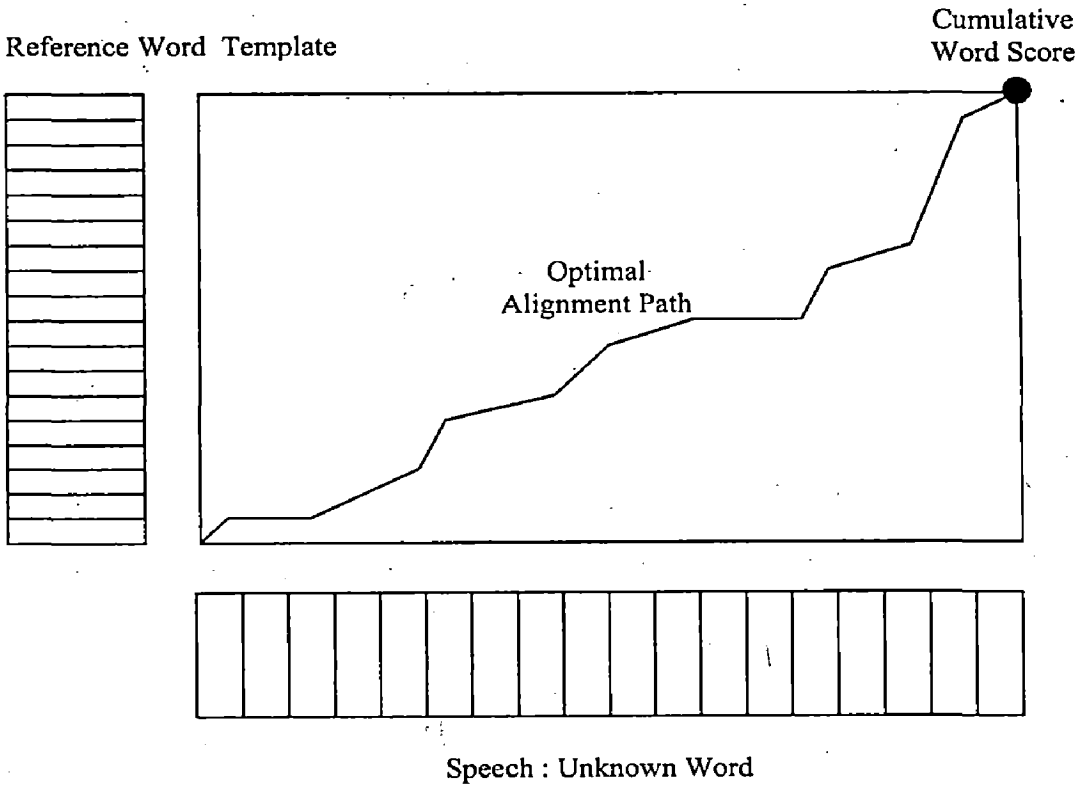


Figure 2.6 DTW Technique

If $D(x,y)$ is the Euclidean distance between frame x of the speech sample and frame y of the reference template, and if $C(x,y)$ is the cumulative score along an optimal alignment path that leads to (x,y), then

$$C(x,y) = MIN(C(x-1,y), C(x-1,y-1), C(x,y-1)) + D(x,y) \qquad (2.5)$$

The resulting alignment path may be visualized as a low valley of Euclidean distance scores, meandering through the hilly landscape of the matrix, beginning at (0, 0) and ending at the final point (X, Y). By keeping track of back pointers, the full alignment path can be recovered by tracing backwards from (X, Y). An optimal alignment path is computed for each reference word template, and the one with the lowest cumulative score is considered to be the best match for the unknown speech sample.

There are many variations on the DTW algorithm. For example, it is common to vary the local path constraints, e.g., by introducing transitions with slope 1/2 or 2, or weighting the transitions in various ways, or applying other kinds of slope constraints. While the reference word models are usually templates, they may be state-based models. When using states, vertical transitions are often disallowed (since there are fewer states than frames), and often the goal is to maximize the cumulative score, rather than to minimize it.

A particularly important variation of DTW is an extension from isolated to continuous speech. This extension is called the *One Stage DTW* algorithm. Here the goal is to find the optimal alignment between the speech sample and the best sequence of reference words . The complexity of the extended algorithm is still linear in the length of the sample and the vocabulary size. The only modification to the basic DTW algorithm is that at the beginning of each reference word model (i.e., its first frame or state), the diagonal path is allowed to point back to the end of all reference word models in the preceding frame.

Local back pointers must specify the reference word model of the preceding point, so that the optimal word sequence can be recovered by tracing backwards from the final point of the word W with the best final score. Grammars can be imposed on continuous speech recognition by restricting the allowed transitions at word boundaries.

2.6 VECTOR QUANTIZATION AND CLUSTERING

While introducing Vector Quantization the first algorithm that comes into our mind is K-means clustering. While considering clustering lot of issues crop up. This can briefly classify clustering as hierarchical clustering, divisive (top-down) clustering, agglomerative (bottom-up) clustering. This concept has a slew of applications in speech recognition

Signal representation produces feature vector sequence which is multi-dimensional sequence. It can be processed by methods that directly model continuous space or by quantizing and modelling of discrete symbols

Main advantages and disadvantages of quantization are reduced storage and computation costs. Potential loss of information due to quantization is a major disadvantage. It is used in signal compression, speech and image coding more efficient information transmission than scalar quantization (can achieve less that 1 bit/parameter) Used for discrete acoustic modelling since early 1980s. Based on standard clustering algorithms Individual cluster centroids are called codewords Set of cluster centroids is called a codebook.

LVQ networks classify input vectors into target classes by using a competitive layer to find subclasses of input vectors, and then combining them into the target classes. Unlike perceptions, LVQ networks can classify any set of input vectors, not just linearly separable sets of input vectors. The only requirement is that the competitive layer must have enough neurons, and each class must be assigned enough competitive neurons.

To ensure that each class is assigned an appropriate amount of competitive neurons, it is important that the target vectors used to initialize the LVQ network have the same distributions of targets as the training data the network is trained on. If this is done, target classes with more vectors will be the union of more subclasses.

Chapter 3

NEURAL NETWORKS IN SPEECH RECOGNITION

3.1 INTRODUCTION

In this section we will briefly review the fundamentals of neural networks and how they are employed in speech recognition. There are many different types of neural networks, but they all have four basic attributes:

- A set of processing units
- A set of connections
- A computing procedure
- A training procedure

Let us now discuss each of these attributes.

3.2 PROCESSING UNITS

A neural network contains a potentially huge number of very simple processing units, roughly analogous to neurons in the brain. All these units operate simultaneously, supporting massive parallelism. All computation in the system is performed by these units and there is no other processor that oversees or coordinates their activity. At each moment in time each unit simply computes a scalar function of its local inputs, and broadcasts the result (called the *activation value*) to its neighbouring units. The units in a network are typically divided into *input units*, which receive data from the environment (such as raw sensory information); *hidden units*, which may internally transform the data representation; and/or *output units*, which represent decisions or control signals .

3.3 CONNECTIONS

The units in a network are organized into a given topology by a set of *connections*, or *weights*, shown as lines in a diagram. Each weight has a real value, typically ranging from $-\infty$ to $+\infty$, although sometimes the range is limited. The value (or *strength*) of a weight describes how much influence a unit has on its neighbor; a

positive weight causes one unit to excite another, while a negative weight causes one unit to inhibit another. Weights are usually one-directional (from input units towards output units), but they may be two-directional (especially when there is no distinction between input and output units).

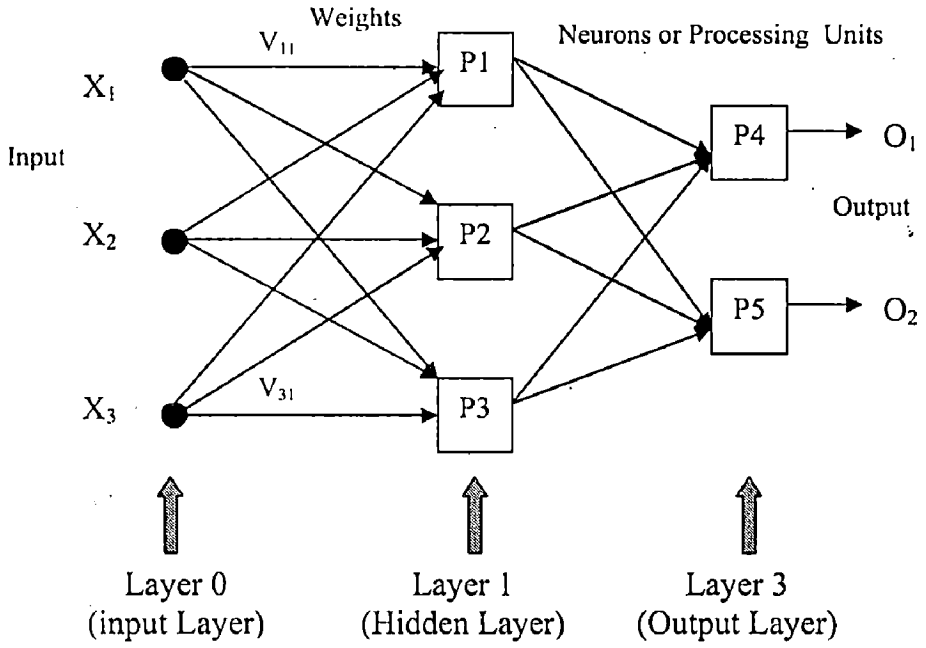


Figure 3.1 Typical Neural Network Architecture

A network can be connected with any kind of topology. Common topologies include unstructured, layered, recurrent, and modular networks, as shown in Figure 2.7 Each kind of topology is best suited to a particular type of application. For example: **unstructured networks** are most useful for pattern completion (i.e., retrieving stored patterns by supplying any part of the pattern) **layered networks** are useful for pattern association (i.e., mapping input vectors to output vectors) **recurrent networks** are useful for pattern sequencing (i.e., following sequences of network activation over time) and **modular networks** are useful for building complex systems from simpler components.

3.4 COMPUTATION

Computation always begins by presenting an input pattern to the network, or *clamping* a pattern of activation on the input units. Then the activations of all of the

remaining units are computed, either *synchronously* (all at once in a parallel system) or *asynchronously* (one at a time, in either randomized or natural order), as the case may be. In unstructured networks, this process is called *spreading activation*; in layered networks, it is called *forward propagation*,as it progresses from the input layer to the output layer. In feedforward networks (i.e., networks without feedback), the activations will stabilize as soon as the computations reach the output layer; but in recurrent networks (i.e., networks with feedback), the activations may never stabilize, but may instead follow a dynamic trajectory through state space, as units are continuously updated.

A given unit is typically updated in two stages: first we compute the unit's *net input* (or internal activation), and then we compute its *output activation* as a function of the net input. In the standard case, as shown in Figure 3.2(a), the net input xj for unit j is just the weighted sum of its inputs:

$$x_j = \sum_i y_i w_{ji} \qquad (2.6)$$

where yi is the output activation of an incoming unit, and wji is the weight from unit i to unit j.

3.5 TRAINING

Training a network, in the most general sense, means adapting its connections so that the network exhibits the desired computational behavior for all input patterns. The process usually involves modifying the weights (moving the hyperplanes/hyperspheres); but sometimes it also involves modifying the actual topology of the network, i.e., adding or deleting connections from the network (adding or deleting hyperplanes/hyperspheres).

In a sense, weight modification is more general than topology modification, since a network with abundant connections can learn to set any of its weights to zero, which has the same effect as deleting such weights. However, topological changes can improve both generalization and the speed of learning, by constraining the class of functions that the network is capable of learning.

Finding a set of weights that will enable a given network to compute a given function is usually a nontrivial procedure. An analytical solution exists only in the simplest case of pattern association, i.e., when the network is linear and the goal is to

map a set of orthogonal input vectors to output vectors. In this case, the weights are given by in general, networks are nonlinear and multilayered, and their weights can be trained only

by an iterative procedure, such as *gradient descent* on a global performance measure. This requires multiple passes of training on the entire training set (rather like a person learning a new skill); each pass is called an *iteration* or an *epoch*. Moreover, since the accumulated knowledge is distributed over all of the weights, the weights must be modified very gently so as not to destroy all the previous learning. A small constant called the *learning rate* (e) is thus used to control the magnitude of weight modifications. Finding a good value for the learning rate is very important — if the value is too small, learning takes forever; but if the value is too large, learning disrupts all the previous knowledge. Unfortunately, there is no analytical method for finding the optimal learning rate; it is usually optimized empirically, by just trying different values.

$$w_{ji} = \sum_{p} \frac{y_i^p t_j^p}{\left\| y^p \right\|^2} \tag{2.6}$$

where y is the input vector, t is the target vector, and p is the pattern index. Most training procedures are essentially variations of the *Hebbian Rule* which reinforces the connection between two units if their output activations are correlated.

3.6 A TAXONOMY OF NEURAL NETWORKS

Now that we have presented the basic elements of neural networks, we will give an overview of some different types of networks. This overview will be organized in terms of the learning procedures used by the networks. There are three main classes of learning procedures:

supervised learning, in which a "teacher" provides output targets for each input pattern, and corrects the network's errors explicitly.

semi-supervised (or **reinforcement**) **learning**, in which a teacher merely indicates whether the network's response to a training pattern is "good" or "bad"; and

unsupervised learning, in which there is no teacher, and the network must find regularities in the training data by itself.

Most networks fall squarely into one of these categories, but there are also various anomalous networks, such as **hybrid networks** which straddle these categories, and **dynamic networks** whose architectures can grow or shrink over time.

3.7 SUPERVISED LEARNING

Supervised learning means that a "teacher" provides output targets for each input pattern, and corrects the network's errors explicitly. This paradigm can be applied to many types of networks, both feedforward and recurrent in nature. We will discuss these two cases separately.

3.7.1 FEEDFORWARD NETWORKS

Perceptrons are the simplest type of feedforward networks that use supervised learning. A perceptron is comprised of binary threshold units arranged into layers. It is trained by the Delta Rule or variations thereof.

3.8 SEMI-SUPERVISED LEARNING

In semi-supervised learning (also called *reinforcement learning*), an external teacher does not provide explicit targets for the network's outputs, but only evaluates the network's behavior as "good" or "bad". Different types of semi-supervised networks are distinguished not so much by their topologies (which are fairly arbitrary), but by the nature of their environment and their learning procedures. The environment may be either static or dynamic, i.e., the definition of "good" behavior may be fixed or it may change over time; likewise, evaluations may either be deterministic or probabilistic. This algorithm assumes stochastic output units which enable the network to try out various behaviors. The problem of semi-supervised learning is reduced to the problem of supervised learning, by setting the training targets to be either the actual outputs or their negations, depending on whether the network's behavior was judged "good" or "bad"; the network is then trained using the Delta Rule, where the targets are compared against the network's mean outputs, and error is back propagated through the network if necessary.

Another approach, which can be applied to either static or dynamic environments, is to introduce an auxiliary network which tries to model the environment . This auxiliary network maps environmental data (consisting of both the input and output of the first network) to a reinforcement signal. Thus, the problem of semi-supervised learning is reduced to two stages of supervised learning with known

targets — first the auxiliary network is trained to properly model the environment, and then backpropagation can be applied through both networks, so that each output of the original network has a distinct error signal coming from the auxiliary network.

A similar approach, which applies only to dynamic environments, is to enhance the auxiliary network so that it becomes a *critic* ,which maps environmental data plus the reinforcement signal to a prediction of the future reinforcement signal. By comparing the expected and actual reinforcement signal, we can determine whether the original network's performance exceeds or falls short of expectation, and we can then reward or punish it accordingly.

3.9 UNSUPERVISED LEARNING

In unsupervised learning, there is no teacher, and a network must detect regularities in the input data by itself. Such *self-organizing* networks can be used for compressing, clustering, quantizing, classifying, or mapping input data. One way to perform unsupervised training is to recast it into the paradigm of supervised training, by designating an artificial target for each input pattern, and applying backpropagation. In particular, we can train a network to reconstruct the input patterns on the output layer, while passing the data through a bottleneck of hidden units. Such a network learns to preserve as much information as possible in the hidden layer; hence the hidden layer becomes a compressed representation of the input data. This type of network is often called an *encoder*, especially when the inputs/outputs are binary vectors. We also say that this network performs *dimensionality reduction.*

Other types of unsupervised networks (usually without hidden units) are trained with Hebbian learning. Hebbian learning can be used, for example, to train a single linear unit to recognize the familiarity of an input pattern, or by extension to train a set of M linear output units to project an input pattern onto the M principal components of the distribution, thus forming a compressed representation of the inputs on the output layer. With linear units, however, the standard Hebb Rule would cause the weights to grow without bounds, hence this rule must be modified to prevent the weights from growing too large.

3.10 EXISTING RECOGNIZERS AND SHORTCOMINGS

At present most of the recognizers are based on Hidden Markov Models. These recognizers are based on statistical techniques .They model the statistical properties of the natural language. They have to be trained using clean data base. These HMM based models give a good performance in noise free conditions but in the presence of noise the performance degrades considerably.

There are purely Neural Network based models. These models are good approximators they give a good performance in presence of noise they. But these recognizers are poor in temporal modeling. We need a recognizer which is good in modeling temporal information and fault tolerant

Chapter 4

ADAPTIVE AND RADIAL BASIS NETWORKS

4.1 INTRODUCTION

Artificial neural networks have been very successful in solving many classification and pattern recognition problems. However, there is always the question of what momentum, learning rate, epoch or sigmoid parameter to use to get the best trained network that could generalize appropriately. Most of the time, this is done by trial and error and using prior experience. Genetic algorithm (GA) was used to determine the optimal control parameters (learning rate, momentum, epoch and sigmoid parameter) for a RBF network and a second GA to optimize the frequencies of the training data. They found that the optimal learning rate and sigmoid parameter were higher than normal heuristic values and that the momentum and epoch optimised to a lower value. GAS is used in the optimization of two parameters, namely the learning rate and momentum which were found to converge to larger than conventional values. GAS[3] was to determine the topology as well as the learning rate and exponential decay (of the learning rate) for a RBF neural network. It was found that high learning rates were optimal and that the network learned much faster with these genetically derived parameters.

One of the main problems of using the RBF neural network as a classifier is that of generalization. If a large representative set of training samples is available, all parts of the decision space are well represented and the trained network is capable of classifying all test samples.

However, if only a small training sample set is available, the network will be trained only by this limited sample set which may not represent all aspects of the decision space of the problem. As such, the network will not accurately classify data which has not been represented in the training set. This is called poor generalization of the network.

One way to overcome poor generalization is to add noise to the training data to prevent over specification. Another method was investigated by Hunt et al [4] who

used fuzzy memberships as an adjunct to RBF to overcome this problem. It was found that the performance of the Fuzzy Radial Basis Neural (FRBNN) network was better especially for problems where there was much overlapping of classes and where the training data sets were quite small.

4.2 ADAPTIVE NEURAL NETWORK

In crisp sets, an element either belongs or does not belong to a set. In fuzzy sets, it is possible for an element to belong partially or wholly to more than one set. A membership value of 0 means that the element is not a member of the set and a value of 1 means that the element belongs entirely in the set. A membership value of 0.3 means that the element belongs partly (0.3) in this set and partly in another [5].

Fuzzy back propagation (FBP) algorithm by incorporates the membership values (p(k)) into the SBP. p(k) is multiplied into the total training error expression before the errors are propagated backwards to initiate a change in weights. The rationale behind this is that the training error should be weighted more if the kth pattern has a high membership value in the class and weighted less if it has a low membership value. Hunt [4] suggested that the membership function could be varied according to the classification problem at hand by the introduction of a fuzzy/concentration parameter m. This fuzzy parameter determines the shape of the membership function.

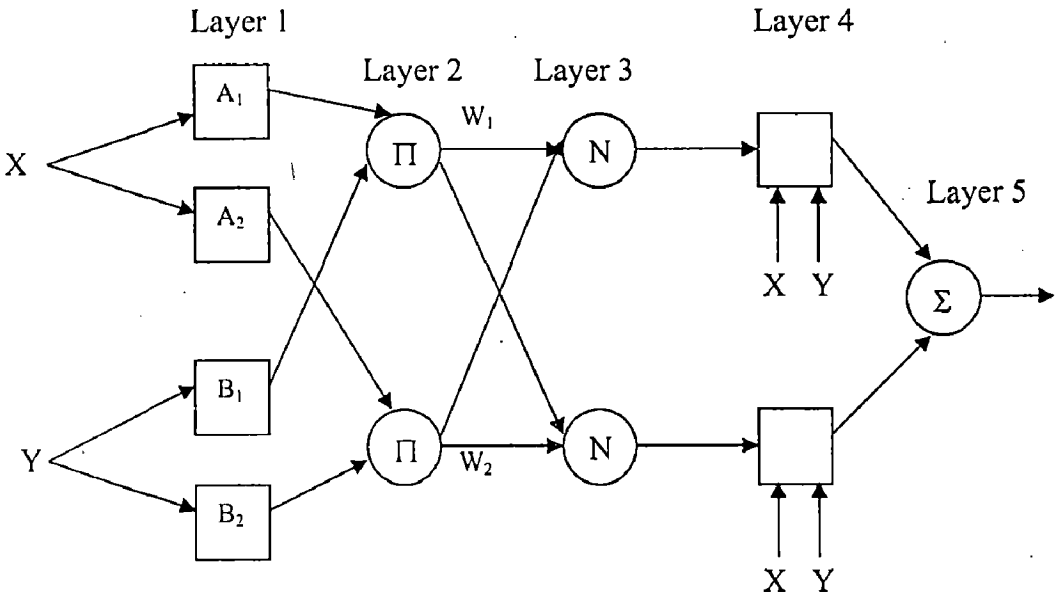


Figure 4.1 Adaptive Neuro-Fuzzy Inference System

4.3 RADIAL BASIS FUNCTIONS NETWORKS

The process of learning an input-output mapping can be regarded as finding an approximation of a multidimensional function. As the set of exampies is finite, the problem can be seen as that of hypersurface reconstruction. This point of view is the motivation behind the regularization networks [3] which include RBFNs as a special case. The two layers of a RBFN have different interpretations. Each neuron in the hidden layer can be seen as a *center* ci, whose coordinates are their weights wi 2 <n. Each center

computes a function that decreases (or increases) monotonically as the distance to that center grows. A common choice is the Gaussian function and a possible distance measure is the squared Euclidean norm:

$$f_j(x) = w_{j0} + \sum_{i=1}^{m} w_{ji}\varphi_i(x) \qquad (4.1)$$

where σ is the *width* associated with the i^{th} center increases rapidly if i is small and slowly if it is large. The neurons of the output layer have a more usual interpretation. Each one computes the following summation: Once the locations and widths of the centers are defined, the network can be seen as a linear model [17], and the weights wi can be calculated either by an algebraic singleshot process or by a gradient descent method as in [24, 25]. Several approaches have been proposed to determine the configurations of the centers. In the seminal work of [3], the centers of the regularization networks were coincident with the input vectors. Alternatively [11],

the input vectors are clustered, and the center of each cluster becomes a network center. The widths of the centers can be determined using various k-nearest-neighbor heuristics. There is, however, a drawback associated with this approach: the optimal location of the centers do not necessarily lie within the convex hull of the training data [15, 25]. A different approach is the *Forward Selection* algorithm [17] and its derivatives [5, 16, 18]. They are similar to connectionist constructive methods. The process starts with an empty subset to which one center (taken from the set of input vectors) is added at a time: the one which most reduces the cost function. This is clearly a hill-climbing method. Although these are effective ways of training a RBFN, there is no guarantee that the configuration found is the best possible combination of

centers and weights. In fact, there is some evidence in the literature that "moving" the centers while determining the weights can improve significantly the performance of the network [24]. Given a deterministic algorithm to compute the weights of the output layer, what is needed, therefore, is an effective way of trying different centers positions and widths. As one cannot test every possible combination, a heuristic is needed, and the GA is a natural candidate.

There are several possibilities of using a GA to configure a RBFN. A straightforward approach is to fix a topology and use the GA as an optimization tool to compute all free-parameters. It has been done in [7] for time-series forecasting. In [1], the number of hidden neurons was also fixed, and the GA optimized only the location of the centers. The k-nearest neighbor heuristic and the singular valûe decomposition computed the widths of the centers and output weights, respectively. Whitehead and Choate [26] also fixed the number of centers, and evolved their locations and widths, but on a completely different and interesting— approach: instead of encoding a network in each individual, the entire set of chromosomes cooperate to constitute a RBFN, each one being a center.

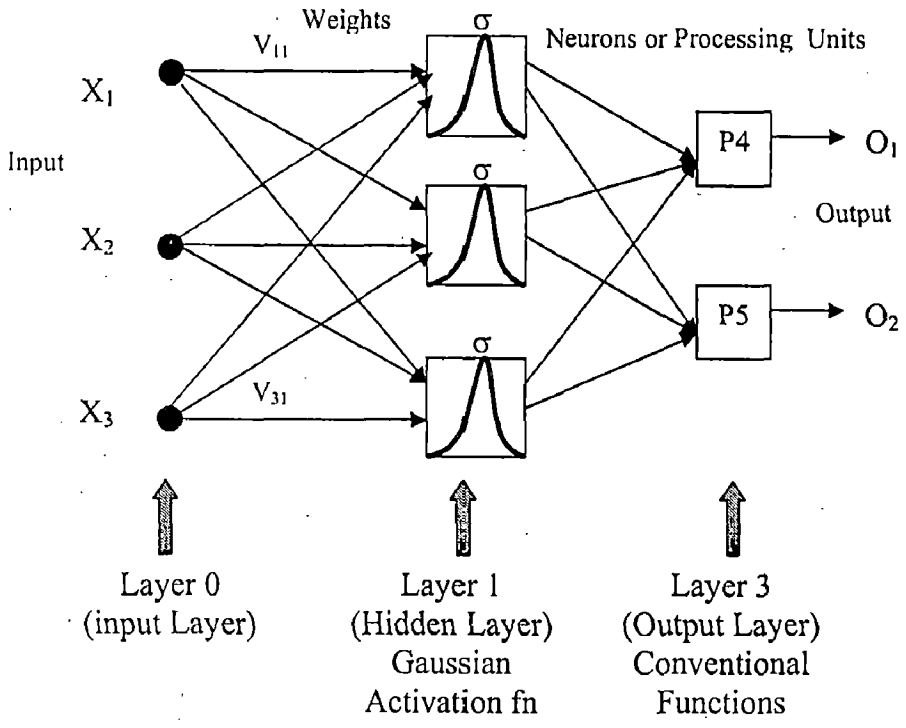


Figure 4.2 Radial Basis Function Neural Network

Another idea is to hybridize the configuration process, using the GA as a support tool. Chen et. al. [8] presented a two-level learning method for RBFN. A regularized orthogonal least squares (ROLS) algorithm1 was employed to construct the RBFNs at the inner level, while the two main parameters of this algorithm were optimized by a GA process at the outer level. In [3], the GA was used to optimize the number and initial positions of the clusters' centers of the k-means algorithm; the RBFN training then proceeded as in [11]. The most common approach (and most promising, in our opinion) is to use a GA to set the network's topology and centers' locations and widths, while the weights in the output layer are computed by an algebraic or gradient-descent method. Naturally, there are many differences between them: for example, an indirect representation is used; the locations of the centers are governed by space-filling curves, whose parameters evolve genetically. Another example is, [4], in which the basis functions are not restricted to Gaussians and are also subjected to evolution. The method proposed here can be included in this category and, like others, has its own peculiarities. They will be presented in the next section.

4.4 RADIAL BASIS FUNCTIONS

The RBF neural network used, has an identical architecture to the MLP with three layers of neurons fully connected. The first layer is responsible for coupling the input vector to the network and have a linear neuron function. The last layer have a number of neurons equivalent to the speaker classes to be identified and uses an adjustable sigmoid as neuron function. It is in the hidden layer that we can find the main difference between the MLP and RBF because it uses special type of neural functions.

These neurons present a neural function with a response that decrease in a monotonic way in relation to a central point, the radial basis function [2], thus making an activation region that forms the basis of the concept of the kernel classifiers. In Table 4.1 the most common radial basis function are illustrated [8]. The use of radial basis functions in the hidden layer defines a nonlinear mapping of the input vector to a projection space, trying to make the input vectors linearly separable. This property of the radial basis functions is described in the Cover's theorem of pattern separation [3] which states that bringing a pattern classification problem into a highly

dimensional space can make observable properties that are hidden in the original space and thus making the problem linearly separable.

Table 4.1 Radial Basis Functions

Function	Formula
Gaussian	$\phi(z) = e^{-z}$
Multiquadratic	$\phi(z) = (1+z)^{\frac{1}{2}}$
Inverse Multiquadratic	$\phi(z) = (1+z)^{-\frac{1}{2}}$
Cauchy	$\phi(z) = (1+z)^{-1}$

In this work the neuron function chosen for the hidden layer is the Gaussian function, making the response of the corresponding neurons to vary inversely with the distance from the function center. The training of a RBF neural network can be divided in two distinct phases: the first one comprises the determination of the radial basis function centers and widths, and the second one consists in updating the connection weights according to the training algorithm. For the determination of RBF centers there are several alternatives, the most used are the K-Means algorithm [11], the LBG [4] and learning vector quantization (LVQ) [12], which are applied over the training set of samples in order to choose or generate a representation for each class of patterns in the problem. Frequently the literature points to the random choice of vectors from training set for building the RBF centers, but this is not a good choice because we can not ensure a good coverage of the sample space. It is preferable to use an algorithm that makes this choice in some statistical way. In this work the centers were obtained using the LBG algorithm and the widths of the RBFs were obtained through a distance analysis of the centers obtained using a Euclidean distance.

Chapter 5

DESIGN AND ANALYSIS

5.1 GENETIC ALGORITHM.

Genetic Algorithms (GAS) are search algorithms based on the mechanics of natural selection and genetics. They were originally developed by John Holland at the University of Michigan in the early 1970's [8]. The aim for their development is to produce algorithms which can solve difficult search problems just as nature has done through 'survival of the fittest'. Individuals in a population are represented by a string of O's and 1's. These individuals compete against one another to survive: in the next generation. This is done by evaluating a fitness value for each individual. This fitness value determines the selection of the next generation. Other genetic mechanisms like crossover and mutation also operate on the new population. Crossover is the process whereby a pair of individuals exchange parts of their genes to form new structures. Mutation is a mechanism whereby a value of one or more of the O's and 1's that make up an individual is changed to form a new structure. Mutation is only introduced at a very small rate but it is necessary in order to ensure that the search space is more thoroughly explored for a potentially better solution [7].

A GA was used to predetermine the control parameters of the SBP and FBP neural network. The genetic algorithm used is GENESIS version 5.0 . Representation of Genes. The SBP structure contains a string of length 33 which represents the sigmoid parameter, learning rate, momentum and epoch respectively.

The FBP structure contains a string of length 42 which represents the fuzzy parameter, sigmoid parameter, learning rate, momentum and epoch respectively .

Ranges.

0 <= fuzzy parameter <= 5.11 (Only for the FBP)

O<= learning rate <= 5.11

0 <= momentum <= 0.99

0 <= epoch <= 256 (This range varies according to the size

of the training data used)

0 <= sigmoid parameter <= 5.11

These ranges have been chosen to reflect normal

heuristic values.

GA Parameters.

Fitness: RMS Error of test & train data

,Population Size: 50

No. of Generations 20

Mutation Rate: 0.01

Crossover Rate: 0.6

Generation Gap: 1.0

The elitist strategy and ranking were used in the GA. No initialization tile was used as early experiments showed that a better solution was reached when the first population was not restricted by the use of an initialization file. Architecture of the neural network A network with 1 hidden layer was used. The number of nodes in the hidden layer was set for each data set. The architecture for the bottle data was chosen from previous experiments done in [23]. All other architectures were selected to be as similar as possible. It is to be noted that to some degree the architecture used is irrelevant since a relative performance is sought rather than an absolute one.

5.2 A FUZZY MEMBERSHIP FUNCTION

A bottom-up, rule-based, acoustic-phonetic decoder retrieves and scores the phonetic hypotheses from a speech signal [3]. To improve recognition performances by restoring the phonetic list, additional knowledge sources (top-down rules) have been determined to verify coarticulation features. Applied to a phonetic hypothesis, each rule returns a numeric parameter related to a fuzzy number via a procedure described in [5] a fuzzy set of rule parameters is made up and the membership function CR () is drawn from one-speaker database histograms.

Let F be the class of unvoiced fricatives. A 400-observation histogram HRl is drawn for correct recognition of F-phonemes by a given rule R, and a 250observation histogram HR2 is drawn for non-F- phonemes recognized as F-phonemes at bottom-up de-coding (the vertical axis is the zero-crossing rate parameters returned by R). The few number of observations and the fact that the rule is only applied to a one-speaker database explain why the statistics are not sufficient for generalization, and why a fuzzy member ship function is needed gives a method to compute a possibilistic function 7ri () from histogram HRi.

The more is the possibility of an erroneous recognition, the less must be the possibility of a correct recognition. Our fuzzy function CR () is more robust to irrelevant histogram variations. In case of null probabilities, the value of ignorance is 0.5. In case of a HR2 peak higher than the corresponding H RI peak, the c:R ()values are under 0.5. The possibility function 7r() is not able to distinguish between ignorance and a higher HR2 peak.

5.3 EVOLUTION

The task assigned to GA consists in searching, through all the rotation axes defined by PCA, the best base of vectors. Evolution is driven by a fitness function defined in term of recognition rate. According to the algorithm given in Figure 5.1, the search of the best projecting space is operated. After the creation of the clean corpus (acoustical analysis + PCA Projected), the ANN is trained until it has converged. The Principal Components used for the projection of the corpus allow the creation of a population of individuals which are able to evolve to adapt to the noisy environment. The best projections basis are selected for the reproduction. Proceed PCA of data in canonical environment Train the ANN on these PCA data.

Individuals are selected by a classical roulette wheel [3] which consist in selecting individual proportionally their fitness. This fitness is computed by the evaluation function. The reproduction is operated by mutation and crossover. As the genotype is encoded with real values, GA operates a gaussian mutation with a standard deviation of 0.1.

```
Generate a population of components
For n generation do
    For each set of Components do
        Project noisy data in the subspace defined by these Components
        Evaluate these Components by Testing the ASRS with the projected
    data
    End For
    Select and Reproduce Components
End For.
```

Figure 5.1 Genetic Algorithm implementation of ASR

The crossover is a simple recombination which consist in selecting a position cut, and flipping the right parts of the genotype as the example below shows:

Table 5.1. Operator of crossover

Parent1 : 01001|1101	Child1 : 01001|0100
Parent 2 :10111|0100	Child2 : 10111|1101

5.4 VECTOR QUANTIZATION

The use of the Kohonen's SOM to train a RBFN is not a new approach. In [2] for example, Kohonen's SOM is employed to find the initial centers of the radial units and, then, a modified Learning Vector Quantization (LVQ2.1) algorithm [17] is utilized to tune all the parameters of the RBFN. Here, the Kohonen's SOM will be applied to train a RBFN employed to pattern recognition in a FDI scheme. Some changes will be made to adequate the Kohonen's SOM in this problem. The first step is to separate the training set according to the different classes. This procedure is adopted to avoid that patterns belonging to different classes are tuned to the same radial unit. Thus, the algorithm described below should be repeated for each class. Initially, all patterns of each class are chosen as radial unit centers. The neuron

activations of all radial units for each training pattern are calculated employing the Equation (5) and the unit with the highest activation is selected according to

$$h_c(t) = \max_j \{h_j(x(t))\} \tag{5.1}$$

where j=1,...,m_k (m_k is the number of patterns in the class k), k=1,...,q (q is the number of classes) and t=1,...,t_{max} is the discrete-time coordinate. The next step is to update the radial unit centers according to

$$\mu_j(t+1) = \mu_j(t) + \alpha(t)\beta(t)[x(t) - \mu_j(t)] \tag{5.2}$$

where $\alpha(t)$ is a decaying function of time that defines the learning rate and $\beta(t)$ is a function of the vector distance from the radial unit center μj to the radial unit center μc. Here, this function is given by

$$\beta(t) = \begin{cases} \dfrac{1}{1 + \left\| R^{-1}(\mu_c - \mu_j) \right\|^2}, if \left\| R^{-1}(\mu_c - \mu_j) \right\| < \sigma(t), \\ 0 \quad otherwise \end{cases} \tag{5.3}$$

where $\sigma(t)$ is a decaying function of time that defines the neighborhood size around the radial unit center μc. If the number of iterations (tmax) is sufficiently large and the training parameters are chosen appropriately, the radial unit centers in the same cluster will move to the cluster center. Thus, as some radial units have centers very near, they will be grouped. This is made calculating the distance between the radial unit centers. If the norm of the distance of two radial unit centers is very small, one radial unit is pruned. Thus, the complexity of the network is reduced because the number of adaptive parameters decreases. This procedure is important because if there are radial unit centers very close, then the matrix inverse used to determine the optimal weights will have ill-posed problems.

Example 1: For the training, two classes with 20 random patterns each (normal distribution) in a 2-dimensional input space are generated. The patterns in the first class are generated with mean=[0.5 0.5]T and variance=[0.01 0.09]T and in the second class with mean=[0 0]T and variance=[0.01 0.09]T. The two training algorithms employ the same input patterns and the diagonal of the matrix R that defines the size of the receptive field is [0.2 0.6]T. The FS uses a fixed threshold to

halt the radial unit selection. The FS algorithm selects two radial units centered in [0.047 - 0.083]T and in [0.578 0.589]T. The Kohonen's SOM algorithm selects two radial units centered in [0.015 0.039]T and in [0.513 0.560]T. For the generalization test, 200 patterns with the same characteristics of the training set are employed. Figure 3 displays the generalization test patterns and the receptive field formed by the RBFN trained with FS and the displays for the RBFN trained with Kohonen's SOM.

5.5 SUMMARY

In this section, the neural network based transformation methods have been evaluated on a large vocabulary continuous speech recognition task in a noisy reverberant enclosure. It has been experimentally proved that the conditional probability of a feature vector given a state is a better optimization criterion than the mean squared error for the synergistic use of neural network and HMM. The MLNN can be combined with traditional neural network that uses the stereo data. The combined networks further improves the performance by doing both feature and model transformation at the same time. The MLNN can also be applied to the unsupervised speaker adaptation.

Chapter 6

RESULTS

6.1 SIGNAL ANALYSIS

Speech data base is composed of three speakers uttering digits from zero to nine. Figure 6.1 is an example of the signal in time domain. These speech files in wav format is fed as input to the signal processing module .The signal processing module extracts the necessary features which are pertinent to speech recognition.

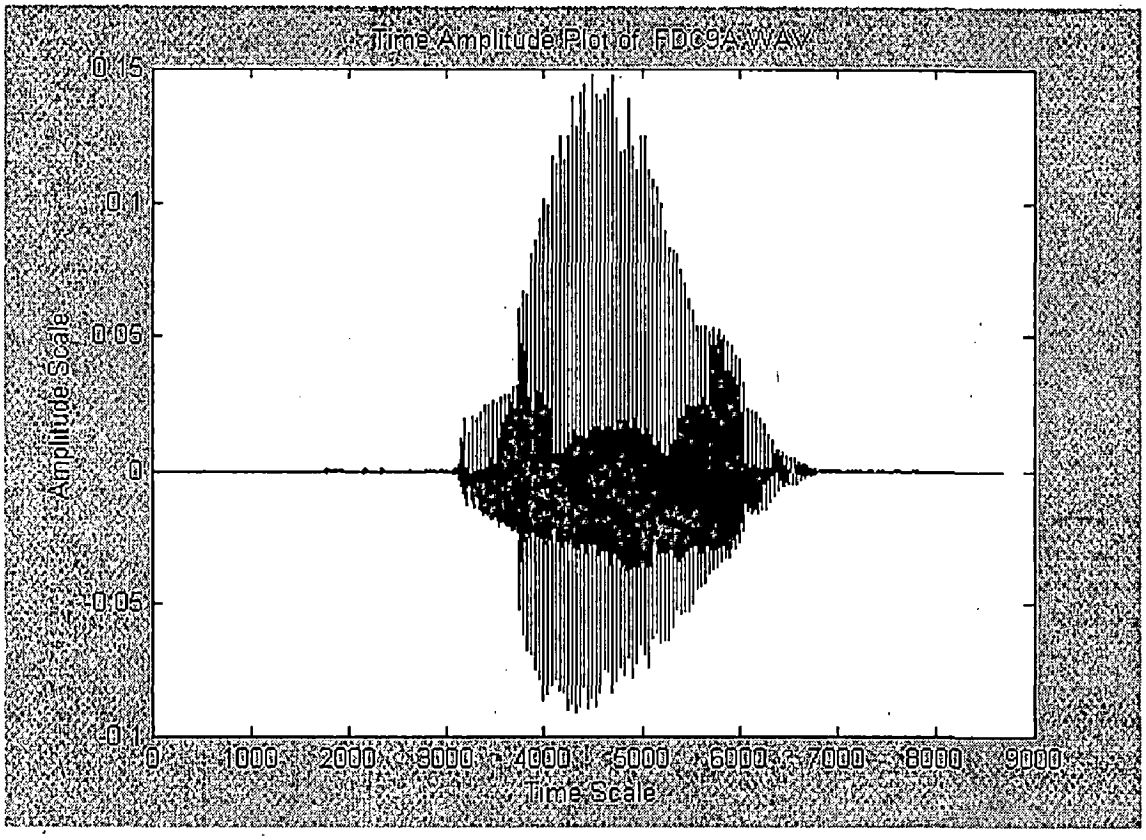


Figure 6.1 Time Domain Signal

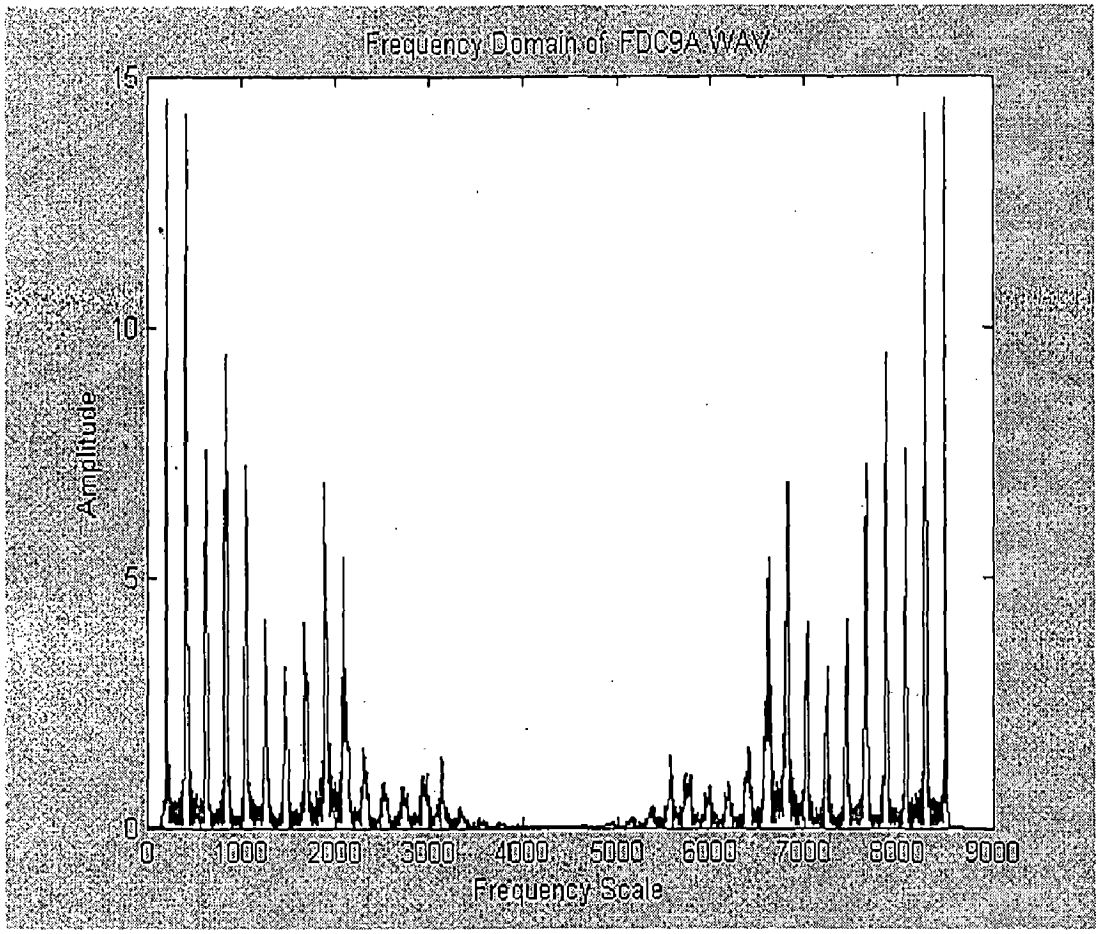


Figure 5.2 Frequency Domain Signal

In the above Figure 5.2 we observe the frequency histograms of the same signal. Here we observe that like in any speech signal most of the energy of the signal is concentrated in the frequency band between 300 – 3000 hz. This signal is also interspersed with noise. The noise is Additive White and Gaussian Noise. As need a "clean" speech signal. We need some mechanism to remove the noise.

Spectrogram is the time by frequency plot of the speech signal the darkness indicates the intensity of the speech signal it forms the third dimension of the signal. The spectrogram gives a clear idea about the speech sound uttered by the speakers.

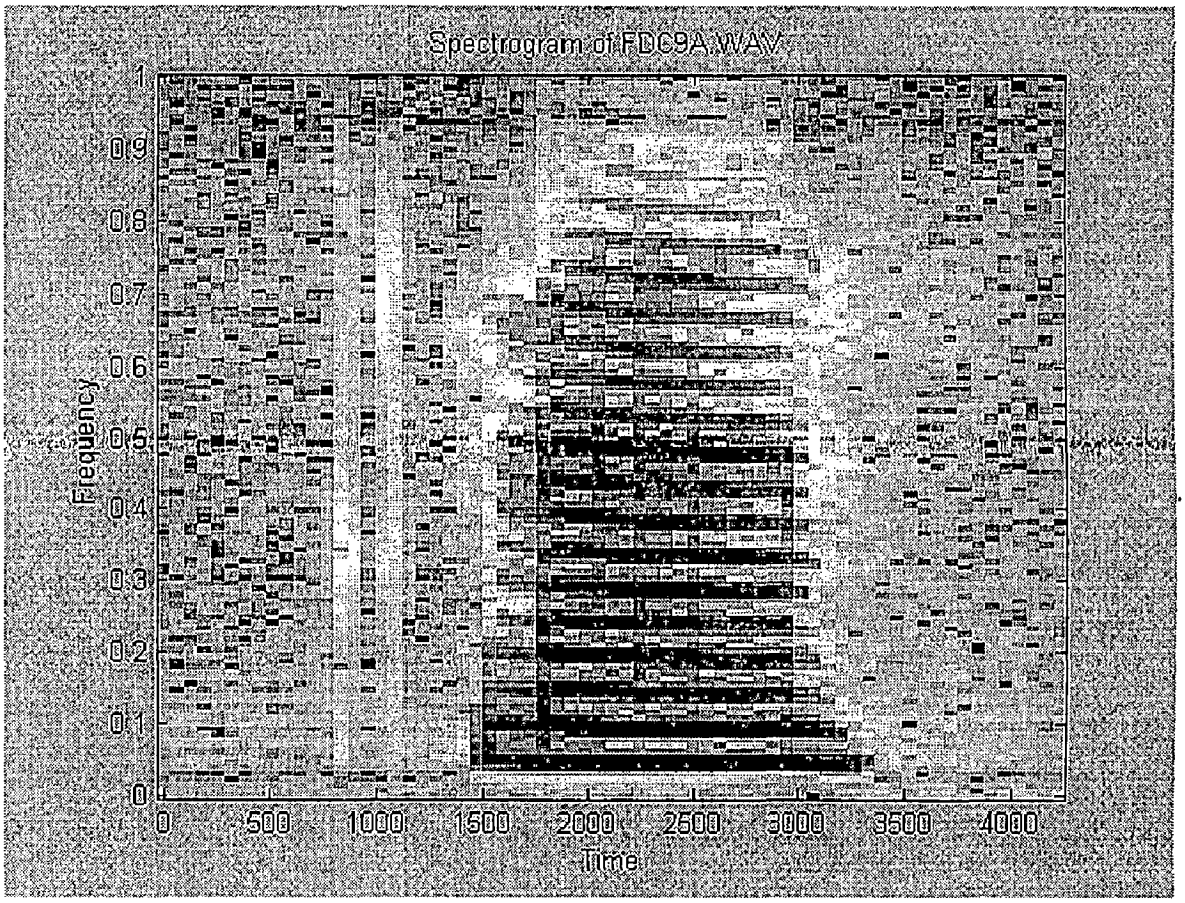


Figure 6.3 Spectrogram of Speech Signal

6.2 PRE-PROCESSING

To train our recognizer we need a set of training data that is free from noise or the maximum signal power to noise ratio should not be less than 30dB. Since it not feasible to simulate such kind of environment in the laboratory we have to explore other techniques. So we introduce adaptive noise cancellation which cancels out the noise. Only this data is used for training.

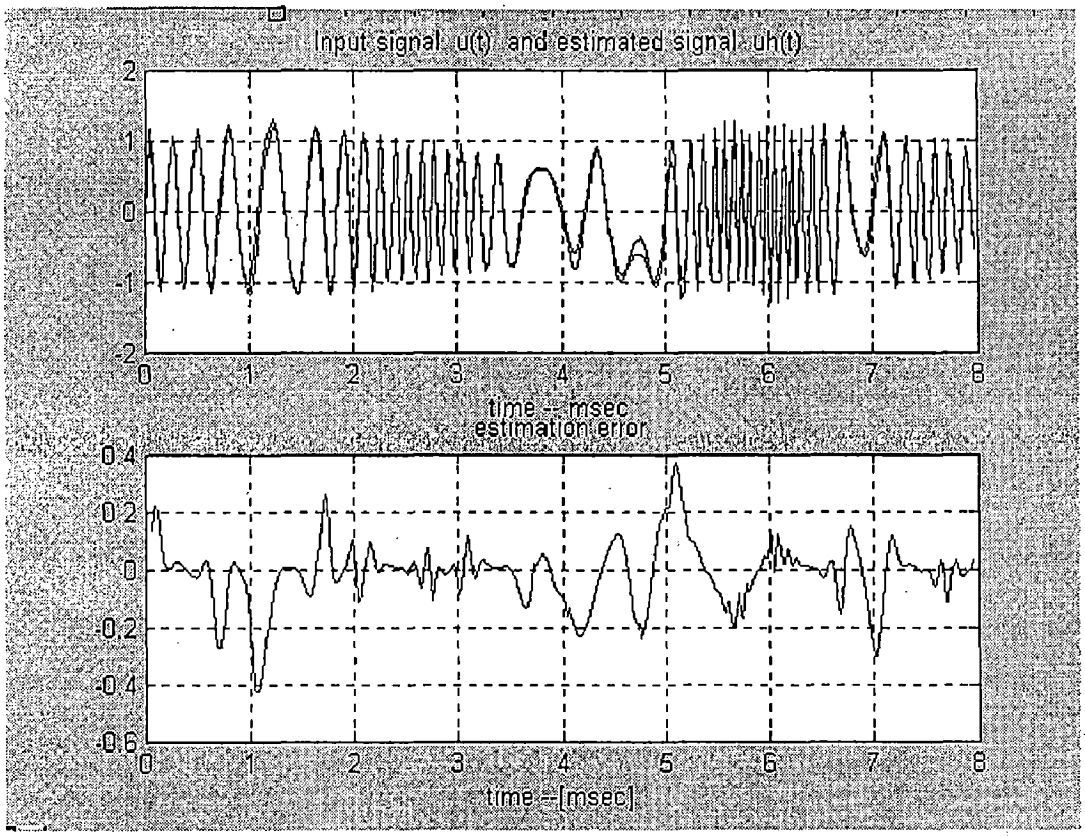


Figure 6.4 Speech Signal with Error after Noise Cancellation

This is achieved by an ANFIS with the following fuzzy membership function

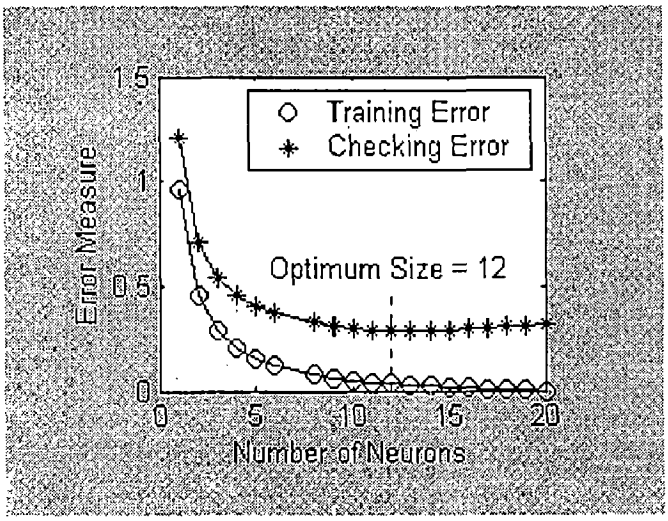


Figure 6.5 Result of ANFIS in noise cancellation

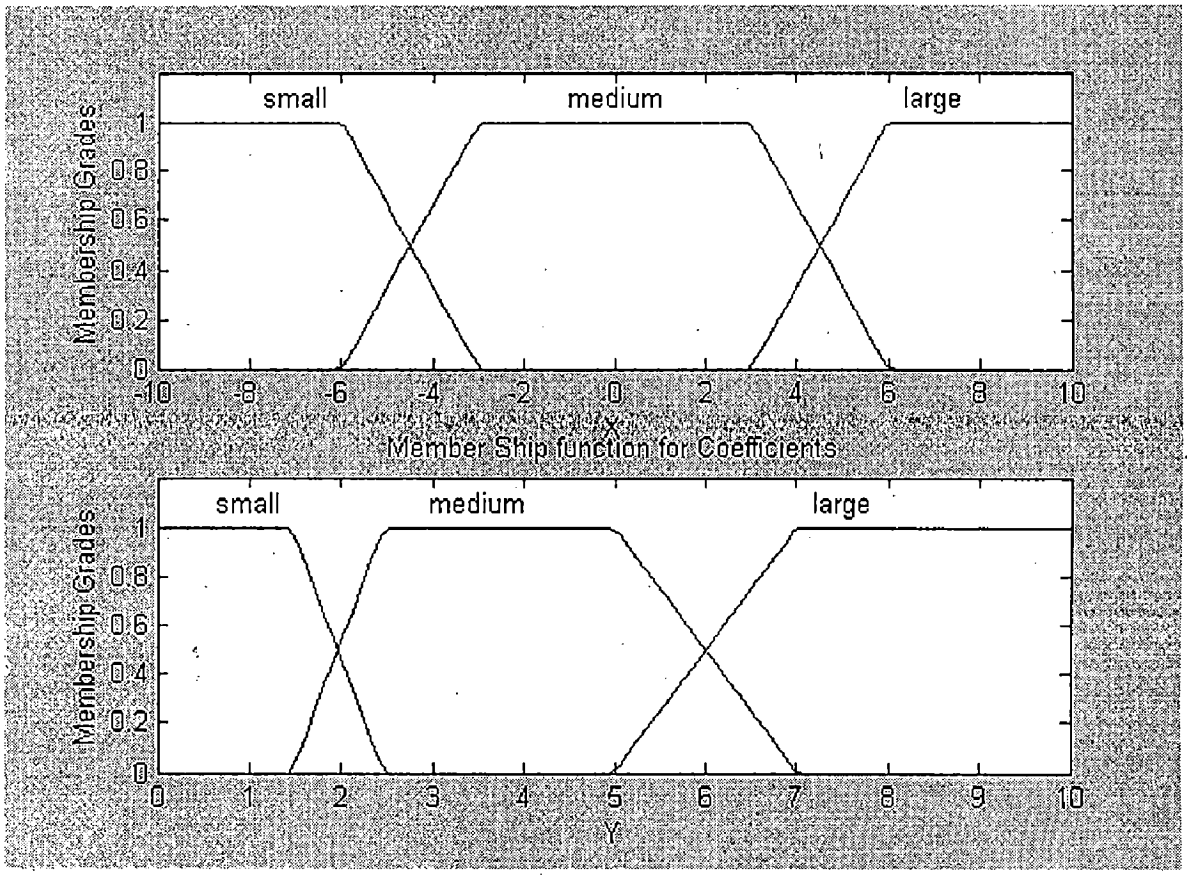


Figure 6.6 Fuzzy Membersip Functions used in noise canceller

6.3 VECTOR QUANTIZATION AND FUZZY C MEANS CLUSTERING

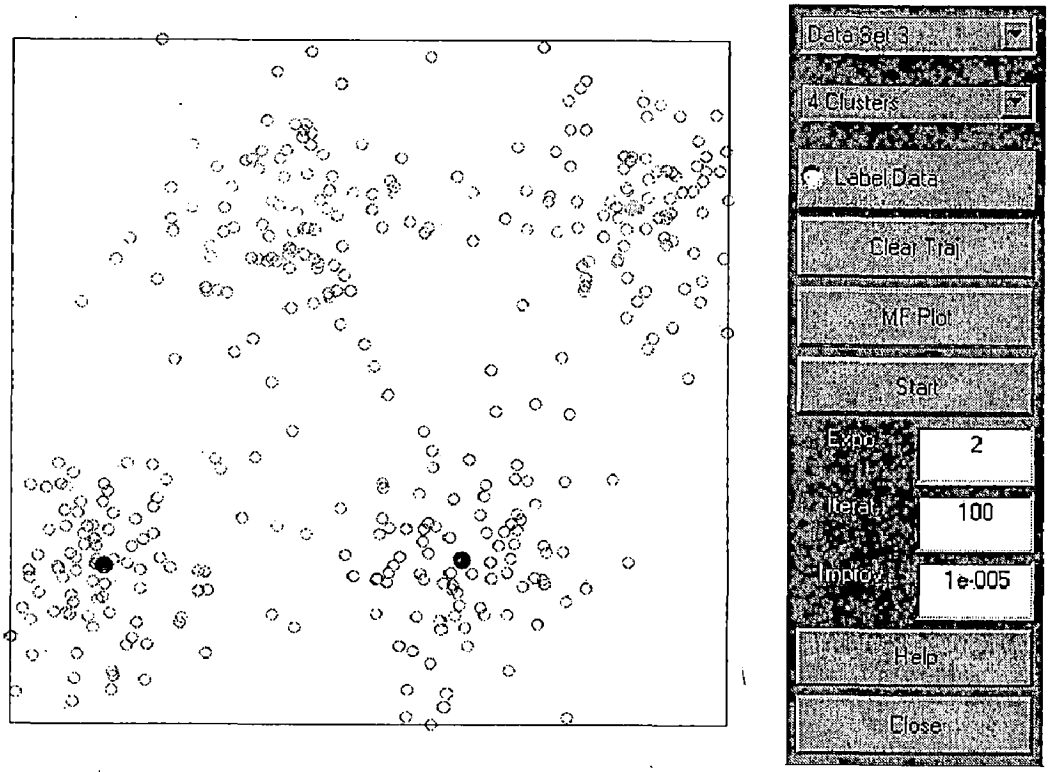


Figure 6.6 Fuzzy C Means clustering

Vector Quantization is employed to reduce the amount of data. Neural based techniques such as learning vector quantizer. Kohonen self organizing map is used .Here Fuzzy C means technique is used. To achieve this we can either use a conventional algorithm such as Lindo Buzo Gray (LBG) algorithm or some self-learning Neural-Network such as LVQ network.

LBG algorithm is time intensive ie. It computes Euclidian Distance between every node. LVQ does not approximate accurately because it uses winner take all paradigm. We find that Fuzzy C Means clustering gives the better performance than the other algorithm because it is fast and approximates fuzzy data well.

6.4 RADIAL-BASIS FUNCTION RECOGNIZER

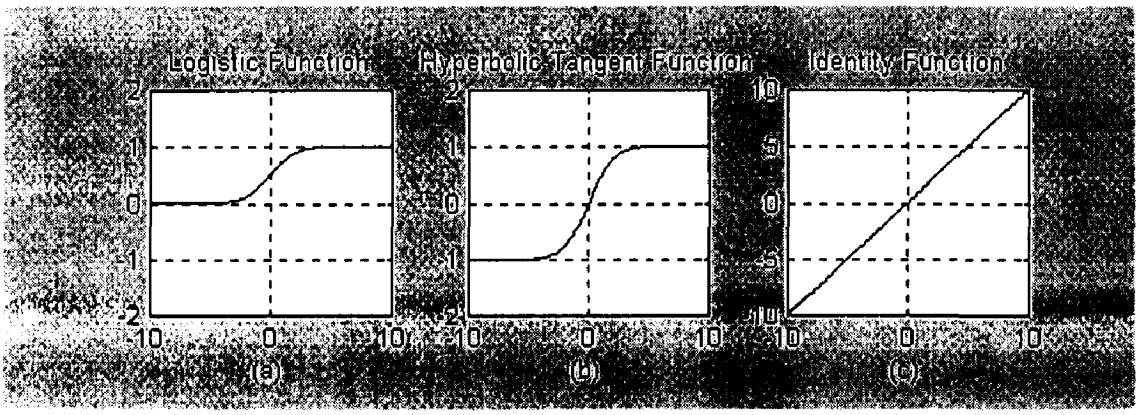


Figure 6.7 Various activation functions used for output layer

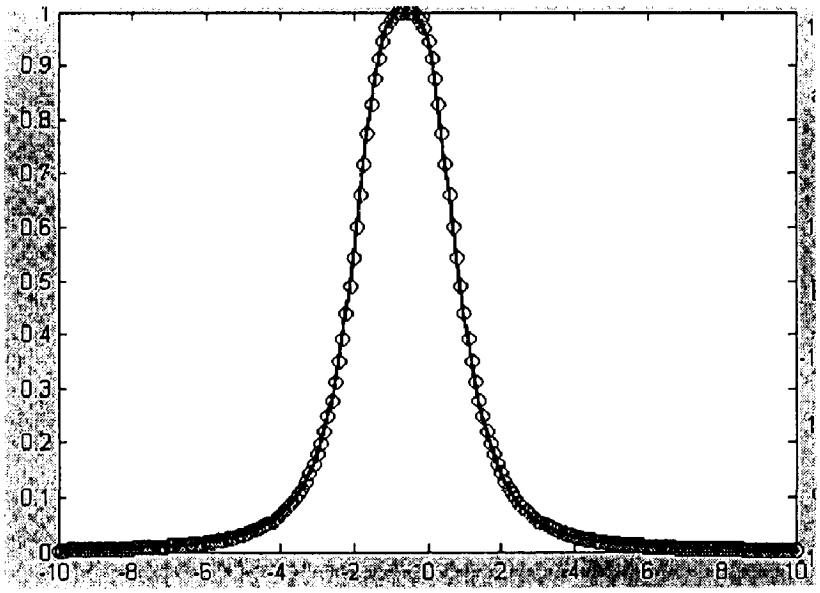


Figure 6.7 Gaussian activation function used for hidden layer

We use the Gaussian function as the radial basis function in the kernel of the neural network. Here by controlling the standard deviation of the Gaussian function and the number of neurons in the hidden layer we can achieve a greater accuracy of approximation.

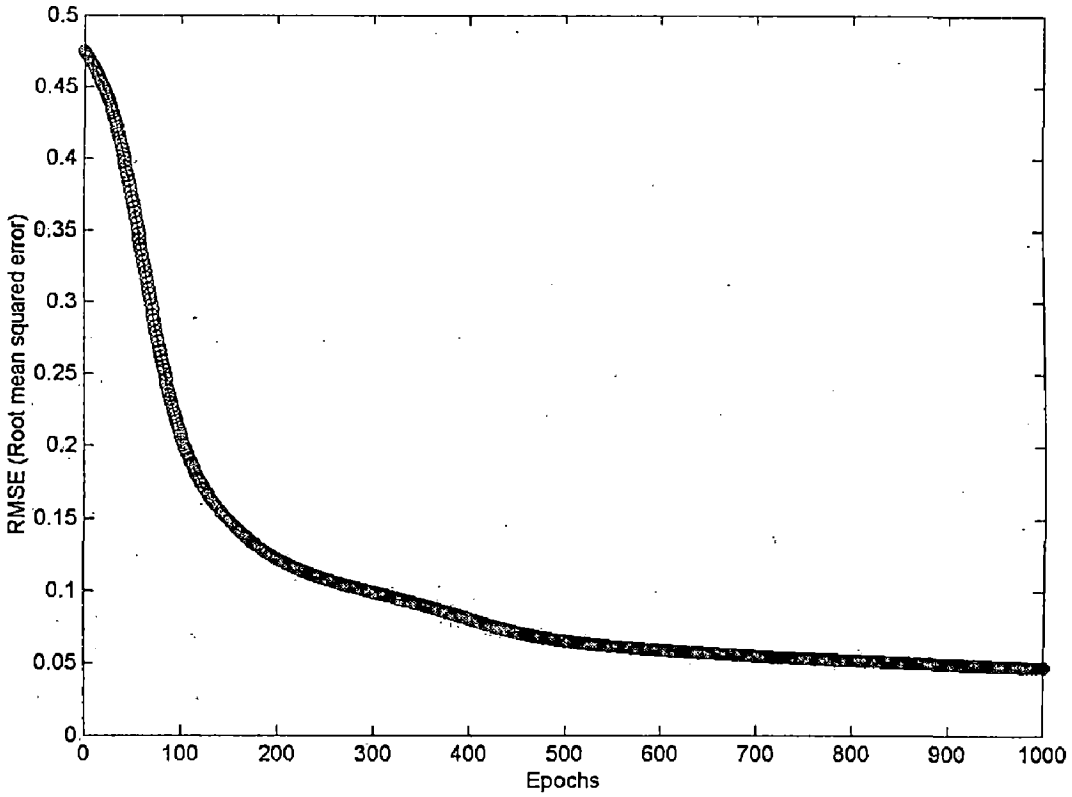


Figure 6.8 Convergence characteristics of RBFNN

In figure 6.8 the convergence characteristics of RBFNN is observed. We need to a larger training pattern set to train the network for accurate recognition. The error rate decreases exponentially for the initial pattern and it slows down as number of patterns are increased.

6.4.1 OPTIMIZATION OF RBFNN USING GENETIC ALGORITHMS

The mean square error has to be decreased and computation time also has to be decreased subject to constraints .The variables in our control are standard deviation of the kernel function and the number of neurons in the hidden layer. The chromosome is encoded using these characteristics. The fitness is computed as a function of the chromosome mapping on to MSE. In figure 6.8 we observe the fitness is increasing as the number of generations increase.

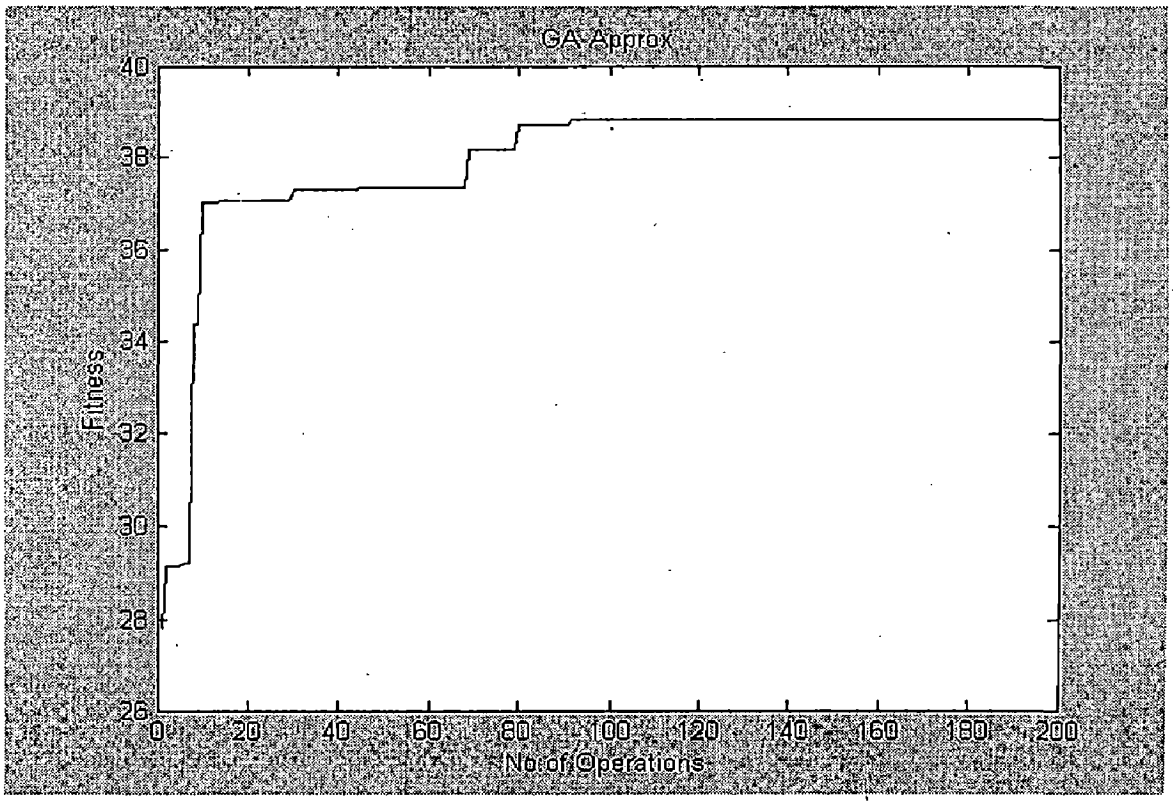


Figure 6.8 Genetic Algorithm for optimizing the architecture of RBFNN

6.5 ANFIS RECOGNIZER

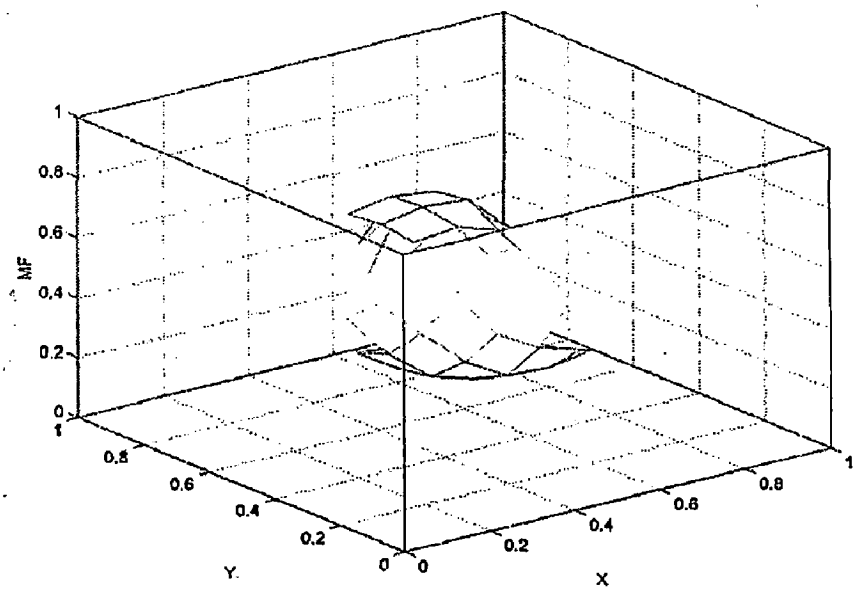


Figure 6.8 Membership Function of ANFIS

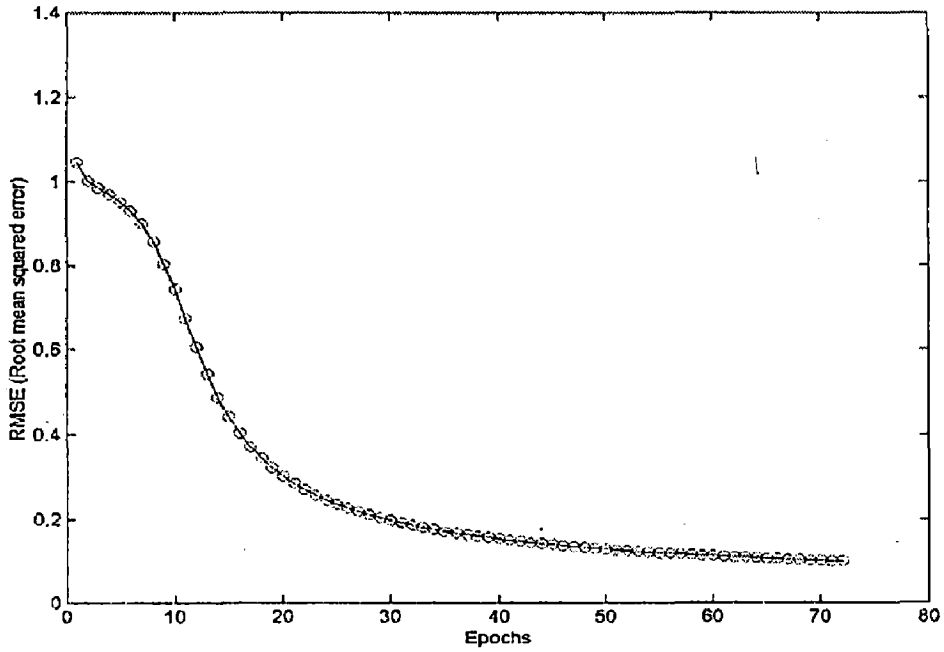


Figure 6.9 Convergence Characteristics of ANFIS Recognizer

It is observed that ANFIS converges faster than RBFNN during training. Here the fuzzy membership function is responsible for speed and accuracy of the recognizer. In figure 6.10 step size of the membership function is changed with epochs. In figure 6.11 step size of the membership function is held constant with number of training epochs. This enables the selection of the right membership function

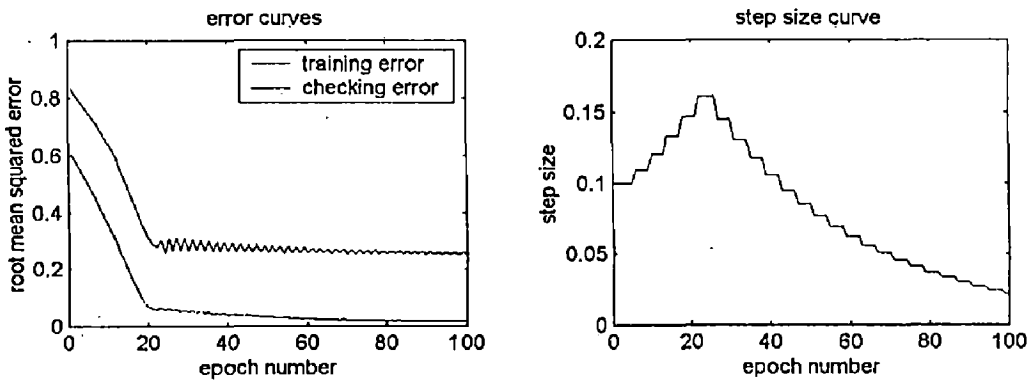


Figure 6.10 Convergence Characteristics and variable Step Size of M F

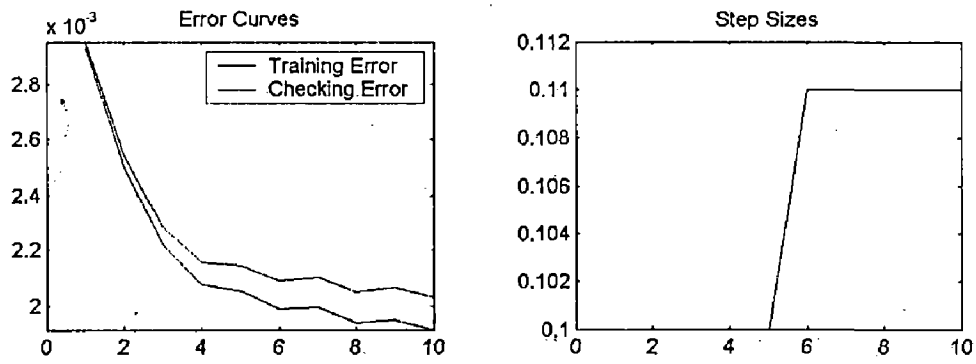


Figure 6.10 Convergence Characteristics and constant Step Size of Membership Function

6.6 COMPARISON OF RECOGNITION ACCURACY

Here it is observed that the ANFIS gives a better performance than RBFNN in terms of recognition in the presence of noise.Since anfis models fuzzy inputs better this accounts for the obtained results

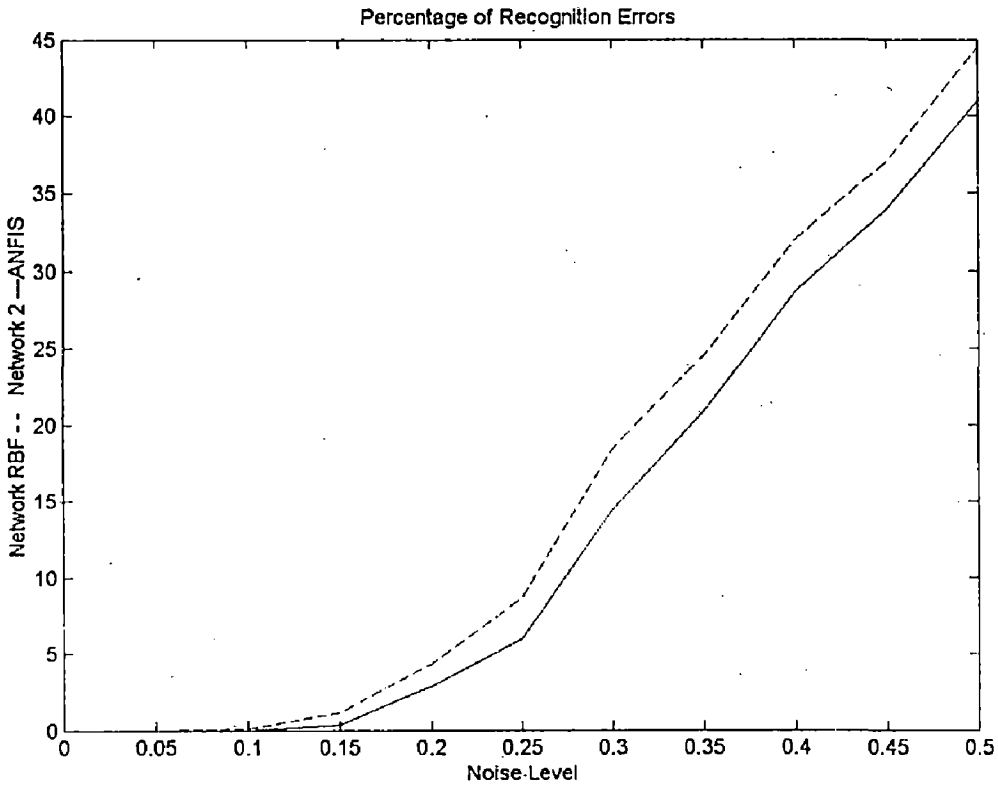


Figure 6.11 Comparison of RBF and ANFIS in the presence of Noise.

Chapter 7

CONCLUSIONS AND FUTURE WORK

With recent advances in speech recognition technology, continuous density Hidden Markov Model (HMM) based speech recognizers have achieved a high level of performance in controlled environments, such as close-talking and matched training and testing acoustical environments. However, the recognition performance is typically degraded if the training and testing environments are not matched. Examples of such mismatches include different ambient noise levels, close talking vs. distant talking, different microphones, and different transmission channels (e.g., telephone speech). To improve performance, speech recognizers are usually trained under the specific application environment where the recognizer will be actually used. However, training a speech recognizer for each particular environment is an expensive and time consuming task in terms of training data collection and computation. Furthermore, because the recognizer is trained in an adverse environment, performance is usually diminished from that in a pristine environment. In this dissertation, a softcomputing based transformation approach for robust speech recognition has been explored.

The neural network, referred to as Radial Basis Function Neural Network (RBFNN), is trained to maximize the likelihood of the speech from the testing environment. Because it requires only a small amount of training data, the proposed approach is especially cost-effective when it is expensive to collect data in a new environment. It therefore permits the recognizer which has been trained once on clean close-talking speech to be used in a wide variety of less favorable environments. The advantages of the approach are as follows. First, it does not require retraining of the speech recognizer, so the expensive task in terms of training data collection and computational time is avoided. Second, it does not require any knowledge about the distortion, yet it automatically learns the mapping function between the training and testing environments. Third, since the Radial Basis Function Neural Network is known to be able to model nonlinear functions, the neural network based approach is able to handle nonlinear distortions. Finally, the feature transformation neural network using stereo data can learn an inverse distortion function, so its performance upper

bound is that of a clean speech recognizer with matched training and testing environments. This bound is typically higher than the recognizer laboriously retrained for the specific environment. Further, the model transformation does not require stereo data. It can be used where the inverse function may not be physically realizable or where the network cannot be well trained with a limited amount of information.

Additive noise causes non-linear distortion in the cepstral domain. A feature transformation neural network that uses stereo data and mean squared error as its objective function has been used to handle the non-linear distortion. From the experiment of continuous speech recognition in adverse acoustical environments, it has been found that this non-linear transformation works well in additive noise case. The anomaly of the traditional mean squared error criterion for the objective function of neural networks has been analyzed. A new objective function for the neural network has been established.

The adaptation can be done in the feature domain or in the model domain. The new objective function has been demonstrated for both feature transformation and model transformation. In feature transformation, feature vectors are transformed to best match a clean speech statistics. In model transformation, both mean vectors and covariance matrices are transformed to best match a testing environment. The tandem combination of feature transformation and model transformation has been established. Feature transformation that uses stereo data can learn complex inverse transformation functions, while the feature transformation RBFNN may not, in practice. The mean and variance transformation RBFNN can learn the distortion function that degrades clean speech statistics. It has been found that the network and the model transformation RBFNN are complementary, and that tandem use of the networks is advantageous.

RBFNN also has its shortcomings. It is slow in execution it cannot model unclear inputs properly. Even though it is flexible it does not respond fast to give a better fit. ANFIS comes in to solve this problem. Not only is ANFIS fast to respond to different input patterns it also models fuzzy inputs properly. It also gives a very good performance in the presence of noise. Only drawback of ANFIS being its complex membership function

Training can be done in an unsupervised fashion by making use of the recognized output for unknown speech. The proposed algorithm has been applied to large vocabulary continuous speech recognition and evaluated under various adverse acoustical environments, which involves background noise, reverberation, differences in microphones, and telephone band limitation. It has also been applied to unsupervised speaker adaptation.

The model transformation RBFNN and ANFIS experiment done in this research uses only one network to transform all means (or variances) of a recognizer. This can be modified to be state-dependent, where each state has its own neural network to transform the parameters. The states can be grouped using tree structure so that those states that do not have enough training data can share a network. The RBFNN and ANFIS are a good candidates for discriminative training methods, because alternative hypotheses or confusable targets can be provided from a speech recognizer. In this case, *mutual information* is a good candidate for the neural network objective functions. In future work these techniques have to be applied for large vocabulary speech recognition systems. A better integration hybrid of HMM and neural networks has to be achieved. We can dramatically improve the speech recognition by using language modeling.

REFERENCES

[1] A. Robinson. An application of recurrent nets to phone probability estimation. *IEEE Transactions on Neural Networks*, 5(2): 298–305, March 1994.

[2] A. Sankar and C. Lee. A maximum likelihood approach to stochastic matching for robust speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4(3):190–202, May 1996.

[3] A.Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):328–339, March 1989.

[4] C. Lee, B. Juang, W. Chou, and J. Molina-Perez. A study on task-independent subword selection and modeling for speech recognition. *International Conference on Spoken Language Processing*, 3:1820–1823, October 1996.

[5] Douglas O' Shaugnessy, "Speech Communication, Human and Machines" Universities Press ,2001.

[6] J.S.R.Jang, C.T.Sun, E.Mizutani, Neuro-Fuzzy And Soft Computing –A Computational Approach to Learning and Machine Intelligence, *Pearson Education* 2004.

[7] J.Wesley Hines, "Fuzzy and Neural Approaches in Engineering" John Wiley & Sons 1997

[8] Jacek M. Zurada , Introduction to Artificial Neural Systems, West Publishing Co,1992

[9] K. Lang and A. Waibel. A time-delay neural network architecture for isolated word recognition. IEEE *Neural Networks*, 3:23–43, 1990.

[10] L. Bahl, S. DeGennaro, P. Gopalakrishnan, and R. Mercer. A fast approximate acoustic match for large vocabulary speech recognition. *IEEE Transactions on Speech and Audio Processing*, 1(1):59–67, January 1993.

[11] L. Barbier and G. Chollet. Robust speech parameters extraction for word recognitionin noise using neural networks. IEEE International Conference on Acoustics, Speech, and Signal Processing, 1:145–148, May 1991.

[12] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*, Pearson Education, 2003.

[13] Leung .H., Chigier. B., and Glass, J., "A Comparative Study of Signal Representation and Classification Techniques for Speech Recognition," *Proc. ICASSP*, Vol. II, 680-683, 1993.

[14] J.-S. Roger Jang. ANFIS: Adaptive-network-based fuzzy inference systems. IEEE Trans. On Systems, Man, and Cybernetics, 1992. Vol I 233-238

[15] Levin, E. (1990). Word Recognition using Hidden Control Neural Architecture. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, 1990.*.vol 1 232-237

[16] Linsker, R. (1986). From Basic Network Principles to Neural Architecture. *Proc. National Academy of Sciences, USA* 83, 7508-12, 8390-94, 8779-83.

[17] Lippmann, R. and Singer, E. (1993). Hybrid Neural Network/HMM Approaches to Wordspotting. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, 1993.*456-459

[18] McDermott, E. and Katagiri, S. (1991). LVQ-Based Shift-Tolerant Phoneme Recognition. *IEEE Trans. on Signal Processing*, 39(6):1398-1411, June 1991.

[19] Mellouk, A. and Gallinari, P. (1993). A Discriminative Neural Prediction System for Speech Recognition. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, 1993.* 256-261

[20] Mermelstein, P. and Davis, S., "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Trans. ASSP*, Vol. ASSP-28, No. 4, 357-366, 1980.

[21] Oppizzi Olivier, Quelavoine Regis .Rescoring under Fuzzy Measures with a Multilayer Neural Network in a rule-based speech recognition system *.Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP '97) 1723-1726

[22] P. Woodland, M. Gales, and D. Pye. Improving environmental robustness in large vocabulary speech recognition. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1:65–68, May 1996.

[23] R. Schwartz, Y. Chow, O. Kimbal, S. Roucos, M. Kransner, and J. Makhoul. Context-dependent modeling for acoustic-phonetic recognition of

continuous speech. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1205–1208, March 1985.

[24] S. Furui. Cepstral analysis technique for automatic speaker verification. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-29(2):254–272, September 1996.

[25] S. Haykin, Neural Networks – A Comprehensive Foundation. Pearson Education Asia 1999.

[26] Thomas F. Quatieri, Discrete-Time Speech Signal Processing –Principles and practice Pearson Education 2004.

[27] Tohkura, Y., "A Weighted Cepstral Distance Measure for Speech Recognition," *IEEE Aug 85*, 86-91

[28] X. Aubert and H. Ney. Large vocabulary continuous speech recognition using word graphs. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1:49–52, May 1995.