



Information Technology and Electrical Engineering



Speech Recognition Using TESPAP and Neural Networks

By

**Whee Kian Ryan Lee
Student no. : 40165008**

**Submitted for the degree of
Bachelor of Engineering**

May 2003

Whee Kian Ryan Lee

3/1 Bellevue Tce

St Lucia, QLD 4067

23rd May 2003

The Head of School

School of Information Technology and Electrical Engineering

University of Queensland

St Lucia QLD 4067

Dear Professor Kaplan,

In accordance with the requirements of the degree of Bachelor of Engineering (Honours) in the division of Electrical Engineering, I present the following thesis entitled "Speech Recognition using TESPAP and Neural Networks" This work was performed under the supervision of Dr John Homer.

I declare that the work submitted in this thesis is my own, except as acknowledged in the text and references, and has not been previously submitted for assessment at the University of Queensland or any other academic institution.

Yours sincerely,

Ryan Lee

Acknowledgement

I would like take this opportunity to express my utmost gratitude to Dr John Homer for his relentless guidance, encouragement, and support throughout the entire thesis project.

Special thanks to a group of friends who have patiently recorded numerous speech samples for this project.

Finally I would also like to thank my granny, parents and sister for their constant love and encouragement throughout my studies in Australia.

Abstract

Speech Recognition has found widespread use in electronic applications. Its appeal lies in its simplicity and together with the ease of operating a device using speech has enormous advantages. However the most modern speech recognition techniques used today have recognition capabilities well below those of a child. These modern techniques are also highly unstable, subjective to noise interferences, require large memory space and complex computations. These factors lead to an increase in production cost of the electronic devices.

TESPAR (Time Encoded Signal Processing and Recognition) and Neural Networks technique is the propose approach to speech recognition system in this thesis. TESPAR is a relatively new approach in the field of speech recognition and its coding is based upon approximations to the locations of the real and complex zeros, derived from an analysis of the bandlimited waveforms. The output stream of symbols from the TESPAR coder can be readily converted into a variety of progressively informative fixed dimension TESPAR matrices. TESPAR matrices are ideally matched to the processing requirements of Artificial Neural Networks for which the used of fixed sized training and interrogation vectors is typically essential [1]. The Multi-Layer Perceptron of the Neural Networks is the proposed approach in the classification stage. In the classification stage, each layer of perceptrons is interrogated separately to produce an individual score. An “Accept” or “Reject” decision is made by examining the total score collected.

The combination of TESPAR and Neural Network approach results in faster computations, higher accuracy and its architectures can be embodied cheaply in DSP devices.

A successful software prototype is being developed to verify the strength of TESPAR and Neural Networks technique. The positive results collected from the assessment stage further substantiate the theories behind TESPAR and Neural Networks.

Table of Contents

Acknowledgements	I
Abstract	II
List of Figures	V
List of Tables	VII
1 Introduction	1
2 Literature Review	5
2.1 General Idea of Speech Recognition	5
2.2 Speech Production	7
2.3 Hearing and Perception	9
2.4 Speech Processing	10
2.4.1 Sampling	10
2.4.2 Quantisation	11
2.4.3 Speech Coding	11
2.5 Approaches to Speech Recognition	12
2.5.1 Pattern Recognition	12
2.5.2 Acoustic-Phonetic Approach	13
2.5.3 Artificial Intelligence	13
2.6 Dynamic Time Warping	14
2.7 Hidden Markov Model	15
3 TESPAP & Neural Networks	16
3.1 Infinite Clipping	16
3.2 Zero-Based Analysis of Signals	17
3.3 TESPAP	17
3.3.1 TESPAP Extraction and Coding	18
3.3.2 TESPAP Alphabets	19
3.3.3 TESPAP Matrices	20

3.4	Neural Network	21
3.4.1	Neural Model	22
3.4.2	Hard Limit Transfer Function	22
3.4.3	Decision Boundary	23
3.4.4	Constructing Learning Rules	24
4	System Operation	27
4.1	Obtaining Speech Waveform	27
4.2	Pre-Extraction Process	28
4.2.1	Quality Process	28
4.3	Extraction Process	30
4.4	Coding Process	31
4.5	Classification using TESPAP and NEURAL NETWORKS	31
4.5.1	Database	32
4.5.2	Neural Network Process	32
4.5.3	Classification Process	33
5	Discussion of Results	34
5.1	Assessment Stage 1	34
5.1.1	Discussion for Assessment stage 1	38
5.1.2	Conclusion on Assessment Stage 1	39
5.2	Assessment Stage 2	39
5.2.1	Discussion for Assessment Stage 2	42
5.2.2	Conclusion on Assessment Stage 2	43
5.3	Assessment Stage 3	43
5.3.1	Discussion for Assessment Stage 3	46
5.3.2	Conclusion on Assessment Stage 3	46
6	Conclusion	47
7	Future Work and Comments	48
	References	51
	Appendices	
A	Program Listings	A1

List of Figures

2.1	Schematic View of the Human Speech Apparatus	8
2.2	Speech Waveform	8
2.3	Schematic Diagram of the Human Ear	9
2.4	Aliasing Distortion by Improperly Sampling	10
2.5	Analog Input and Accumulator Output Waveform	11
2.6	Dynamic Time Warping	14
3.1	Infinite Clipping	16
3.2	TESPAR Symbol Stream	18
3.3	S-Matrix	20
3.4	A-Matrix	21
3.5	Schematic Drawing of Biological Neurons	21
3.6	Single Input Neuron	22
3.7	Hard Limit Transfer Function	23
3.8	Multiple Input Neuron	23
3.9	Perceptron Decision Boundary	23
3.10	Perceptron learning rule	24
3.11	Single Layer Neuron	24
3.12	Graphical Analysis 1	24
3.13	Graphical Analysis 2	24
3.14	Graphical Analysis 3	25
3.15	Graphical Analysis 4	25
3.16	Graphical Analysis 5	26
3.17	Graphical Analysis 6	26
4.1	Speech Waveform Before Centring Process	28
4.2	Speech Waveform After Centring Process	28
4.3	Speech Waveform Before Filtering	29
4.4	Speech Waveform After Filtering	29
4.5	Sampling Waveform	30
4.6	TESPAR Symbol Stream	31
4.7	Multi-Layer Perceptron	32
5.1	Speaker A's reference S-Matrix of the word "Queensland"	35

5.2	Speaker B's reference S-Matrix of the word "Queensland"	35
5.3	Speaker C's reference S-Matrix of the word "Queensland"	35
5.4	Speaker D's reference S-Matrix of the word "Queensland"	35
5.5	S-Matrix of the word "Clock" (1 syllable)	35
5.6	S-Matrix of the word "Nose" (1 syllable)	35
5.7	S-Matrix of the word "Keyboard" (2 syllables)	36
5.8	S-Matrix of the word "Speaker" (2 syllables)	36
5.9	S-Matrix of the word "Logitech" (3 syllables)	36
5.10	S-Matrix of the word "Microphone" (3 syllables)	36
5.11	S-Matrix of the word "Calculator" (4 syllables)	36
5.12	S-Matrix of the word "Evolution" (4 syllables)	36
5.13	1 st sample word "Clock"	40
5.14	2 nd sample word "Clock"	40
5.15	Reference S-Matrix of word "Clock"	40
5.16	1 st sample word "Speaker"	40
5.17	2 nd sample word "Speaker"	40
5.18	Reference S-Matrix of the word "Speaker"	40
5.19	1 st sample word "Microphone"	41
5.20	2 nd sample word "Microphone"	41
5.21	Reference S-Matrix of word "Microphone"	41
5.22	1 st sample word "Evolution"	41
5.23	2 nd sample word "Evolution"	41
5.24	Reference S-Matrix of the word "Evolution"	41
5.25	Reference S-Matrix of the word "Clock"	44
5.26	1 st Speaker (3 rd sample of the word "Clock")	44
5.27	2 nd Speaker (2 nd sample of the word "Clock")	44
5.28	3 rd Speaker (3 rd sample of the word "Clock")	44
5.29	4 th Speaker (1 st sample of the word "Clock")	44
5.30	5 th Speaker (2 nd sample of the word "Clock")	45
5.31	6 th Speaker (1 st sample of the word "Clock")	45
5.32	7 th Speaker (1 st sample of the word "Clock")	45
5.33	8 th Speaker (2 nd sample of the word "Clock")	45
5.34	9 th Speaker (1 st sample of the word "Clock")	45
5.35	10 th Speaker (1 st sample of the word "Clock")	45

List of Tables

3.1	TESPAR Codebook	19
3.2	Data collection for A-Matrix	21
4.1	Score Concept	33
5.1	Scoring Table for Assessment Stage 1	37
5.2	Scoring Table for Assessment Stage 2	42
5.3	Scoring Table for Assessment Stage 3	46

Chapter 1

Introduction

Speech recognition is a process used to recognize speech uttered by a speaker and has been in the field of research for more than five decades since 1950s [6]. Voice communication is the most effective mode of communication used by humans. In this world of communication, humans interact with each other through speech and even the training of animals in the zoo is also done by using speech. As we can see voice is the most natural mode of control as it is fast, hands free and eyes free. Voice of a person is very unique just like fingerprint of human beings.

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

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

Although speech recognition usage has many advantages, the algorithms used in the past to classify it are complicated and expensive. It requires large memory space and enormous power to operate it. Raw speech signals need to be filtered, sampled at twice the bandwidth and converted to digital format using complex Fourier transformation. In addition, raw speech signals are also susceptible to noise interference. Besides the

tedious work in collecting and converting the signals, human speech signals are highly dynamic and stochastic in nature. Every single utterance produce different waveform and this ultimately affects the processing of the speech signals.

TESPAR (Time Encoded Signal Processing and Recognition) combined with Neural Network processing is a relatively new approach to speech recognition. It involves the integration of TESPAR extraction and coding procedures with Neural Network classification procedures. The final “Accept” or “Reject” decision lies in the classification procedure. The past coding procedures are complex and tedious whereas TESPAR coding is simple and straight forward. It has the ability to code speech signals without the need to use complex Fourier transformations. TESPAR has the ability to code time varying speech waveforms into optimum configurations for processing by Neural Networks. TESPAR matrices are of fixed size and dimension, therefore they are ideally matched to the input requirements of Neural Networks [1]. With these two methods combined, the system is able to classify signals that remain indistinguishable in the frequency domain. The use of TESPAR and Neural Networks has opened the door to several useful applications:

- Telephone integrated voice response systems

In this system, caller dialling into a company’s enquiry line does not need to use the phone keypads for answering the questions asked by the automated voice machine. For example, the caller just needs to say “ONE” to access option 1 of the procedure.

- Cellular phones

The small sized and fixed dimension of TESPAR architectures has allowed it to be embodied cheaply into a cellular phone. Users just have to use voice control to make a phone call to anyone without having to key in the phone numbers manually on the keypads. But of course training is needed to be done first to enable the phone system to recognize the user’s voice.

➤ Computers

Activating programs in the computer can be made easy using speech recognition. Users just have to simply say the software name to activate the program. This saves time in searching for the program.

➤ High Security Areas

Because of the unique speech waveform of each individual speaker, speech recognition can be used as a means to prevent any unauthorised access.

➤ Automated Teller Machine

The process of remembering Personal Identification numbers (PIN) can be eliminated. With the uniqueness of voice speech, this has increase security features of the ATM.

➤ Household Appliances

Many household appliances can be controlled by voice control. For example, turning the lights “On” and “Off”, controlling the television, air conditioner and fans.

The applications mentioned above are a few examples of what speech recognition can do to everyday chores. The full potential of TESPAP and Neural Networks can be further exploited in this field of study.

This thesis examines the process involved in TESPAP extracting, coding and classification using Neural Networks. TESPAP and Neural Network approach is broken down into three main processes, namely extraction, coding and classification. Each of these processes are thoroughly examined and discussed in this thesis. A speech recognition program is also developed for substantiating the theories behind TESPAP and Neural Networks.

This thesis report is structured into 6 main sections, namely introduction, theoretical background, technical design, discussion of results, conclusions, recommendation of future work. The introduction written in chapter 1 provides an overview of speech recognition methods used in the past and in this thesis. Objectives of this thesis can be found at the end of chapter 1. The theoretical background of the successful techniques used in the past is explained in Chapter 2. Chapter 3 provides a clear understanding of the TESPAP and Neural Network techniques used in this thesis.

Technical design is provided in Chapter 4. It gives thorough explanations of the design aspects, consideration of the processes and the formulation of the system operations. Presentation and discussion of results are presented in Chapter 5. Discussion and conclusion of results is given in detail. A final conclusion on the results is presented in a well structured statement in Chapter 6.

Finally a few suggestions on improving the system discriminating power can be found in chapter 7. Chapter 7 also includes a personal statement on the student performance in the thesis.

Objectives

The prime objective of this thesis is development of a speaker recognition system using TESPAP and NEURAL NETWORKS. This is achieved by:

- Fully understand the concept on TESPAP and Neural Networks.
- Develop software to accurately code up the speech waveforms using TESPAP so that the coding captures the ‘essential’ characteristics of the original waveform.
- Develop software to convert 10 samples of a speaker’s speech into a reference TESPAP S-Matrix and store in the database.
- Develop software to create a ‘live’ TESPAP S-Matrix of an unknown speaker.
- Develop software to compare this TESPAP S-Matrix with the reference TESPAP S-Matrix in the database.
- Implement a Neural Network Classification Method to compute a “True” detection or “False” Detection decision.

Chapter 2

Literature Review

This chapter focuses on the background information on speech signals and a general summary of the traditional methods used in speech recognition. It begins with a general overview of the background information on speech signals, speech production and speech processing. This is followed by a brief description of the three different approaches to speech recognition. Two powerful methods, the Hidden Markov Model and Dynamic Time Warping will be explained in the following sections.

2.1 General Idea of Speech Recognition

Human speech presents a formidable pattern classification task for speech recognition system [7]. Numerous speech recognition techniques have been formulated yet the very best techniques used today have recognition capabilities well below those of a child. This is due to the fact that human speech is highly dynamic and complex. There are generally several types of disciplines present in the human speech. A basic understanding of these disciplines is needed in order to create an effective system. The following provide a brief description of the disciplines that have been applied to speech recognition problems [7]:

- **Signal Processing**

This process extracts the important information from the speech signal in a well-organised manner. In signal processing, spectral analysis is used to characterize the time varying properties of the speech signal. Several other types of processing are also needed prior to the spectral analysis stage to make the speech signal more accurate and robust.

- Acoustics

The science of understanding the relationship between the physical speech signal and the human vocal tract mechanisms that produce the speech and with which the speech is distinguished.

- Pattern Recognition

A set of coding algorithm used to compute data to create prototypical patterns of a data ensemble. It is used to compare a pair of patterns based on the features extracted from the speech signal.

- Communication and Information Theory

The procedures for estimating parameters of the statistical models and the methods for recognizing the presence of speech patterns.

- Linguistics

This refers to the relationships between sounds, words in a sentence, meaning and logic of spoken words.

- Physiology

This refers to the comprehension of the higher-order mechanisms within the human central nervous system. It is responsible for the production and perception of speech within the human beings.

- Computer Science

The study of effective algorithms for application in software and hardware. For example, the various methods used in a speech recognition system.

- Psychology

The science of understanding the aspects that enables the technology to be used by human beings.

2.2 Speech Production

Speech is the acoustic product of voluntary and well-controlled movement of a vocal mechanism of a human. During the generation of speech, air is inhaled into the human lungs by expanding the rib cage and drawing it in via the nasal cavity, velum and trachea [8]. It is then expelled back into the air by contracting the rib cage and increasing the lung pressure. During the expulsion of air, the air travels from the lungs and passes through vocal cords which are the two symmetric pieces of ligaments and muscles located in the larynx on the trachea. Speech is produced by the vibration of the vocal cords. Before the expulsion of air, the larynx is initially closed. When the pressure produced by the expelled air is sufficient, the vocal cords are pushed apart, allowing air to pass through. The vocal cords close upon the decrease in air flow. This relaxation cycle is repeated with generation frequencies in the range of 80Hz – 300Hz. The generation of this frequency depends on the speaker's age, sex, stress and emotions. This succession of the glottis openings and closure generates quasi-periodic pulses of air after the vocal cords [8]. Figure 2.1 shows the schematic view of the human speech apparatus.

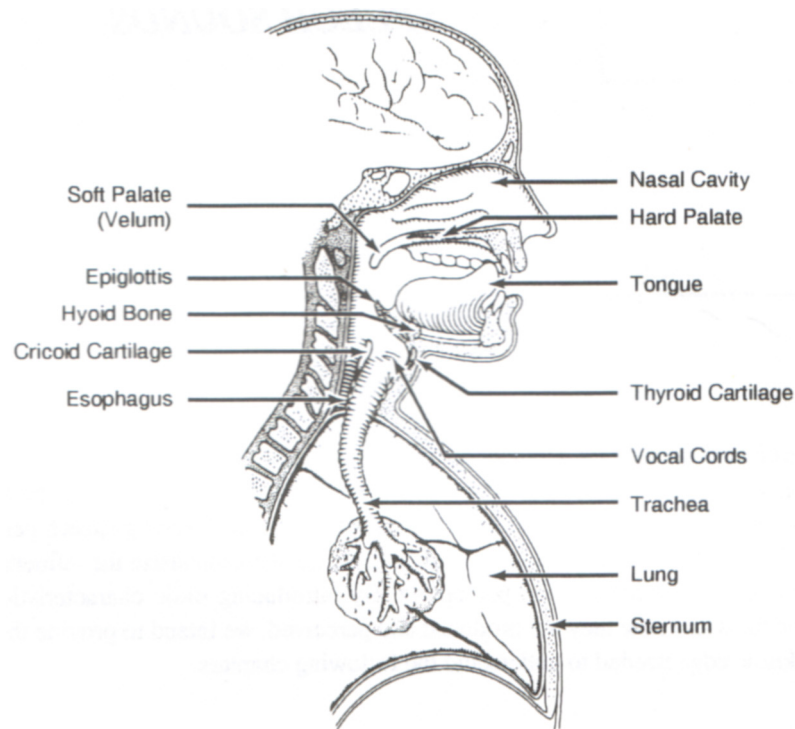


Figure 2.1 Schematic View of the Human Speech Apparatus [8]

The speech signal is a time varying signal whose signal characteristics represent the different speech sounds produced. There are three ways of labelling events in speech. First is the silence state in which no speech is produced. Second state is the unvoiced state in which the vocal cords are not vibrating, thus the output speech waveform is aperiodic and random in nature. The last state is the voiced state in which the vocal cords are vibrating periodically when air is expelled from the lungs. This results in the output speech being quasi-periodic. Figure 2.2 below shows a speech waveform with unvoiced and voiced state.

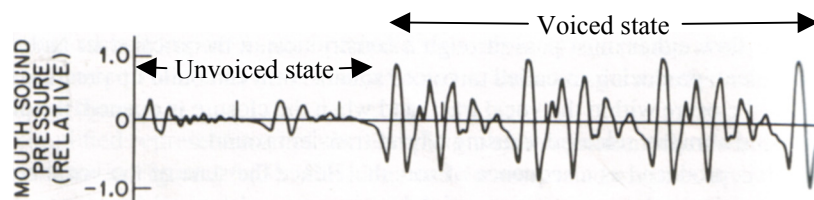


Figure 2.2 Speech Waveform [7]

Speech is produced as a sequence of sounds. The type of sound produced depends on shape of the vocal tract. The vocal tract starts from the opening of the vocal cords to the end of the lips. Its cross sectional area depends on the position of the tongue, lips, jaw and velum. Therefore the tongue, lips, jaw and velum play an important part in the production of speech.

2.3 Hearing and Perception

Audible sounds are transmitted to the human ears through the vibration of the particles in the air. Human ears consist of three parts, the outer ear, the middle ear and the inner ear. The function of the outer ear is to direct speech pressure variations toward the eardrum where the middle ear converts the pressure variations into mechanical motion. The mechanical motion is then transmitted to the inner ear, which transforms these motion into electrical potentials that passes through the auditory nerve, cortex and then to the brain [8]. Figure 2.3 shows the schematic diagram of the human ear.

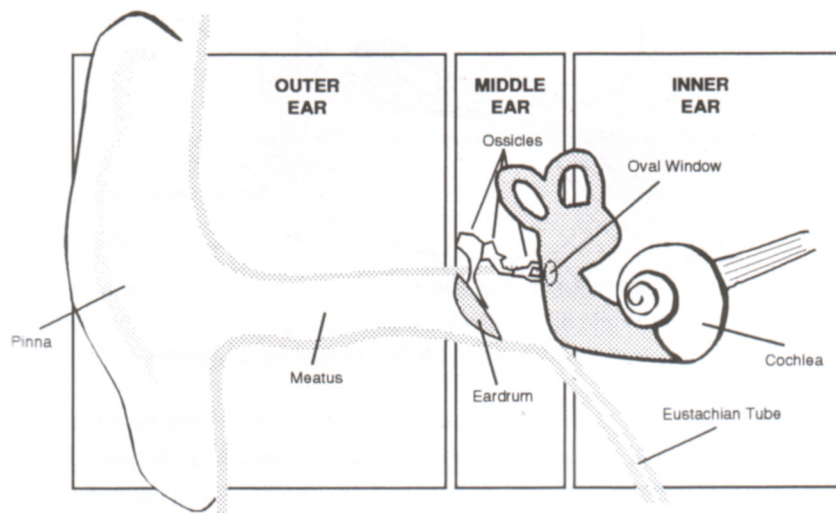


Figure 2.3 Schematic Diagram of the Human Ear [8]

2.4 Speech Processing

The speech waveform needs to be converted into digital format before it is suitable for processing in the speech recognition system. The raw speech waveform is in the analog format before conversion. The conversion of analog signal to digital signal involves three phases, mainly the sampling, quantisation and coding phase. In the sampling phase, the analog signal is being transformed from a waveform that is continuous in time to a discrete signal. A discrete signal refers to the sequence of samples that are discrete in time. In the quantisation phase, an approximate sampled value of a variable is converted into one of the finite values contained in a code set. These two stages allow the speech waveform to be represented by a sequence of values with each of these values belonging to the set of finite values. After passing through the sampling and quantisation stage, the signal is then coded in the coding phase. The signal is usually represented by binary code. These three phases need to be carried out with caution as any miscalculations, over-sampling and quantisation noise will result in loss of information. Below are the problems faced by the three phases.

2.4.1 Sampling

According to the Nyquist Theorem, the minimum sampling rate required is two times the bandwidth of the signal. This minimum sampling frequency is needed for the reconstruction of a bandlimited waveform without error. Aliasing distortion will occur if the minimum sampling rate is not met. Figure 2.4 shows the comparison between a properly sampled case and an improperly sampled case.

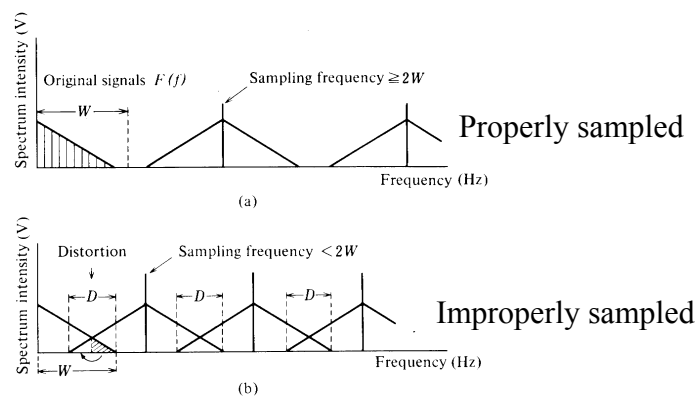


Figure 2.4 Aliasing Distortion by Improperly Sampling [10]

2.4.2 Quantisation

Speech signals are more likely to have amplitude values near zero than at the extreme peak values allowed. For example, in digitising voice, if the peak value allowed is 1V, weak passages may have voltage levels on the order of 0.1V. Speech signals with non-uniform amplitude distribution are likely to experience quantising noise if the step size is not reduced for amplitude values near zero and increased for extremely large values. The quantising noise is known as the granular and slope overload noise. Granular noise occurs when the step size is large for amplitude values near zero. Slope overload noise occurs when the step size is small and cannot keep up with the extremely large amplitude values. To solve the above quantising noise problem, Delta Modulation (DM) is used. Delta Modulation works by reducing the step size for amplitude values near zero and increasing the step size for extremely large amplitude values. Figure 2.5 below shows a diagram on the two types of noises.

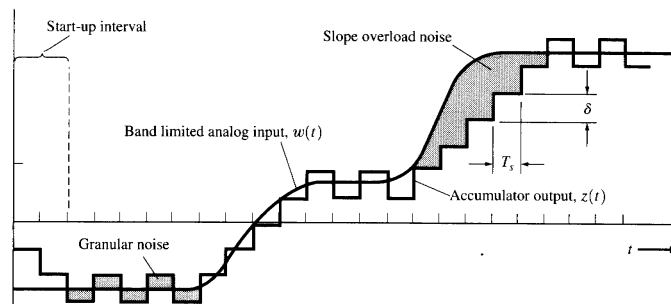


Figure 2.5 Analog Input and Accumulator Output Waveform [9]

2.4.3 Speech Coding

A digital speech coder can be classified into two main categories, mainly waveform coders and vocoders. Waveform coders employ algorithms to encode and decode speech signals so that the system output is an approximation to the input waveform. Vocoders encode speech signals by extracting a set of parameters that are digitised and transmitted to the receiver. This set of digitised parameters is used to set values for parameters in function generators and filters, which in turn synthesize the output speech signals. The vocoder output waveform does not approximate the input waveform signals and may produce an unnatural sound.

2.5 Approaches to Speech Recognition

Human beings are the best “machine” to recognize and understand speech. We are able to combine a wide variety of linguistic knowledge concerned with syntax and semantics and adaptively use this knowledge according to the difficulties and characteristics of the sentences [8]. The speech recognition system is built with this aim in mind to match or exceed human performance. There are generally three approaches to speech recognition, namely, acoustic-phonetic, pattern recognition and the artificial intelligence approach. These three approaches will be explained in greater detail in the following sections.

2.5.1 Pattern Recognition

This direct approach involves manipulating the speech signals directly without explicit feature extraction of the speech signals. There are two stages in this approach, mainly the training of speech patterns and recognition of patterns via pattern comparison. Several identical speech signals are collected and sent to the system via the training procedure. With adequate training, the system is able to characterise the acoustics properties of the pattern. This type of classification is known as the pattern classification. The recognition stage does a direct comparison between the unknown speech signal and the speech signal patterns learned in the training phase. It generates a “accept” or “reject” decision based on the similarity of the two patterns.

Pattern recognition has been the method of choice for many researchers around the world due to the fact that [7]:

1. It is simple to use and the method is fairly easy to understand
2. It has robustness to different speech vocabularies, users, features sets, pattern comparison algorithms and decision rules.
3. It has been proven that this method generates the most accurate results.

2.5.2 Acoustic-Phonetic Approach

The acoustic-phonetic approach has been studied in depth for more than 40 years. It is based on the theory of acoustics phonetics that suggest that there exist finite, distinctive phonetic units of spoken language and that the phonetic units are widely characterized by a set of properties that are manifest in the speech signal, or its spectrum, over time. The first step in this approach is to segment the speech signal into discrete time regions where the acoustics properties of the speech signal are represented by one phonetic unit. The next step is to attach one or more phonetic labels to each segmented region according to the acoustic properties. Finally the last step attempts to determine a valid word from the phonetic labels generated from the first step. This is consistent with the constraints of the speech recognition task [7].

2.5.3 Artificial Intelligence

This approach is a combination of the acoustic-phonetic approach and the pattern recognition approach. It uses the concept and ideas of these two approaches. Artificial intelligence approach attempts to mechanise speech recognition process according to the way a person applies its intelligence in visualizing and analysing. In particular among the techniques used within this class of methods are the use of an expert system for segmentation and labelling so that this crucial and most complicated step can be performed with more than just the acoustic information used by pure acoustic-phonetic methods [7]. Neural Networks are often used in this approach to learn the relationship between the phonetic events and all the known inputs. It can also be used to differentiate similar sound classes.

2.6 Dynamic Time Warping

Dynamic Time Warping is one of the pioneer approaches to speech recognition. It first operates by storing a prototypical version of each word in the vocabulary into the database, then compares incoming speech signals with each word and then takes the closest match. But this poses a problem because it is unlikely that the incoming signals will fall into the constant window spacing defined by the host. For example, the password to a verification system is Queensland. When a user utter “Queeeeeensland”, the simple linear squeezing of this longer password will not match the one in the database. This is due to the longer constant window spacing of the speech “Queeeeeensland”. Dynamic Time Warping solves this problem by computing a non-linear mapping of one signal onto another by minimizing the distances between the two. Thus Dynamic Time Warping (DTW) is a much more robust distance measure for time series, allowing similar shapes to match even if they are out of phase in the time domain.

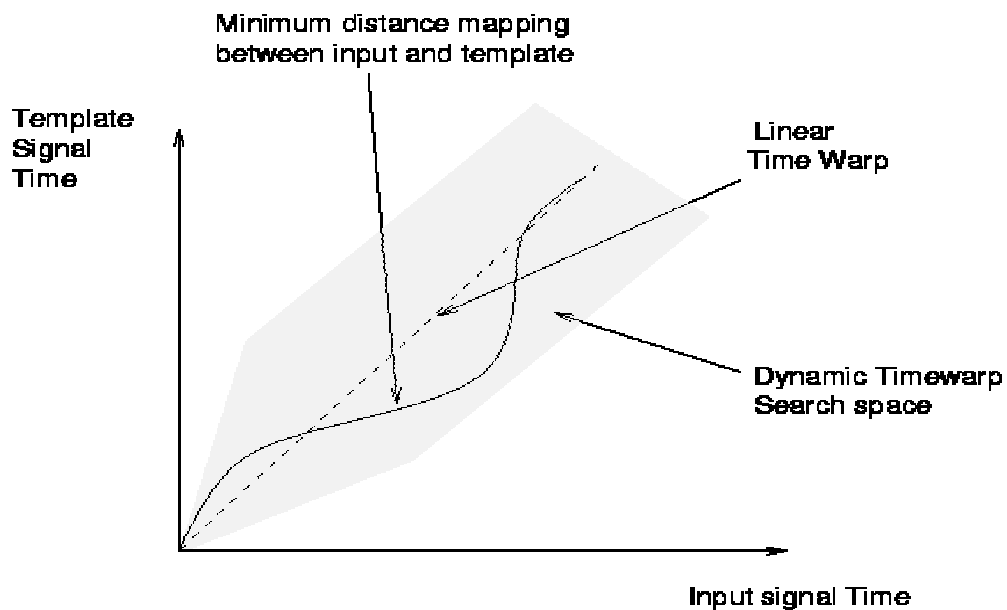


Figure 2.6 Dynamic Time Warping [11]

Figure 2.6 shows the graph on Dynamic Time Warping, where the horizontal axis represents the time sequence of the input stream, and the vertical axis represents the time sequence of the template stream. The path shown results in the minimum distance between the input and template streams. The shaded in area represents the search space for the input time to template time mapping function.

2.7 Hidden Markov Model

Speech Recognition is one of the daunting challenges facing researchers throughout the world. The complete solution is far from over and enormous efforts have been spent by companies to reach the ultimate goal. One of the techniques that gain acceptance from researchers is the state of art, Hidden Markov Model (HMM) technique. This model can also be incorporated with other techniques like Neural Network to form a formidable technique.

The Hidden Markov Model approach is widely used in sequence processing and speech recognition. The key features of the Hidden Markov Model lies in its ability to model temporal statistics of data by introducing a discrete hidden variable that goes through a transition from one time step to the next according to the stochastic transition matrix. Distribution of the emission symbols is embodied in the assumption of the emission probability density.

A Hidden Markov Model may be viewed as a finite machine where the transitions between the states are dependent upon the occurrence of some symbol. Each state transition is associated with an output probability distribution which determines the probability that a symbol will occur during the transition and a transition probability indicating the likelihood of this transition. Several analytical techniques have been developed for estimating these probabilities. These analytical techniques have enabled HMM to become more computationally efficient, robust and flexible [11].

In speech recognition, the HMM model optimises the probability of the training set to detect a particular speech. The probability function is performed by the Viterbi algorithm. This algorithm is a procedure used to determine an optimal state sequence from a given observation sequence.

Chapter 3

TESPAR (Time Encoded Signal Processing and Recognition) and Neural Networks

During the early days, conventional speech recognition approaches were tedious and difficult to implement. In recent years, many evolutionary methods of speech recognition were created to produce simpler and more accurate models. This chapter takes a closer look at how TESPAR and Neural Networks function. The actual process of TESPAR which includes the extraction, coding process are presented in detail in the following sections. The Neural Network process is also explained in greater detail in this chapter.

3.1 Infinite Clipping

Infinite clipping format was originally developed by Licklider and Pollack [2] but before that they were studying the effects of amplitude clipping on the intelligibility of the speech waveform. They extended this format to form the infinite clipping format whereby the amplitude of the speech signal is being removed, preserving only the zero-crossing points of the original speech signal [1]. Subsequently they did a test on the signal with only the zero crossings. Mean random intelligibility scores of 97.9% were achieved. From this test, it appears that the point of interest in a speech signal lies in the zeros crossing of the speech waveform. Figure 3.1 shows an example of infinite clipping.

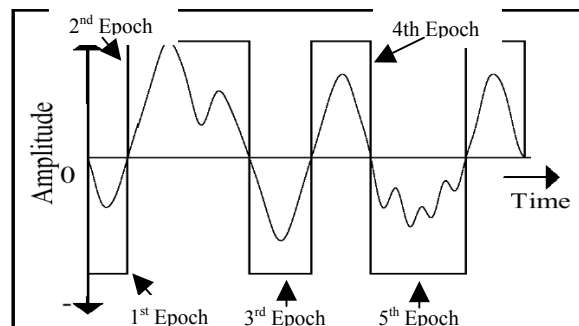


Figure 3.1 Infinite Clipping [1]

Infinite clipping samples represent the time duration of the intervals between the zero-crossings of the speech waveform. These time durations are independent of the regular sampling rate as they depend solely on the speech waveform. In speech waveform, these samples will be spaced randomly. These observations provided a key foundation for the development of Time Encoded Signal Processing and Recognition.

3.2 Zero-Based Analysis of Signals

Experiments carried out by Titchmarsh [3] show that the inadequacies of the infinite clipping format can be overcome by the introduction of complex zeros. However there is a practical problem in zero extraction. The real zeros which correspond to the zero-crossing points of a signal are easy to determine. On the other hand, the complex zeros are much harder to determine. The only method in finding the location of all the complex zeros is by numerical factorisation of a $2TW^{\text{th}}$ -order trigonometric polynomial where W refers to the bandwidth and T refers to the time duration of a waveform [1]. However this calculation is computationally infeasible and this has prevented the further development of the zero model. TESPAPAR uses another method to exploit the zero model in which the complex zeros can be classified directly from the waveform.

3.3 TESPAPAR

TESPAPAR is a new simplified digital language, first proposed by King and Gosling [4] for coding speech signal. Its coding is based upon the approximation of the locations of the real and complex zeros of the speech waveform. The real zeros are represented by the zero crossings of the waveform. The complex zeros are determined by the “shape” of the signal that appear between successive zero crossings. Not all complex zeros can be identified from the shape thus the approximation is limited to those zeros that can be identified.

There are three main processes in speech recognition using TESPAPAR namely the extraction, coding and classification processes. Extraction and coding process will be explained in the next section. The features of the speech waveform are first extracted and subsequently coded into stream of TESPAPAR alphabets. This process of coding is

known as the TESPAP coding process. After the coding process, the stream of symbols is then converted into a variety of fixed dimension TESPAP matrices. The classification is processed through a Neural Network which classifies the speech waveform into “True” or “False” detection.

3.3.1 TESPAP Extraction and Coding

As mentioned above, TESPAP coding is based on the real and complex zeros of the epochs in the speech waveform. An Epoch is the segment between two successive zero crossings as illustrated in Figure 3.3. Duration (D) and Shape (S) parameters are derived from the epoch. The D parameter is derived from the duration between two successive zero crossing. The S parameter corresponds to the shape of the epoch and is represented by the number of positive minimas or negative maximas in the epoch. The selection of positive maximas or negative minimas depends on the polarity of the signal. For an epoch in the positive region, negative minimas are selected whereas for an epoch in the negative region, positive maximas are selected. Figure 3.3 shows the selection of positive minimas or negative maximas.

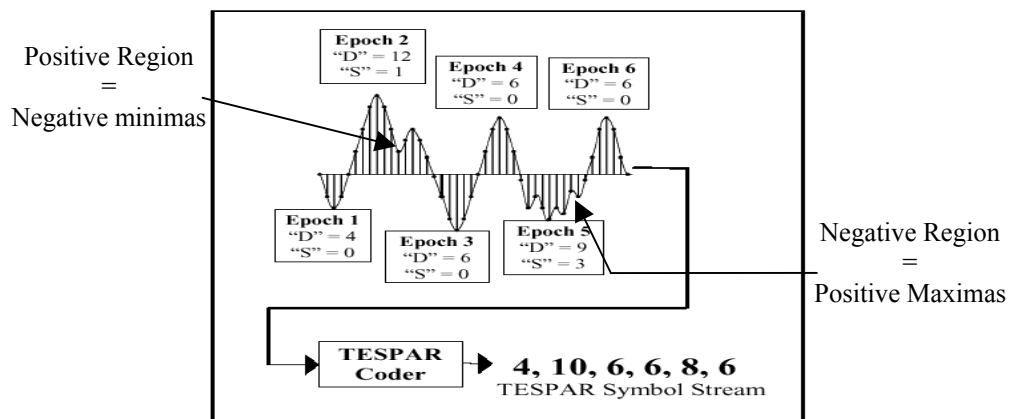


Figure 3.2 TESPAP Symbol Stream [1]

Once the D/S parameters of each epoch are determined, the TESPAP coder pairs them up to generate the TESPAP symbol stream. The symbols are actually elements of the TESPAP alphabet which will be discussed in the next section.

3.3.2 TESPAP Alphabets

As mentioned above the pairing of the D/S parameters is used to generate TESPAP alphabet symbols. A TESPAP codebook comprises of a symbol table of 28 different symbols and is used to map the duration/shape (D/S) parameters of each epoch into a single symbol. For most applications, a standard TESPAP alphabet comprising of 28 different symbols has proven sufficient to represent the original speech waveform. [1]

	S = 0	S = 1	S = 2	S = 3	S = 4	S = 5
D = 1	1					
D = 2	2	2				
D = 3	3	3	3			
D = 4	4	4	4	4		
D = 5	5	5	5	5		
D = 6	6	6	6	6	6	
D = 7	6	6	6	6	6	
D = 8	7	8	8	8	8	8
D = 9	7	8	8	8	8	8
D = 10	7	8	8	8	8	8
D = 11	9	10	10	10	10	10
D = 12	9	10	10	10	10	10
D = 13	9	10	10	10	10	10
D = 14	11	12	13	13	13	13
D = 15	11	12	13	13	13	13
D = 16	11	12	13	13	13	13
D = 17	11	12	13	13	13	13
D = 18	11	12	13	13	13	13
D = 19	14	15	16	17	17	17
D = 20	14	15	16	17	17	17
D = 21	14	15	16	17	17	17
D = 22	14	15	16	17	17	17
D = 23	14	15	16	17	17	17
D = 24	18	19	20	21	22	22
D = 25	18	19	20	21	22	22
D = 26	18	19	20	21	22	22
D = 27	18	19	20	21	22	22
D = 28	18	19	20	21	22	22
D = 29	18	19	20	21	22	22
D = 30	18	19	20	21	22	22
D = 31	23	24	25	26	27	28
D = 32	23	24	25	26	27	28
D = 33	23	24	25	26	27	28
D = 34	23	24	25	26	27	28
D = 35	23	24	25	26	27	28
D = 36	23	24	25	26	27	28
D = 37	23	24	25	26	27	28

Table 3.1 – TESPAP Codebook

3.3.3 TESPAP Matrices

The output symbol stream from TESPAP coder is based on the D/S attributes of corresponding epochs. These symbols can be easily converted into a variety of progressively informative fixed dimension TESPAP matrices. There are two main types of TESPAP matrices, namely the S-Matrix and the A-Matrix.

S-Matrix

S-Matrix is a single-dimension 1x28 vector histogram that records the number of times each TESPAP alphabet symbol appears in the TESPAP symbol stream. The figure shown below is an example of a S-Matrix.

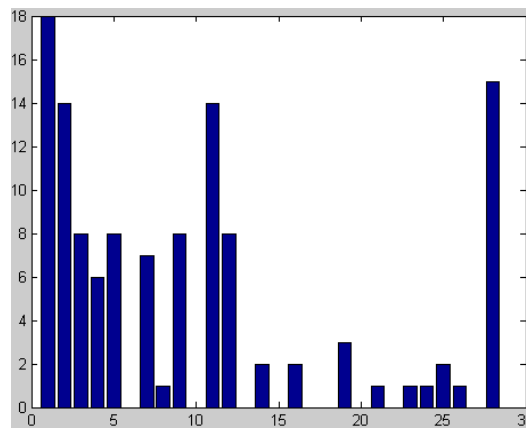


Figure 3.3 S-Matrix

A-Matrix

A-Matrix is a two dimensional 28x28 vector matrix that records the number of times each pair of symbols in the alphabet appears n symbols apart in the symbol stream. [3] This n attribute is known as the delay between symbols. $n < 10$ is used for characterizing the high frequency content of the signal while and $n > 10$ is used for characterizing the low frequency content of the signal. Figure 3.4 below is an example of an A-matrix. Table 3.2 shows an example of how data is collected to form A-matrix.

Example: $n = 2$, TESPAP symbol stream = 4, 8, 9, 7, 4, 13, 9, 2, 8, 5, 7, 4, 10, 9, 1.

	Count		Count
[4,9]	3	[9,8]	1
[8,7]	2	[2,5]	1
[9,4]	1	[5,4]	1
[7,13]	1	[7,10]	1
[13,2]	1	[10,1]	1

Table 3.2 Data collection for A-Matrix

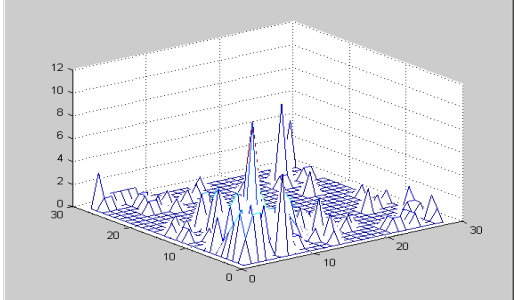


Figure 3.4 A-Matrix

3.4 Neural Networks

As you read these words, your brain is actually using its complex network of 10^{11} neurons to facilitate your readings [5]. Each of these neurons has a blazing processing speed of a microprocessor, which allows us to read, think and write simultaneously. Scientists have found out that all biological neural functions including memory are stored in the neurons and in the connections between them. As we learn new things everyday, new connections are made or modified. Some of these neural structures are defined at birth while others are created everyday and others waste away. In this thesis, the Neural Network algorithm is actually about Artificial Neural Networks and not the actual neurons in our brain. A picture illustrating the biological neurons is shown in Figure 3.5

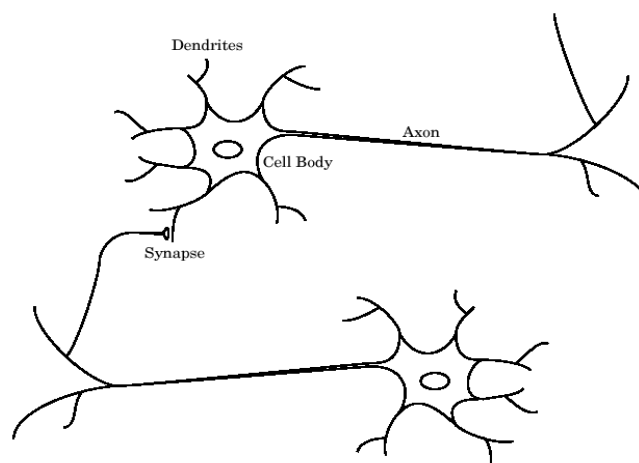


Figure 3.5 Schematic Drawing of Biological Neurons [5]

3.4.1 Neural Model

Figure 3.6 shows a single input neuron. The scalar input p is multiplied by the scalar weight w to form wp which is then sent to the summer. In the summer, the product scalar wp is added to the bias b and passed through the summer [5]. The summer output n , goes into the transfer function f which generates the scalar neuron output a . The value of output neuron “ a ” depends on the type of transfer function used. This whole idea of the artificial neuron is similar to biological neurons shown in Figure 3.5. The weight w corresponds to the strength of the synapse, the cell body is equivalent to the summation and the transfer function and finally the neuron output “ a ” corresponds to the signal travelling in the axon [5].

$$\text{Summer output } n = wp + b$$

$$\text{Neuron output } a = f(wp + b)$$

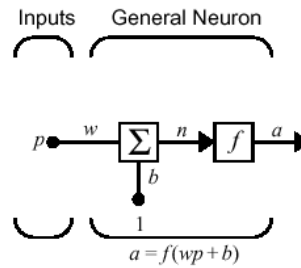


Figure 3.6 Single Input Neuron [5]

3.4.2 Hard Limit Transfer Function

The hard limit transfer function shown on the left side of Figure 3.7 sets the output neuron a to 0 if the summer output n is less than 0. If the summer output n is greater than or equal to 0, it sets the output neuron a to 1. This transfer function is useful in classifying inputs into two categories and in this thesis it is used to determine true or false detection of the speech signal. The figure on the right shows the effect of the weight and the bias combined together.

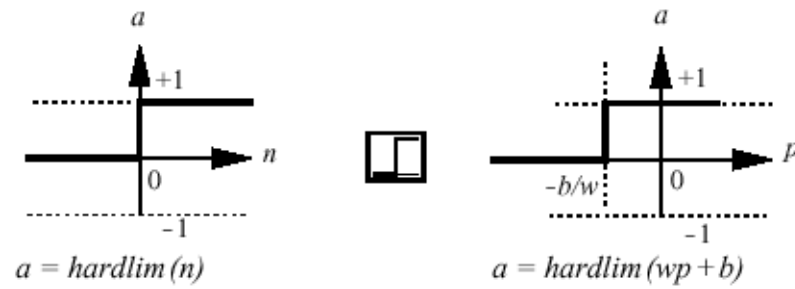


Figure 3.7 Hard Limit Transfer Function [5]

3.4.3 Decision Boundary

A single layer perceptron consists of the input neuron, the weight, bias, summer and the transfer function. Figure 3.8 shows a diagram of a single layer perceptron. A single layer perceptron can be used to classify input vectors into two categories. The weight is always orthogonal to the decision boundary. For example in Figure 3.9, the weight w is set to $[-2 \ 3]$. The decision boundary corresponding to the graph in Figure 3.9 is indicated. We can use any points on decision boundary to find the bias as follows: $wp + b = 0$. Once the bias is set, any point in the plane can be classified as lying inside the shaded region ($wp+b>0$) or outside the shaded region ($wp+b<0$).

$$wp + b = 0$$

$$[-2 \ 3][-2 \ 0]^T + b = 0$$

$$4 + b = 0$$

$$b = -4$$

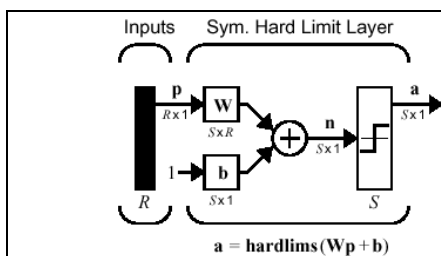


Figure 3.8 Multiple Input Neuron [5]

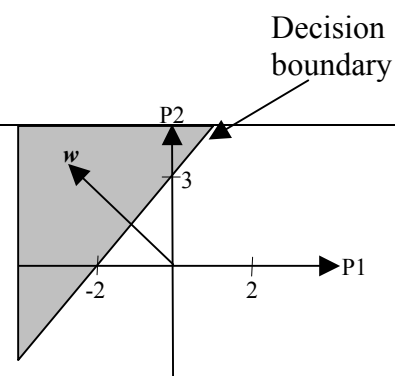
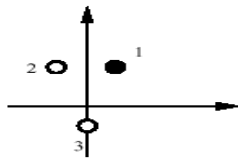


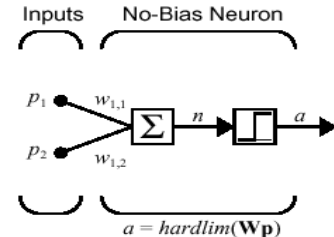
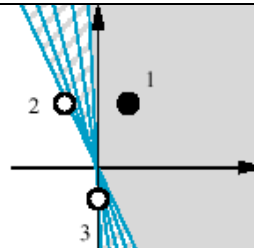
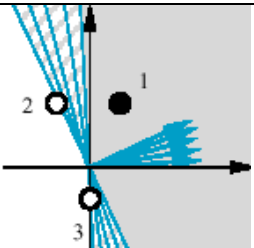
Figure 3.9 Perceptron Decision Boundary [5]

3.4.4 Constructing Learning Rules

In order for a Neural Network to learn how to classify things into categories, certain equations have to be set so that the network can self learn. To further explain the concept of constructing learning rules, below is a test problem to show how the rule works. In a single perceptron network, the inputs for the network are:

 <p>Figure 3.10 Perceptron learning rule [5]</p>	$p_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, t_1 = 1$	$p_2 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}, t_2 = 0$	$p_3 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, t_3 = 0$
--	---	--	--

The problem is display graphically in Figure 3.10, where the two of the inputs, p_2 and p_3 whose target is 0 are represented with a light circle \bigcirc . The other input p_1 whose target is 1 is represented by the dark solid circle \bullet . This is a very simple test problem which will aid us in understanding the concept of the learning rule. Figure 3.11 shows the test problem network with no input bias. This forces the decision boundary to pass through the origin as shown in Figure 3.10 [5].

 <p>Figure 3.11 Single Layer Neuron [5]</p>	 <p>Figure 3.12 Graphical Analysis 1 [5]</p>	 <p>Figure 3.13 Graphical Analysis 2 [5]</p>
---	---	--

There must be an allowable boundary to separate the vectors p_2 and p_3 from the vector p_1 . Figure 3.12 shows that there are infinite numbers of such decision boundaries. Figure 3.13 shows the numerous weight vectors that corresponds to the numerous decision boundaries. The length of the weight vector is not important only its direction is important in separating the inputs. First, a weight vector is randomly generated.

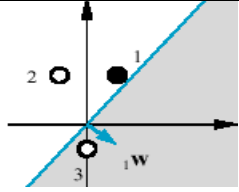
$${}_1w^T = [1.0 \ -0.8]$$

When input p_1 is introduced into the network, it generates an output of:

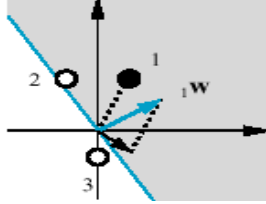
$$a = \text{hardlim}({}_1w^T p_1) = \text{hardlim} \left([1.0 \ -0.8] \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right)$$

$$a = \text{hardlim}(-0.6) = 0$$

The network has returned an incorrect value of 0 as the target response of t_1 is 1. Once the network returned an incorrect value, the learning rule would be used to rectify the problem.

<p style="text-align: center;">Learning Rule Equations</p> <p>If $t = 1$ and $a = 0$, then ${}_1w^{new} = {}_1w^{old} + p$ - Equation 3.1</p> <p>If $t = 0$ and $a = 1$, then ${}_1w^{new} = {}_1w^{old} - p$ - Equation 3.2</p> <p>If $t = a$, then ${}_1w^{new} = {}_1w^{old}$ - Equation 3.3</p>	 <p style="text-align: center;">Figure 3.14 Graphical Analysis 3 [5]</p>
--	---

In the first incorrect case of $t = 0$ and $a = 1$, equation 3.1 is used to rectify the problem. After applying this equation to the test problem, the new value for the weight is :

${}_1w^{new} = {}_1w^{old} + p_1 = \begin{bmatrix} 1.0 \\ -0.8 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2.0 \\ 1.2 \end{bmatrix}$	 <p style="text-align: center;">Figure 3.15 Graphical Analysis 4 [5]</p>
---	--

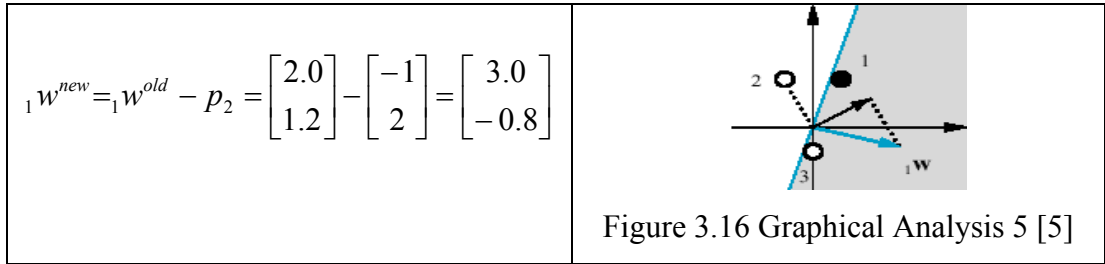
The shifting of the new weight can be seen in Figure 3.15. The new weight would then be applied to input p_2 . This modification to the weight and cycling through the inputs continues until the output target of all the inputs are classified correctly.

Next when input p_2 is presented to the network with the new weight, it generates:

$$a = \text{hardlim}({}_1w^T p_2) = \text{hardlim} \left([2.0 \ 1.8] \begin{bmatrix} -1 \\ 2 \end{bmatrix} \right)$$

$$a = \text{hardlim}(0.4) = 1$$

Again the network has misclassified the output. The target t_2 is equal to 0 but the output of the network is 1. Equation 3.2 is now being employed to rectify the error.



The shifting of the weight can be seen from Figure 3.16.

Next when input p_3 is presented to the network with the new weight, it generates:

$$a = \text{hardlim}({}_1w^T p_3) = \text{hardlim} \left(\begin{bmatrix} 3.0 & -0.8 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \end{bmatrix} \right)$$

$$a = \text{hardlim}(0.8) = 1$$

Again the network has misclassified the output. The target t_3 is equal to 0 but the output of the network is 1. Equation 3.2 is again being employed to rectify the error.

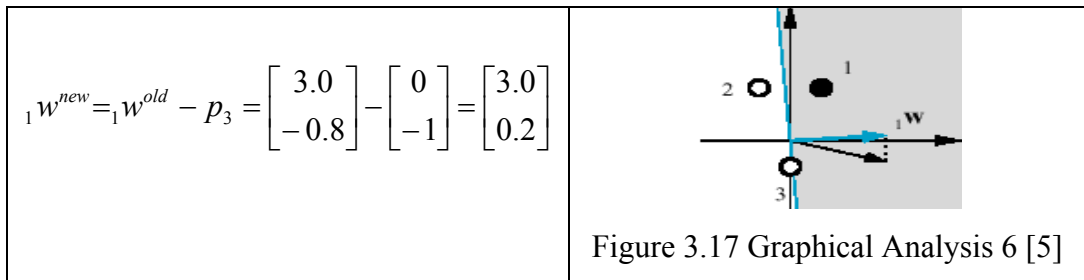


Figure 3.17 shows that the perceptron has finally learned to classify the 3 vectors properly. Now any inputs when presented will be classifying accordingly. For example, when input p_2 is presented to the network with the new weight, it generates:

$$a = \text{hardlim}({}_1w^T p_2) = \text{hardlim} \left(\begin{bmatrix} 3.0 & 0.2 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \end{bmatrix} \right)$$

$$a = \text{hardlim}(-2.6) = 0$$

The target t_2 is equal to 0 and the output of the network is 0. This corresponds to equation 3.3 and the weight need not be modified anymore. Although the perceptron learning rule is simple to implement, it is quite powerful too. It has proven that the rule will always converge to weights that accomplish the desired classification. [5]

Chapter 4

System Operation

This chapter explains the design and operation of the speech recognition program. The program is divided into three main processes, namely the extraction, coding and the classification process. Before the actual extraction process takes place, two other processes need to be completed. The first task is to collect 10 speech samples from each designated speaker and then send them through the compatibility and quality process. After which they are loaded into the program that extract the features (D / S parameters) of the signal. Once the D/S parameters are generated, they are then subjected to the coding process in which they are being converted to TESPAP alphabet symbols and subsequently to an S-Matrix. The 10 S-Matrices of each designated speaker are then averaged out to reduce the inconsistency. The mean S-Matrix from each designated speaker passes through the Neural Network to construct individual weights and biases. The weights and biases are then stored into the database. The program is now ready to classify any incoming speech signal. An unknown speaker speech is recorded and passes through all the processes mentioned above. Once the S-Matrix of the unknown speaker is generated, it will pass through the Neural Network and the network will determine if the unknown speaker belongs to any of the designated speakers in the database.

4.1 Obtaining Speech Waveform

The first task is to record the speech waveform from the speaker and upload it into the TESPAP program. The sound Recorder program in Microsoft Windows is chosen to record the speech waveform. The recorded speech is automatically filtered, sampled at a sampling rate of 22.05 KHz and then saved as a wave file. Wave file format is chosen because it is highly compatible with the Matlab program as it can be easily retrieved via a single command.

4.2 Pre-Extraction Process

The speech wave file is loaded into the Matlab program by using a “wave read” function which limits the amplitude of the speech signal to a magnitude of 1. The signal is then saved as an $M \times 1$ vector where M refers to the total number of samples in the speech signal. Each element in the M vector contains the amplitude of the speech signal at a particular sampling instant. The speech signal is now ready to go through the compatibility and quality process.

4.2.1 Quality Process

Before the actual extraction process takes place, the wave file is subjected to a series of processes to ensure the compatibility and quality of the signal. When the speech signal is being loaded into the Matlab program, the signal is not centered at the $y = 0$ axis. In order to bring the whole signal to centre on the zero-line, special program code was written. This code is used to find the mean of the signal and then subtract this mean from each of the sample values of the signal. This is shown in Figure 4.1 and 4.2 below. The reason for shifting the whole signal to the $y = 0$ axis will be explained in section 4.3.

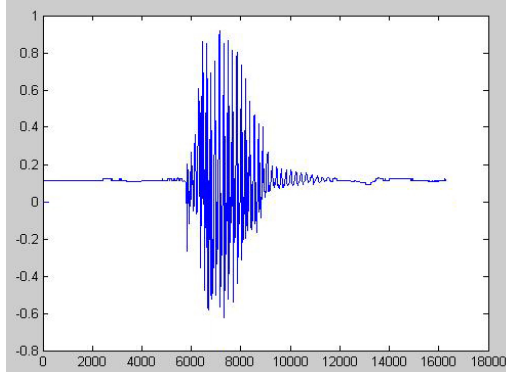
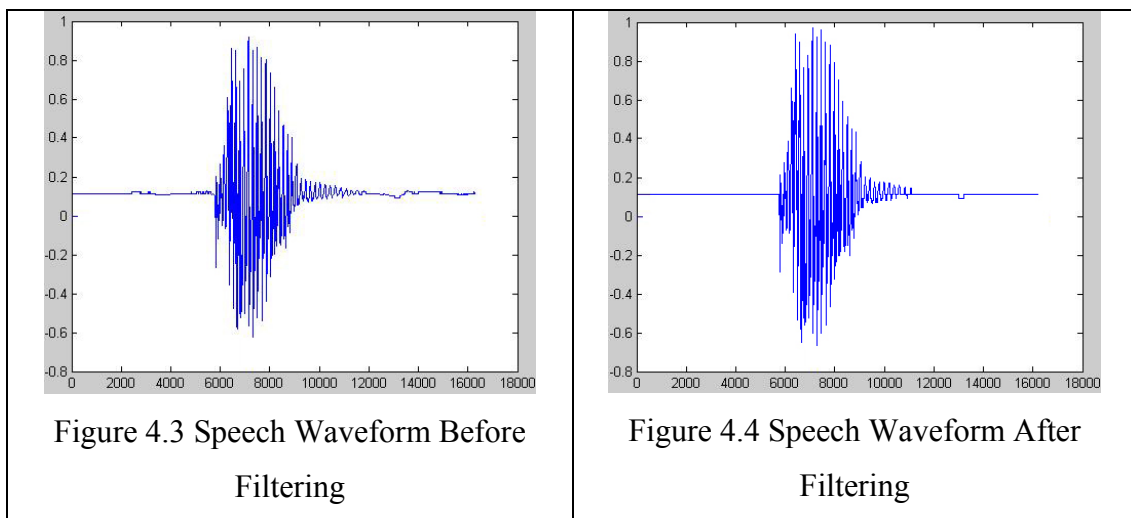


Figure 4.1 Speech Waveform Before
Centring Process

Figure 4.2 Speech Waveform After
Centring Process

The next process is to suppress the noise present in the speech waveform. Although the sound recorder program has performed the initial filtering, some noise is still present in the speech waveform. Another section of the Matlab program code is used to set a threshold value on the speech signal. Any value of the speech signal that falls below this threshold value will be set to zero. This will greatly suppress the unwanted noise and in the meantime preserve the content of the main speech signal. This is illustrated in Figure 4.3 and Figure 4.4 below. After some testing, it was found that a threshold value of 0.02 is most suitable.



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

4.3 Extraction Process

In the extraction process, the most important part is to determine the epochs of the speech waveform. Epochs can be easily detected by comparing the values of the signal. Any difference in the sign of the signal values will indicate that the waveform has crossed the zero line. In TEAPAR, 2 parameters are derived from the epoch, namely the duration (D) and shape (S) parameters. The D parameter which corresponds to the real zeros is actually the duration between successive zero crossings. Value of D can be easily determined by counting the number of signal sample values that are of the same sign in an epoch. The number of pulses in each epoch also depends on the sampling rate used. Figure 4.5 below illustrates how D parameter is calculated. Note: If the initial speech waveform is not centered at the zero line, inaccurate values of D will be produced. This is the reason for applying zero centring process described earlier.

The next step is to retrieve the S parameter. The S parameter corresponds to the shape of the epoch which can be represented by either the number of positive minimas or negative maximas. The procedure for calculating the D parameter is as follows. The value of a signal sample is subtracted from the previous signal sample value. For example, $u(169) - u(168)$. This difference value is then stored in a temporary location. This subtraction process is applied to all adjacent signal samples in the given epoch. The resulting 'difference' signal can then be analysed from the number of zero crossings. This number corresponds to the number of positive minima or negative maxima. The S value is subsequently derived.

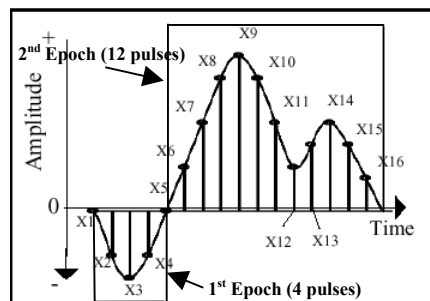


Figure 4.5 Sampling Waveform [1]

4.4 Coding Process

The D/S pairing of each epoch is used to produce TESPAP alphabet symbols. A TESPAP codebook comprises of a symbol table of 28 different symbols and is used to map the duration/shape (D/S) parameters of each epoch into a single symbol. In most applications, a TESPAP alphabet of 28 different symbols is sufficient enough to represent the original waveform. Table 3.1 illustrate a version of the TESPAP codebook. The output of the TESPAP coder is a stream of symbols shown in Figure 4.5 below.

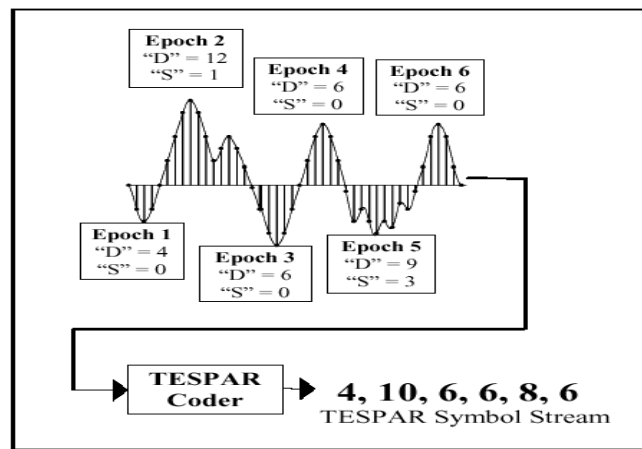


Figure 4.6 TESPAP Symbol Stream

4.5 Classification using TESPAP and NEURAL NETWORK

The TESPAP matrices are ideally matched to the processing requirements of Neural Networks for which the use of fixed sized training and interrogation vectors is typically essential. [1] Figure 4.6 below shows an overview of the classification process using a TESPAP S-Matrix in a Neural Network.

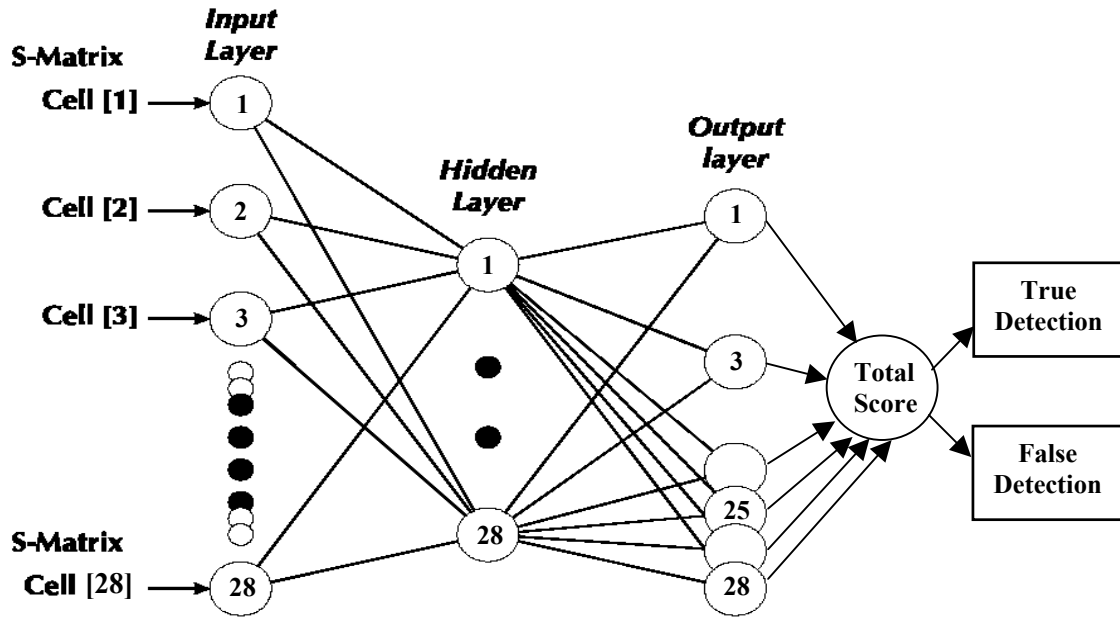


Figure 4.7 Multi-Layer Perceptron [1]

4.5.1 Database

Before the classification takes place, 10 speech samples from each speaker have to be collected, coded, converted to an S-Matrix and stored into the database. The utterance of these 10 speech samples has to be of the same phrase. These 10 speech samples for each speaker are then averaged to produce the mean reference matrix. This averaging process is necessary as it reduces the inconsistency of the speaker's speech. The 10 speech samples must also go through a process to find the standard deviation of the samples.

4.5.2 Neural Network Process

After the mean reference matrix of each individual speaker is obtained, the elements in the matrix are then passed through the Neural Network to construct the weight and bias for each individual speaker. Once the weight and bias of each individual speaker is generated, the program is ready to classify any unknown user.

4.5.3 Classification Process

Once an unknown speaker has recorded the speech into a wave file, the wave file is then loaded into the Matlab program and goes through TESPAP conversion process. A targeted S-Matrix is generated and it goes through the Neural Network for classification. Inside the Neural Network, the targeted S-Matrix is being compared against the reference S-Matrices in the database. An individual score is generated for each individual speaker in the database. The first decision applied is that the speaker with the highest score is deemed the unknown speaker. This is followed by an inspection on the highest score obtained. A second classification decision involves checking on whether the highest score exceeds a threshold. If the threshold is exceeded, then the unknown speaker is classified as the speaker corresponding to the highest score. Table 4.1 below shows an example to further explain the score concept. Note, in this thesis, a threshold of 9 was employed.

1st Scenario (threshold = 9)	2 nd Scenario (threshold = 9)
Ryan's score = 13 Dave's score = 11	Ryan's score = 8 Dave's score = 3
Alan's score = 12 John's score = 5	Alan's score = 5 John's score = 7
Unknown speaker = Ryan	Unknown speaker = Unidentified

Table 4.1 Score Concept

In the second scenario, although speaker Ryan has the highest score, the system does not recognise him as the unknown speaker because the corresponding score of Ryan does not exceed the threshold value.

Chapter 5

Discussion of Results

Product evaluation is needed before a product can be sold in the market. The evaluation results will show if the product conforms to the product working specifications. This will give customers a clearer view on the quality of the product and also provide feedback to the manufacturing department. This chapter thoroughly examines the strengths and weaknesses of the proposed speech recognition procedure. There are altogether three different testing stages each targeted at different characteristics of the program. Results and a detailed discussion will be presented for each stage of the test.

5.1 Assessment Stage 1

This assessment stage is used to test the basic function of the speech recognition program. This phase investigates the correlation between words with one, two three and four syllables. The main purpose of the phase is to show that words of the same syllable are more indistinguishable than words of different syllables. This will thoroughly test the system strength in rejecting words that are different from the chosen word.

In this phase, 5 speakers were needed to run this test. The first 4 speakers each records 10 samples of the word “Queensland” into the computer. These samples are recorded as wave files and loaded into the program. The program then converts them into TESPAS S-Matrices and then converts the average of the 10 S-Matrices of each speaker to produce a reference S-Matrix for each of the 4 speakers. The four reference S-Matrices are then transferred to the Neural Network for training to find the weights and biases. Once the training is done, the program is ready to accept words for recognition. To fully test the robustness of the system, the fifth speaker records four sets of 5 different words of the same syllable. Each recorded word is converted into a TESPAS S-Matrix which then passes through the Neural Network for classification. Using the Neural Network classification method, the score of each word is tabulated in Table 5.1. Figure 5.1, 5.2,

5.3, 5.4 shows the reference S-Matrix of the word “Queensland” uttered by Speaker A, B, C and D. Figure 5.5 – 5.12 shows the S-Matrix of words with different syllables.

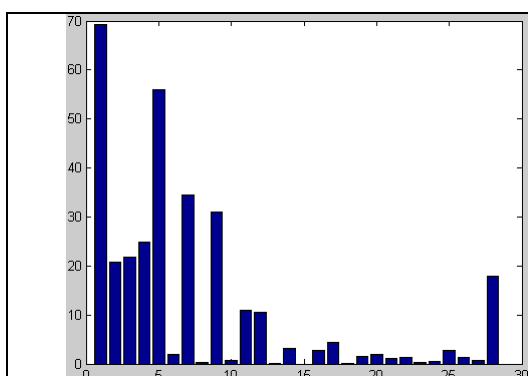


Figure 5.1 Speaker A's reference S-Matrix of the word “Queensland”

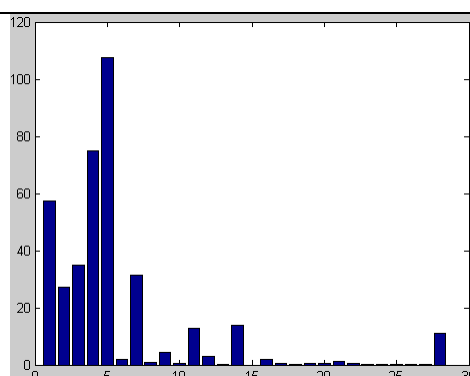


Figure 5.2 Speaker B's reference S-Matrix of the word “Queensland”

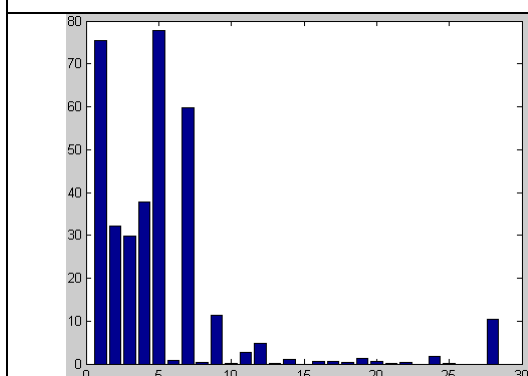


Figure 5.3 Speaker C's reference S-Matrix of the word “Queensland”

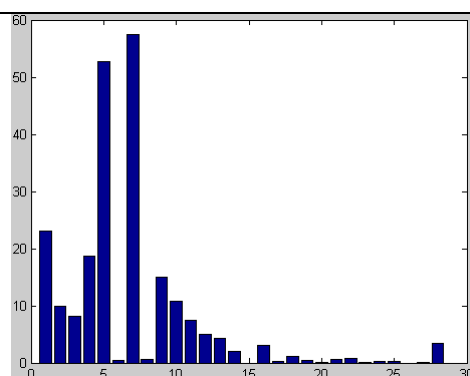


Figure 5.4 Speaker D's reference S-Matrix of the word “Queensland”

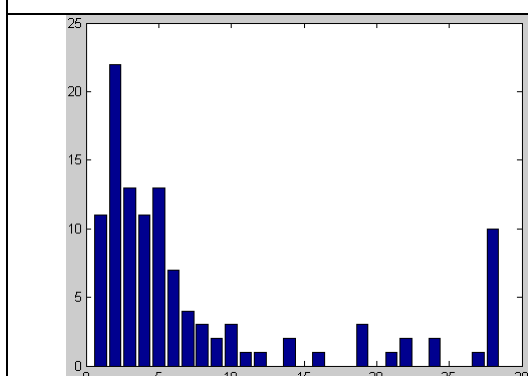


Figure 5.5 S-Matrix of the word “Clock”
(Single syllable)

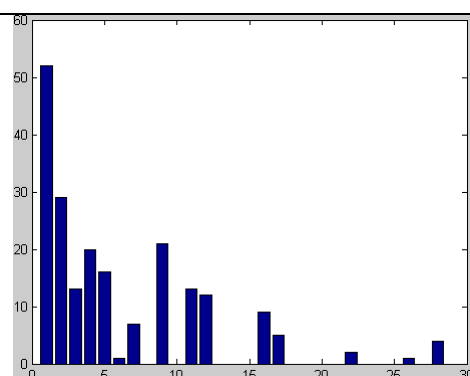


Figure 5.6 S-Matrix of the word “Nose”
(Single syllable)

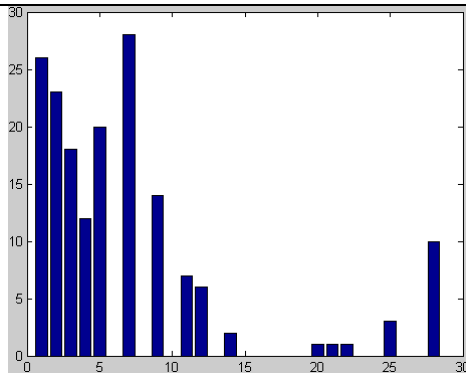


Figure 5.7 S-Matrix of the word
“Keyboard” (Two syllables)

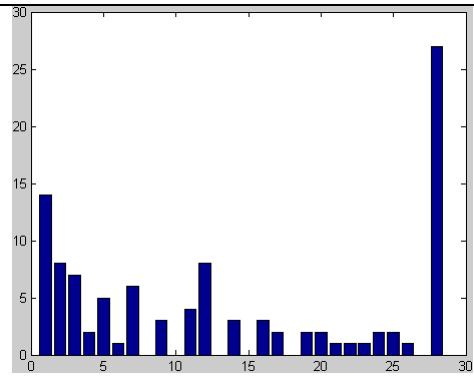


Figure 5.8 S-Matrix of the word “Speaker”
(Two syllables)

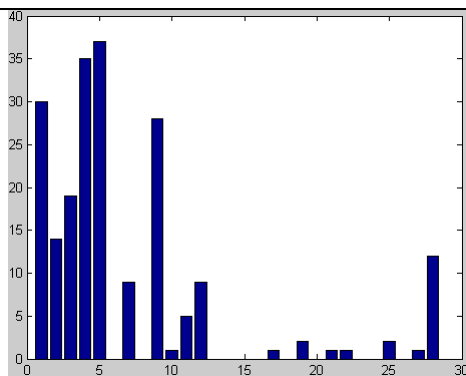


Figure 5.9 S-Matrix of the word
“Logitech” (Three syllables)

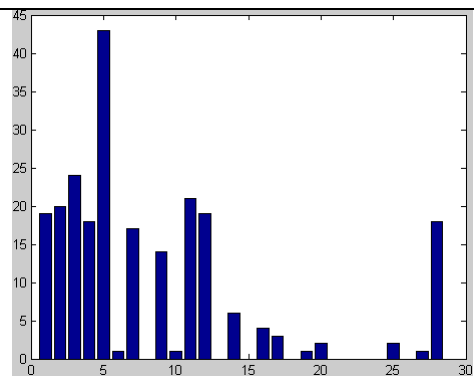


Figure 5.10 S-Matrix of the word
“Microphone” (Three syllables)

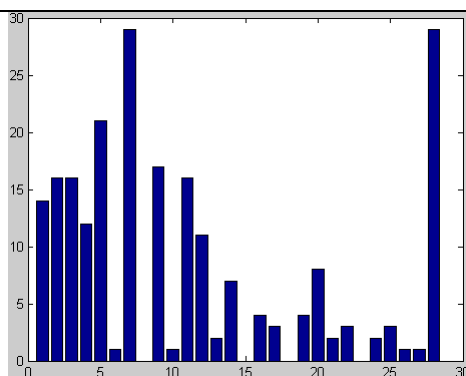


Figure 5.11 S-Matrix of the word
“Calculator” (Four syllables)

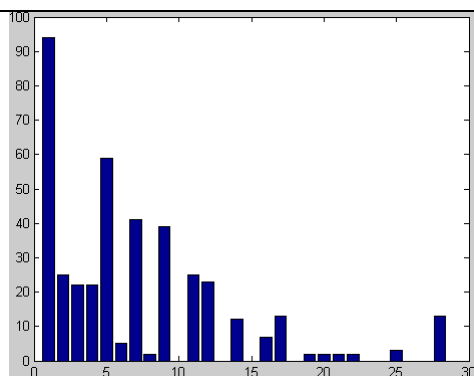


Figure 5.12 S-Matrix of the word
“Evolution” (Four syllables)

Single	Score				
Syllable	Pie	Clock	Glass	Gun	Nose
Speaker A	4	5	8	5	6
Speaker B	8	6	4	5	6
Speaker C	8	7	5	6	6
Speaker D	8	8	5	8	7
Two	Score				
Syllables	Laptop	Hand Phone	Tissue	Keyboard	Speaker
Speaker A	4	7	7	9	7
Speaker B	6	5	4	9	4
Speaker C	6	4	2	9	4
Speaker D	8	8	4	7	6
Three	Score				
Syllables	Toshiba	Logitech	Basketball	Pintara	Microphone
Speaker A	3	8	7	5	6
Speaker B	4	6	5	7	6
Speaker C	2	6	6	4	3
Speaker D	5	9	6	5	5
Four	Score				
Syllables	Roller Coaster	Sound Recorder	Calculator	Evolution	Dictionary
Speaker A	3	1	4	2	2
Speaker B	2	3	2	4	7
Speaker C	3	2	1	4	7
Speaker D	4	2	6	3	5

Table 5.1 Scoring Table for Assessment Stage 1

5.1.1 Discussion for Assessment stage 1

In single syllable words test, there is no “True” detection for any of the words recorded. However for the recorded words of single syllable, the collected score of some words are quite high. For example, the word “Pie” scored 8 points for speaker B, C and D. This means that the word “Pie” uttered by the unknown speaker is somehow similar to the word “Queensland” uttered by speaker B, C and D.

In the double syllables words test, there is 1 “False” acceptance for the words recorded. The word “Keyboard” uttered by the unknown speaker obtained a score of 9 for speaker A, B and C. This score falls above the threshold level so the system categorized it as a “True” detection. The figures above shows that the S-Matrix of the word “Keyboard” bore a close resemblance to the reference S-Matrices of speaker A, B and C.

In the three syllables words test, 1 “False” acceptance is detected. The word “Logitech” uttered by the unknown speaker obtained a score of 9 for speaker D. This score falls above the threshold level so the system categorized it as “True” detection. Figure 5.9 show that the S-Matrix of the word “Logitech” bore a close resemblance to the reference S-Matrix of speaker D. One of the reasons behind this resemblance is due to the length of the word “Logitech” is almost the same with the word “Queensland”. This reduces the system’s ability to discriminate these words. Another reason is due to the inconsistency of the recording of the 10 samples by speaker D.

In the four syllables words test, no “True” detection is detected. Furthermore from the results, we can see that all the scores obtained are very much below the threshold level.

5.1.2 Conclusion on Assessment Stage 1

The whole test in stage 1 yield a 90% success rate as only two “False” acceptances were detected. Although this shows that the system is not foolproof, it can still be said that the system performs its basic task. From the result shown in the Table 5.1, we can see that the “False” acceptance came from two and three syllables words (“Keyboard” and “Logitech”). The test results also shows that four syllables words performed the best as almost all the scores collected are well below the threshold level. The reason behind this is because the reference word stored in the database is a double syllables word “Queensland”. Therefore in this test we can conclude the closer the number of syllable word to the chosen word, the lower the discriminating power of the system.

5.2 Assessment Stage 2

The main purpose of this phase is to examine the discriminating power of words of different syllables. This will show which number of syllable words performed the best under the same condition.

In this phase, only 1 speaker is needed to run this test. Speaker A is to record 10 samples of each word “Clock”, “Speaker”, “Microphone” and “Evolution” into the computer. The program then converts them into respective reference TESPAS S-Matrices. The reference S-Matrix of each word is then transferred to the Neural Network for training. Once the training is done, the program is ready for speech recognition. In the speech recognition process, Speaker A records 2 samples of the word “Clock”, “Speaker”, “Microphone” and “Dictionary”. These samples are then processed into the TESPAS S-Matrix and compared against the individual mean S-Matrix in the database. For example, the 2 samples of the word “Clock” is compared against the mean S-Matrix of the word “Clock”. Using the Neural Network classification method, the score of each word is tabulated in Table 5.2.

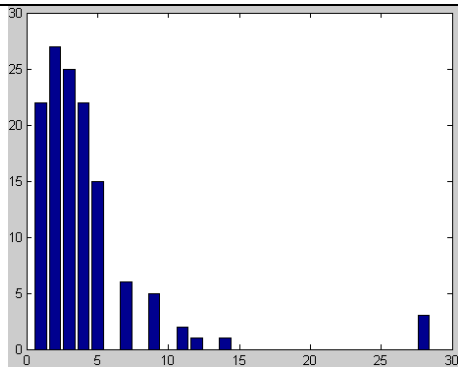


Figure 5.13 1st sample word “Clock”

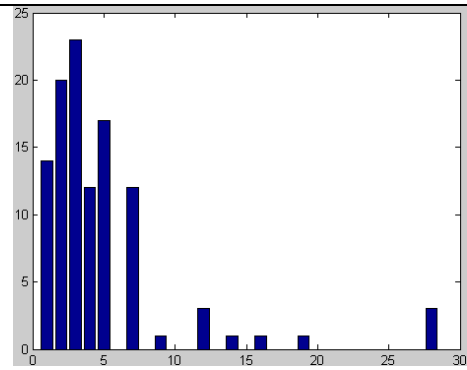


Figure 5.14 2nd sample word “Clock”

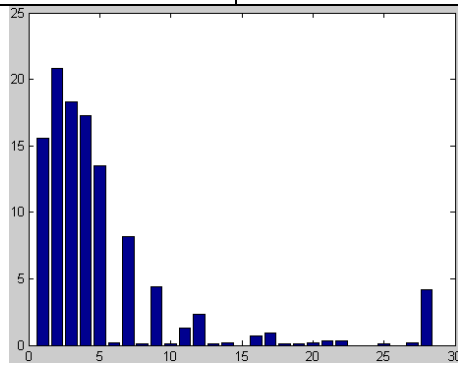


Figure 5.15 Reference S-Matrix of word “Clock”

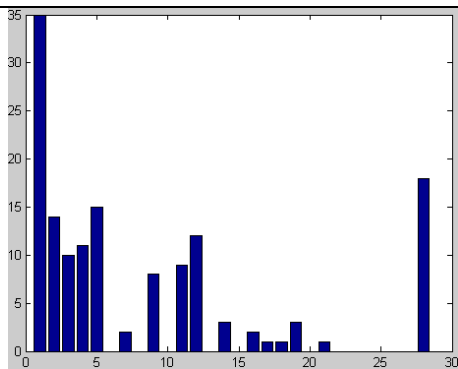


Figure 5.16 1st sample word “Speaker”

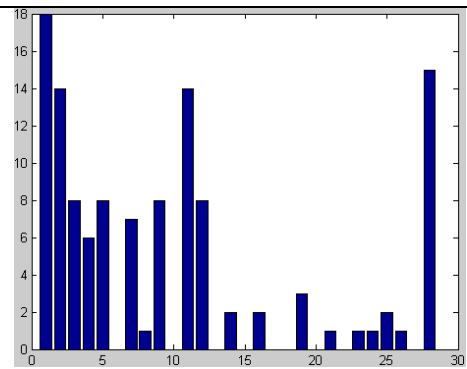


Figure 5.17 2nd sample word “Speaker”

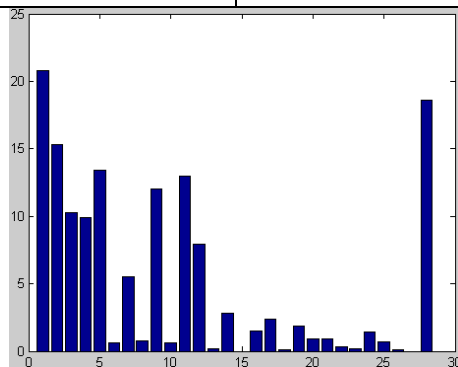


Figure 5.18 Reference S-Matrix of the word “Speaker”

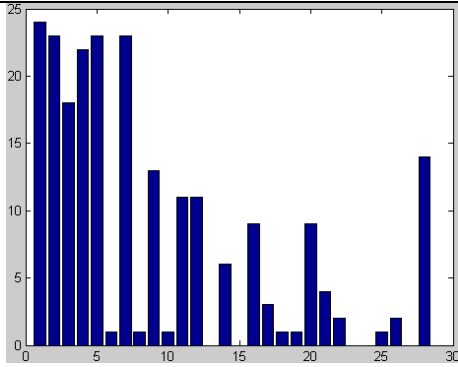


Figure 5.19 1st sample word “Microphone”

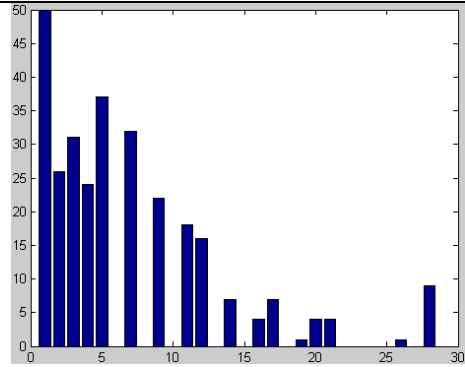


Figure 5.20 2nd sample word “Microphone”

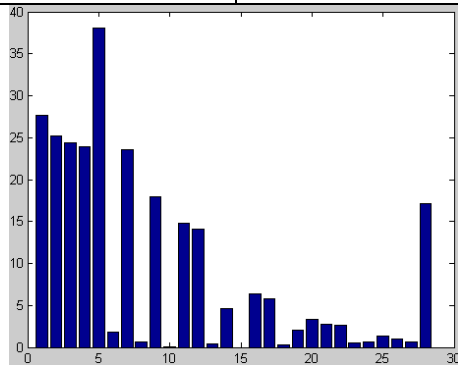


Figure 5.21 Reference S-Matrix of word “Microphone”

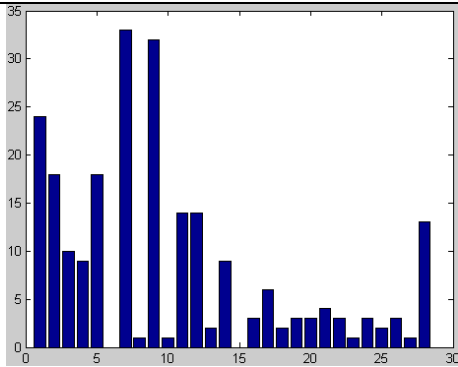


Figure 5.22 1st sample word “Evolution”

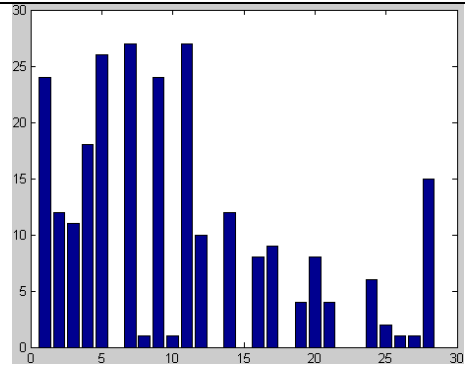


Figure 5.23 2nd sample word “Evolution”

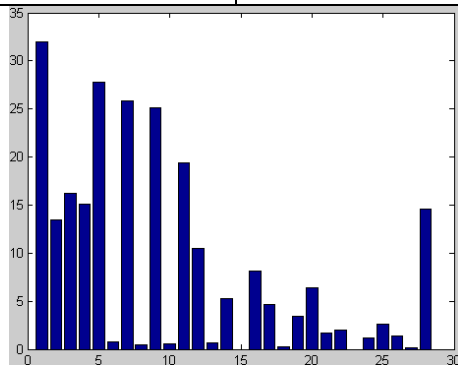


Figure 5.24 Reference S-Matrix of the word “Evolution”

Clock	Single Syllable	Score
	1 st Sample	13
	2 nd Sample	12
Speaker	Two Syllables	Score
	1 st Sample	11
	2 nd Sample	11
Microphone	Three Syllables	Score
	1 st Sample	11
	2 nd Sample	10
Evolution	Four Syllables	Score
	1 st Sample	10
	2 nd Sample	9

Table 5.2 Scoring Table for Assessment Stage 2

5.2.1 Discussion for Assessment Stage 2

From table 5.2 above, we can see that this stage recorded a 100% true detection. For the single syllable test, the collected scores for both the samples are way above the threshold level. This shows that the discriminating power of the system on single syllable words is very high. From Figure 5.13 and 5.15, we can see that the shape of the S-Matrix of the 1st sample is almost identical to that of the reference S-Matrix.

In the double syllables words test, the collected scores are well above the threshold level. From Figure 5.16 to 5.18, we can see that the shape of the samples' S-Matrices is not really identical to that of the reference matrix. However, upon closer examination, we can see that some features of the samples S-Matrices do indeed resemble the reference S-Matrix.

In the three syllables words test, the collected scores are also well above the threshold level. From table 5.2, it is noted that the score in three syllables test is not as ideal as the score in single syllable test. From Figure 5.19 to 5.21, we can see that only part of the samples' S-Matrices resemble the reference S-Matrix.

In the four syllables test, the collected scores fall just above the threshold level. Although the system is able to correctly classify the speech, the collected scores are also

not as ideal as the scores in single syllable test. Upon closer inspection, we can still see some resemblance of the samples' S-Matrices to the reference S-Matrix.

5.2.2 Conclusion on Assessment Stage 2

From the results shown in Table 5.2, we can conclude that the system has higher discriminating power on shorter syllables word. One of the reasons behind this is the consistency in the recording of the 10 samples. Single syllable words are easier to pronounce than longer syllables words, therefore the level of consistency is higher. Certain words like "Microphone" are harder to pronounce consistently. Overall the system has proved to be very reliable on all the different syllables words used.

5.3 Assessment Stage 3

This is the final stage of the assessment phase. The main purpose of this phase is to examine the discriminating power of the system against 10 speakers. This is a very important assessment stage as this test will show the vulnerability of the system. With the 10 speakers uttering the same word three times, the system has got to be competent in discriminating them.

In this phase, 11 speakers are needed to run this test. Speaker A is to record 10 samples of the word "Clock" into the system. The program then converts them into a reference TESPAS S-Matrices shown in Figure 5.25. The reference S-Matrix is then transferred to the Neural Network for training. Once the training is completed, the program is ready for speech recognition. In the speech recognition process, the other 10 speakers each record 3 samples of the word "Clock". These 30 samples are then each processed into an individual TESPAS S-Matrix and compared against the reference S-Matrix of Speaker A in the database. Using the Neural Network classification method, the score of each word is tabulated in Table 5.3. Figure 5.25 shows the reference S-Matrix of Speaker A. Figure 5.26 to 5.35 shows the S-Matrices of the 10 Speakers. Due to space constraint, only 1 sample from each of the 10 speakers is displayed.

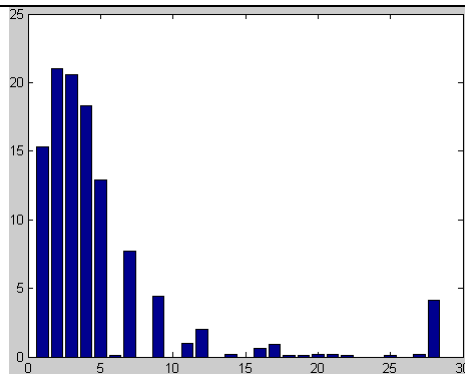


Figure 5.25 Reference S-Matrix of the word "Clock"

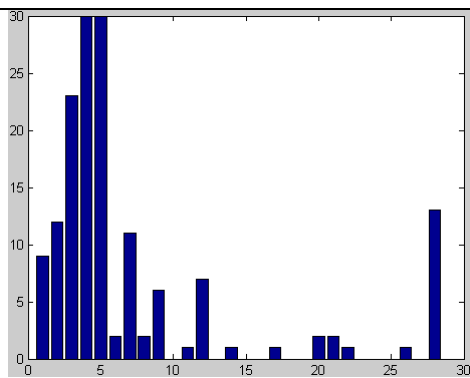


Figure 5.26 1st Speaker
(3rd sample of the word "Clock")

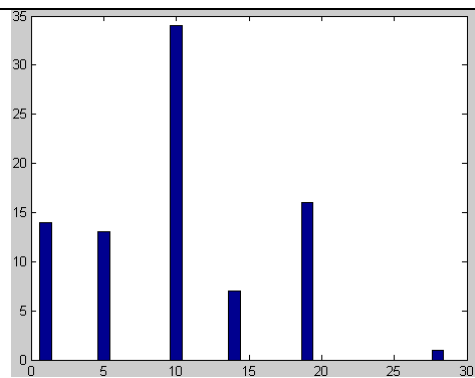


Figure 5.27 2nd Speaker
(2nd sample of the word "Clock")

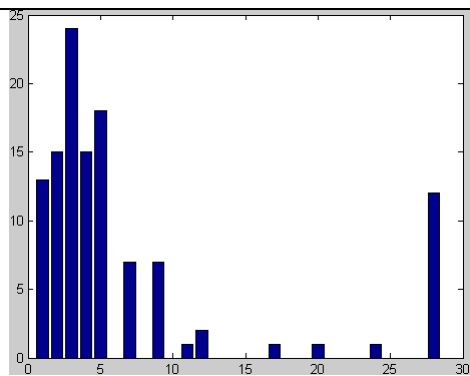


Figure 5.28 3rd Speaker
(3rd sample of the word "Clock")

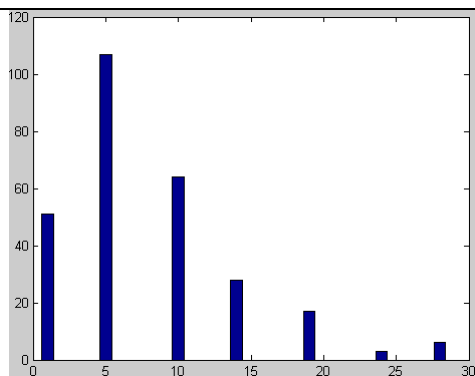


Figure 5.29 4th Speaker
(1st sample of the word "Clock")

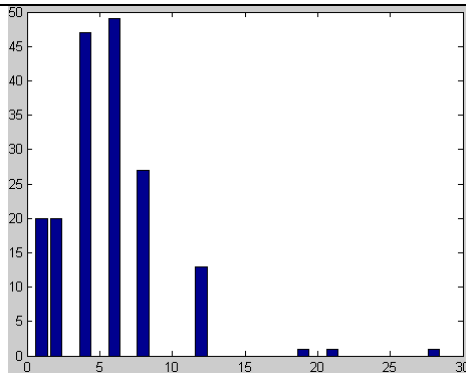


Figure 5.30 5th Speaker
(2nd sample of the word “Clock”)

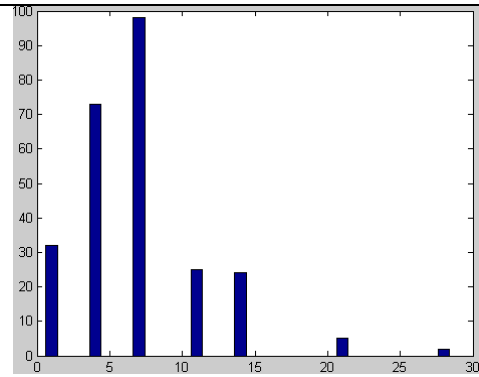


Figure 5.31 6th Speaker
(1st sample of the word “Clock”)

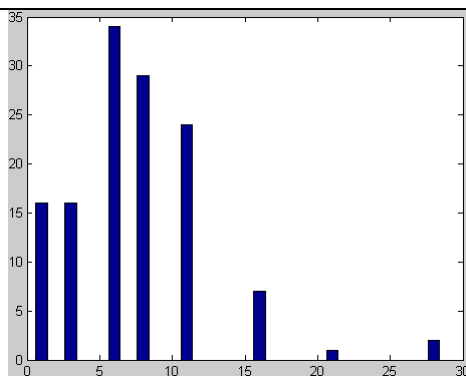


Figure 5.32 7th Speaker
(1st sample of the word “Clock”)

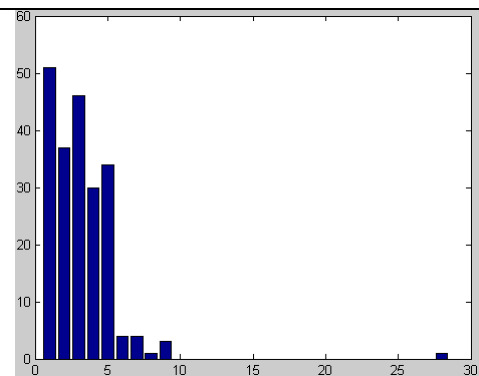


Figure 5.33 8th Speaker
(2nd sample of the word “Clock”)

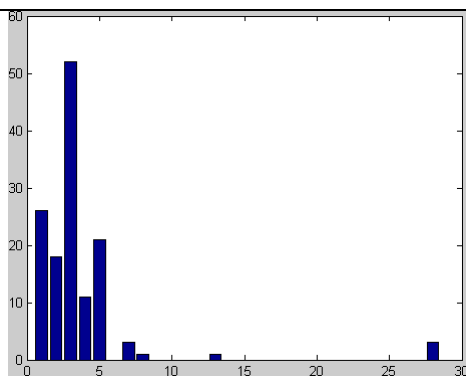


Figure 5.34 9th Speaker
(1st sample of the word “Clock”)

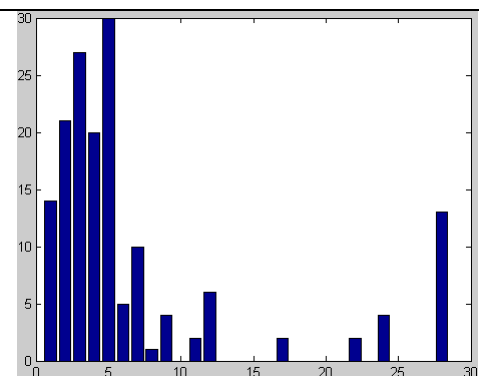


Figure 5.35 10th Speaker
(1st sample of the word “Clock”)

	Score		
	Sample	Sample	Sample
1 st Speaker	8	3	5
2 nd Speaker	7	6	6
3 rd Speaker	5	9	10
4 th Speaker	5	8	8
5 th Speaker	5	6	5
6 th Speaker	6	5	6
7 th Speaker	6	7	7
8 th Speaker	8	8	6
9 th Speaker	6	7	5
10 th Speaker	8	5	4

Table 5.3 Scoring Table for Assessment Stage 3

5.3.1 Discussion for Assessment Stage 3

From table 5.3 above, we can see that this stage recorded 93% accuracy with only two “False” acceptances. This shows that discriminating power of the system on single syllable words is very high. Figure 5.28 which represent the 3rd sample of the 3rd speaker, we can see that the shape of the S-Matrix is very similar to that of the reference S-Matrix of speaker A. Thus in the classification stage, the Neural Network is unable to differentiate it from the reference S-Matrix. The Neural Network generated a score of 10 for this sample which falls above the threshold level and hence the system incorrectly classified it as a “True” detection even though 3rd speaker and speaker A are two different speakers. The rest of the S-Matrices collected are very different from the reference S-Matrix and hence they are all classified as false detection. Figure 5.27 and 5.29 shows the S-Matrices of the 2nd speaker and the 4th speaker. Both S-Matrices are very different from the reference S-Matrix thus both generate very low scores.

5.3.2 Conclusion on Assessment Stage 3

From the results shown in Table 5.3, we can conclude that the system has very strong speaker discriminating power on single syllable words. The above results when combine with assessment stage 2 results shows that not only can the system often correctly identify the speaker, it can also identify the correct word. These positive results generated confirm the usefulness of TESPAP and Neural Networks for speech recognition.

Chapter 6

Conclusion

With the positive results collected, Speech recognition using TESPAP and Neural Network has proven to be excellent in classifying speech signals. Unlike traditional speech recognition techniques which involve complex Fourier transformations, the method used by TESPAP in coding the signal is simple and accurate. The conversion from the D and S parameters to the S-Matrix is quite straight forward. The acoustics characteristics of the speaker's speech can easily be detected from visual inspection on the S-Matrix. The Neural Network classification method used is also reliable and uncomplicated to implement.

From the results of the three assessment stages, it is obvious that single syllable words are more reliable in terms of training. This is probably because humans' pronunciations of single syllable words are more consistent.

Despite the positive results collected in Chapter 5, there are still a few "False" acceptances and "False" rejections being detected. This may be considered a serious issue when it is applied in a high security room. The main reason behind these errors is due to the inconsistency in the human speech. Although TESPAP and Neural Network is a formidable combination, the single layer of Perceptron technique is unable to reduce the inconsistency of the speech signals. Therefore more robust and powerful methods have to be employed to reduce the inconsistency of the speech signals. This will be further explained in the following chapter.

Finally to conclude, TESPAP and Neural Network processing has the ability to discriminate signals that remain indistinguishable in the frequency domain. Furthermore due to their economic, robustness and flexibility, these two combined techniques can be easily implemented on cost effective machines which requires speech verification or identification.

Chapter 7

Future Work and Comments

Speech Recognition using TESPAP and Neural Network has been employed successfully in this thesis. Although the success rate for the recognition of single syllable words is very high, there are still improvements needed for on longer syllable words. There are many other coding and classification methods that can be exploited to improve the performance of the recognition process. The following section examines other possible methods of coding to improve the discriminating power of the system.

7.1 Multiple Multi-Layer Perceptron

The accuracy of the system can be improved by combining several Multi-Layer Perceptrons together in the final classification stage. By varying the sample speech signals used during training, each Multi-Layer Perceptron in the network will be trained under different conditions and they will base their classification on the distinctive properties of the training materials.

At the classification stage, each of these Multi-Layer Perceptrons is examined separately to produce individual decisions. A final decision then is obtained by averaging out the individual decisions. Another method of determining the final decision is by giving a greater score to more reliable and significant networks and lower scores to the unreliable networks [1].

7.2 Other Methods in Neural Networks

Many methods of classifying speech signals can be found in Neural Network. One of the methods is based on the Autoassociative Neural Network model. The distribution capturing ability of the network is exploited to build the speaker speech signal [6]. Another high performance Neural Network based approach is by using a State Transition Matrix. This method has the ability to address inconsistent speech signals. An unsupervised Learning method like Kohonen Self-Organising Map can also be employed.

7.3 Speaker Identification

The current system is more focused on speaker verification which tests an unknown speaker against a known speaker. The method presented in this thesis is still not reliable enough to be used on speaker identification applications. In speaker identification, the aim is to determine whether the utterance of an unknown speaker belongs to any of the speakers from amongst a known group. Of these two applications, speaker identification is generally more difficult to achieve due the larger speaker populations which will produce more errors. Future work should be concentrated on speaker identification as it will increase the commercial value of the system.

7.4 Comments

The usefulness of TESPAP and Neural Network for speech recognition has been proven in this thesis. The results obtained in chapter 5 have further substantiated my claims that I have developed a strong understanding of the concept behind TESPAP and Neural Networks. Generally the results obtained are of high accuracy. One of the reasons why some incorrect results were obtained is due to the strength of the Multi-Layer Perceptron technique I chose for the classification stage. Another reason is due to the limitations of Neural Network itself. Neural Networks are generally more powerful for static pattern recognition. As indicated in chapter 1, human speech is very dynamic and stochastic in nature. Every single utterance produces a different waveform and this ultimately affects the processing of the speech signals.

Compared with a state of the art technique such as Hidden Markov Model, the application of Neural Networks to speech recognition tasks has shown to be limited due to its inability to cope with the highly non-stationary dynamic patterns of speech [6].

References

- [1] R.A. King and T.C Phipps, “Shannon, TESPAP and Approximation Strategies”, ICSPAT 98, Vol. 18, pp 445-453, Great Britain 1999
- [2] J.C.R.Licklider, I. Pollack, “Effects of Differentiation, Integration and Infinite Peak Clipping upon the Intelligibility of Speech”, journal of the Acoustical society of America, Vol. 20, no. 1, pp42-51, Jan 1948.
- [3] E. C. Titchmarsh, “The Zeros of Certain Integral Functions”, Proc. Progress. Math.Soc., Vol. 25, pp. 283-302, May 1926
- [4] R.A. King and W. Gosling, Electronic Letters, Vol. 14, pp.456-457, 1978
- [5] M. T. Hagan and H. B. Demuth and M. Beale, “Neural Network Design” International Thomson Publishing, 1995
- [6] S. Yang, M.J. Er, and Y. Gao, “A High Performance Neural-Network-Based Speech Recognition System”, Proceeding of International Joint Conference on Neural Networks, Vol 2, 2001, pp1527.
- [7] L. R. Rabiner and B.-H. Juang, Fundamentals of speech recognition. Englewood Cliffs, N.J.: PTR Prentice Hall, 1993.
- [8] J.-C. Junqua and J.-P. Haton, Robustness in automatic speech recognition : fundamentals and applications. Boston: Kluwer Academic Publishers, 1996.
- [9] L. W. Couch, Digital and analog communication systems, 6th ed. Upper Saddle River, N.J.: Prentice Hall, 2001.
- [10] S. Saito and K. Nakata, Fundamentals of speech signal processing. New York: Academic Press, 1985.
- [11] D. P. Morgan and C. L. Scofield, Neural networks and speech processing. Boston: Kluwer Academic Publishers, 1991.

APPENDIX A

Program Listings

Main Program (gui_combine.m)

```
function varargout = gui_combine(varargin)
% GUI_COMBINE Application M-file for gui_combine.fig
% FIG = GUI_COMBINE launch gui_combine GUI.
% GUI_COMBINE('callback_name', ...) invoke the named callback.

% Last Modified by GUIDE v2.0 18-May-2003 16:37:08

if nargin == 0 % LAUNCH GUI

    fig = openfig('gui_combine.fig','reuse');

    % Use system color scheme for figure:
    set(fig,'Color',get(0,'defaultUicontrolBackgroundColor'));

    % Generate a structure of handles to pass to callbacks, and store it.
    handles = guihandles(fig);
    guidata(fig, handles);

    if nargout > 0
        varargout{1} = fig;
    end

elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK

    try
        if (nargout)
            [varargout{1:nargout}] = feval(varargin{:}); % FEVAL
        end
    catch
        disp(lasterr);
    end

end

function varargout = Open_Callback(h, eventdata, handles, varargin)
global test
test=['c01.wav';'c02.wav';'c03.wav';'c04.wav';'c05.wav';'c06.wav';'c07.wav';'c08.wav';'c0
9.wav';'c10.wav';'u10.wav'];
set(handles.target_name,'String',sprintf('Files Loaded'))
```

```

if nargout > 0
    varargout{1} = test;
end;
% -----
function varargout = Load_Callback(h, eventdata, handles, varargin)
tic
set(handles.target_name,'String',sprintf('Searching....'))
global test
br = waitbar(0,'Please wait...');
ratio=1.5
for c=1:11
    waitbar(c/11)
    u0=wavread(test(c,:));
    mu0=mean(u0);
    ud=u0-mu0;
    a=length(ud);
    global u
    u=ud(1:4:a);
    f=length(u);
    for w=1:f
        if abs(u(w))<0.02
            u(w)=0;
        else u(w)=u(w);
        end;
    end;
    z=1;
    while abs(u(z))<0.02
        z=z+1;
    end;
    t=f;
    while abs(u(t))<0.02
        t=t-1;
    end;
    sold=z;
    j=0;
    u(w+1)=0;          %%%% this is to force the last value of "u" to be zero
    yyy=1;
    xxx=0;
    www=1;
    zzz=0;
    for s=2+z:t+1
        if sign(u(s))~=sign(u(s-1))
            if u(s-1)~=0
                j=j+1;
                D(j)=s-sold;
                m=0;
                if s-1<sold+2
                    l=s-1:sold+2;
                else

```

```

l=sold+2:s-1;
end;
for i=l
    if u(i)~=u(i-1)
        y(i)=u(i)-u(i-1);
        if yyy==1 %1
            if sign(u(i))==1 %% Positive EPOCH
                if sign(y(i))==-1
                    xxx=1;
                    m=m+1;
                    yyy=0;
                end;
            end;
        end;
        if xxx==1 %2
            if sign(u(i))==1 %% +ve EPOCH
                if sign(y(i))==1
                    yyy=1;
                    xxx=0;
                end;
            end;
        end;
        if www==1 %3
            if sign(u(i))==-1 %% -neg EPOCH
                if sign(y(i))==1
                    zzz=1;
                    m=m+1;
                    www=0;
                end;
            end;
        end;
        if zzz==1 %4
            if sign(y(i))==-1 %% -neg EPOCH
                if sign(y(i))==-1
                    www=1;
                    zzz=0;
                end;
            end;
        end;
    end; %if
    S(j)=m;
    if S(j)<0
        S(j)=0;
    elseif S(j)>5
        S(j)=5;
    end;
end; %for
end;
sold=s-1;

```

```

end;
end;
for k=1:j
if D(k)==1 & S(k)==0
A(k)=1;
elseif D(k)==2 & S(k)==0
A(k)=2;
elseif D(k)==2 & S(k)==1
A(k)=2;
elseif D(k)==3 & S(k)==0
A(k)=3;
elseif D(k)==3 & S(k)==1
A(k)=3;
elseif D(k)==3 & S(k)==2
A(k)=3;
elseif D(k)==4 & S(k)==0
A(k)=4;
elseif D(k)==4 & S(k)==1
A(k)=4;
elseif D(k)==4 & S(k)==2
A(k)=4;
elseif D(k)==4 & S(k)==3
A(k)=4;
elseif D(k)==5 & S(k)==0
A(k)=5;
elseif D(k)==5 & S(k)==1
A(k)=5;
elseif D(k)==5 & S(k)==2
A(k)=5;
elseif D(k)==5 & S(k)==3
A(k)=5;
elseif D(k)==6 & S(k)==0
A(k)=6;
elseif D(k)==6 & S(k)==1
A(k)=6;
elseif D(k)==6 & S(k)==2
A(k)=6;
elseif D(k)==6 & S(k)==3
A(k)=6;
elseif D(k)==6 & S(k)==4
A(k)=6;
elseif D(k)==7 & S(k)==0
A(k)=6;
elseif D(k)==7 & S(k)==1
A(k)=6;
elseif D(k)==7 & S(k)==2
A(k)=6;
elseif D(k)==7 & S(k)==3
A(k)=6;

```



```

elseif D(k)==7 & S(k)==4
    A(k)=6;
elseif D(k)==8 & S(k)==0
    A(k)=7;
elseif D(k)==8 & S(k)==1
    A(k)=8;
elseif D(k)==8 & S(k)==2
    A(k)=8;
elseif D(k)==8 & S(k)==3
    A(k)=8;
elseif D(k)==8 & S(k)==4
    A(k)=8;
elseif D(k)==8 & S(k)==5
    A(k)=8;
elseif D(k)==9 & S(k)==0
    A(k)=7;
elseif D(k)==9 & S(k)==1
    A(k)=8;
elseif D(k)==9 & S(k)==2
    A(k)=8;
elseif D(k)==9 & S(k)==3
    A(k)=8;
elseif D(k)==9 & S(k)==4
    A(k)=8;
elseif D(k)==9 & S(k)==5
    A(k)=8;
elseif D(k)==10 & S(k)==0
    A(k)=7;
elseif D(k)==10 & S(k)==1
    A(k)=8;
elseif D(k)==10 & S(k)==2
    A(k)=8;
elseif D(k)==10 & S(k)==3
    A(k)=8;
elseif D(k)==10 & S(k)==4
    A(k)=8;
elseif D(k)==10 & S(k)==5
    A(k)=8;
elseif D(k)==11 & S(k)==0
    A(k)=9;
elseif D(k)==11 & S(k)==1
    A(k)=10;
elseif D(k)==11 & S(k)==2
    A(k)=10;
elseif D(k)==11 & S(k)==3
    A(k)=10;
elseif D(k)==11 & S(k)==4
    A(k)=10;
elseif D(k)==11 & S(k)==5

```

```

    A(k)=10;
elseif D(k)==12 & S(k)==0
    A(k)=9;
elseif D(k)==12 & S(k)==1
    A(k)=10;
elseif D(k)==12 & S(k)==2
    A(k)=10;
elseif D(k)==12 & S(k)==3
    A(k)=10;
elseif D(k)==12 & S(k)==4
    A(k)=10;
elseif D(k)==12 & S(k)==5
    A(k)=10;
elseif D(k)==13 & S(k)==0
    A(k)=9;
elseif D(k)==13 & S(k)==1
    A(k)=10;
elseif D(k)==13 & S(k)==2
    A(k)=10;
elseif D(k)==13 & S(k)==3
    A(k)=10;
elseif D(k)==13 & S(k)==4
    A(k)=10;
elseif D(k)==13 & S(k)==5
    A(k)=10;
elseif D(k)==14 & S(k)==0
    A(k)=11;
elseif D(k)==14 & S(k)==1
    A(k)=12;
elseif D(k)==14 & S(k)==2
    A(k)=13;
elseif D(k)==14 & S(k)==3
    A(k)=13;
elseif D(k)==14 & S(k)==4
    A(k)=13;
elseif D(k)==14 & S(k)==5
    A(k)=13;
elseif D(k)==15 & S(k)==0
    A(k)=11;
elseif D(k)==15 & S(k)==1
    A(k)=12;
elseif D(k)==15 & S(k)==2
    A(k)=13;
elseif D(k)==15 & S(k)==3
    A(k)=13;
elseif D(k)==15 & S(k)==4
    A(k)=13;
elseif D(k)==15 & S(k)==5
    A(k)=13;

```

```

elseif D(k)==16 & S(k)==0
    A(k)=11;
elseif D(k)==16 & S(k)==1
    A(k)=12;
elseif D(k)==16 & S(k)==2
    A(k)=13;
elseif D(k)==16 & S(k)==3
    A(k)=13;
elseif D(k)==16 & S(k)==4
    A(k)=13;
elseif D(k)==16 & S(k)==5
    A(k)=13;
elseif D(k)==17 & S(k)==0
    A(k)=11;
elseif D(k)==17 & S(k)==1
    A(k)=12;
elseif D(k)==17 & S(k)==2
    A(k)=13;
elseif D(k)==17 & S(k)==3
    A(k)=13;
elseif D(k)==17 & S(k)==4
    A(k)=13;
elseif D(k)==17 & S(k)==5
    A(k)=13;
elseif D(k)==18 & S(k)==0
    A(k)=11;
elseif D(k)==18 & S(k)==1
    A(k)=12;
elseif D(k)==18 & S(k)==2
    A(k)=13;
elseif D(k)==18 & S(k)==3
    A(k)=13;
elseif D(k)==18 & S(k)==4
    A(k)=13;
elseif D(k)==18 & S(k)==5
    A(k)=13;
elseif D(k)==19 & S(k)==0
    A(k)=14;
elseif D(k)==19 & S(k)==1
    A(k)=15;
elseif D(k)==19 & S(k)==2
    A(k)=16;
elseif D(k)==19 & S(k)==3
    A(k)=17;
elseif D(k)==19 & S(k)==4
    A(k)=17;
elseif D(k)==19 & S(k)==5
    A(k)=17;
elseif D(k)==20 & S(k)==0

```

```

    A(k)=14;
elseif D(k)==20 & S(k)==1
    A(k)=15;
elseif D(k)==20 & S(k)==2
    A(k)=16;
elseif D(k)==20 & S(k)==3
    A(k)=17;
elseif D(k)==20 & S(k)==4
    A(k)=17;
elseif D(k)==20 & S(k)==5
    A(k)=17;
elseif D(k)==21 & S(k)==0
    A(k)=14;
elseif D(k)==21 & S(k)==1
    A(k)=15;
elseif D(k)==21 & S(k)==2
    A(k)=16;
elseif D(k)==21 & S(k)==3
    A(k)=17;
elseif D(k)==21 & S(k)==4
    A(k)=17;
elseif D(k)==21 & S(k)==5
    A(k)=17;
elseif D(k)==22 & S(k)==0
    A(k)=14;
elseif D(k)==22 & S(k)==1
    A(k)=15;
elseif D(k)==22 & S(k)==2
    A(k)=16;
elseif D(k)==22 & S(k)==3
    A(k)=17;
elseif D(k)==22 & S(k)==4
    A(k)=17;
elseif D(k)==22 & S(k)==5
    A(k)=17;
elseif D(k)==23 & S(k)==0
    A(k)=14;
elseif D(k)==23 & S(k)==1
    A(k)=15;
elseif D(k)==23 & S(k)==2
    A(k)=16;
elseif D(k)==23 & S(k)==3
    A(k)=17;
elseif D(k)==23 & S(k)==4
    A(k)=17;
elseif D(k)==23 & S(k)==5
    A(k)=17;
elseif D(k)==24 & S(k)==0
    A(k)=18;

```

```

elseif D(k)==24 & S(k)==1
    A(k)=19;
elseif D(k)==24 & S(k)==2
    A(k)=20;
elseif D(k)==24 & S(k)==3
    A(k)=21;
elseif D(k)==24 & S(k)==4
    A(k)=22;
elseif D(k)==24 & S(k)==5
    A(k)=22;
elseif D(k)==25 & S(k)==0
    A(k)=18;
elseif D(k)==25 & S(k)==1
    A(k)=19;
elseif D(k)==25 & S(k)==2
    A(k)=20;
elseif D(k)==25 & S(k)==3
    A(k)=21;
elseif D(k)==25 & S(k)==4
    A(k)=22;
elseif D(k)==25 & S(k)==5
    A(k)=22;
elseif D(k)==26 & S(k)==0
    A(k)=18;
elseif D(k)==26 & S(k)==1
    A(k)=19;
elseif D(k)==26 & S(k)==2
    A(k)=20;
elseif D(k)==26 & S(k)==3
    A(k)=21;
elseif D(k)==26 & S(k)==4
    A(k)=22;
elseif D(k)==26 & S(k)==5
    A(k)=22;
elseif D(k)==27 & S(k)==0
    A(k)=18;
elseif D(k)==27 & S(k)==1
    A(k)=19;
elseif D(k)==27 & S(k)==2
    A(k)=20;
elseif D(k)==27 & S(k)==3
    A(k)=21;
elseif D(k)==27 & S(k)==4
    A(k)=22;
elseif D(k)==27 & S(k)==5
    A(k)=22;
elseif D(k)==28 & S(k)==0
    A(k)=18;
elseif D(k)==28 & S(k)==1

```

```

    A(k)=19;
elseif D(k)==28 & S(k)==2
    A(k)=20;
elseif D(k)==28 & S(k)==3
    A(k)=21;
elseif D(k)==28 & S(k)==4
    A(k)=22;
elseif D(k)==28 & S(k)==5
    A(k)=23;
elseif D(k)==29 & S(k)==0
    A(k)=18;
elseif D(k)==29 & S(k)==1
    A(k)=19;
elseif D(k)==29 & S(k)==2
    A(k)=20;
elseif D(k)==29 & S(k)==3
    A(k)=21;
elseif D(k)==29 & S(k)==4
    A(k)=22;
elseif D(k)==29 & S(k)==5
    A(k)=22;
elseif D(k)==30 & S(k)==0
    A(k)=18;
elseif D(k)==30 & S(k)==1
    A(k)=19;
elseif D(k)==30 & S(k)==2
    A(k)=20;
elseif D(k)==30 & S(k)==3
    A(k)=21;
elseif D(k)==30 & S(k)==4
    A(k)=22;
elseif D(k)==30 & S(k)==5
    A(k)=22;
elseif D(k)==31 & S(k)==0
    A(k)=23;
elseif D(k)==31 & S(k)==1
    A(k)=24;
elseif D(k)==31 & S(k)==2
    A(k)=25;
elseif D(k)==31 & S(k)==3
    A(k)=26;
elseif D(k)==31 & S(k)==4
    A(k)=27;
elseif D(k)==31 & S(k)==5
    A(k)=28;
elseif D(k)==32 & S(k)==0
    A(k)=23;
elseif D(k)==32 & S(k)==1
    A(k)=24;

```

```

elseif D(k)==32 & S(k)==2
    A(k)=25;
elseif D(k)==32 & S(k)==3
    A(k)=26;
elseif D(k)==32 & S(k)==4
    A(k)=27;
elseif D(k)==32 & S(k)==5
    A(k)=28;
elseif D(k)==33 & S(k)==0
    A(k)=23;
elseif D(k)==33 & S(k)==1
    A(k)=24;
elseif D(k)==33 & S(k)==2
    A(k)=25;
elseif D(k)==33 & S(k)==3
    A(k)=26;
elseif D(k)==33 & S(k)==4
    A(k)=27;
elseif D(k)==33 & S(k)==5
    A(k)=28;
elseif D(k)==34 & S(k)==0
    A(k)=23;
elseif D(k)==34 & S(k)==1
    A(k)=24;
elseif D(k)==34 & S(k)==2
    A(k)=25;
elseif D(k)==34 & S(k)==3
    A(k)=26;
elseif D(k)==34 & S(k)==4
    A(k)=27;
elseif D(k)==34 & S(k)==5
    A(k)=28;
elseif D(k)==35 & S(k)==0
    A(k)=23;
elseif D(k)==35 & S(k)==1
    A(k)=24;
elseif D(k)==35 & S(k)==2
    A(k)=25;
elseif D(k)==35 & S(k)==3
    A(k)=26;
elseif D(k)==35 & S(k)==4
    A(k)=27;
elseif D(k)==35 & S(k)==5
    A(k)=28;
elseif D(k)==36 & S(k)==0
    A(k)=23;
elseif D(k)==36 & S(k)==1
    A(k)=24;
elseif D(k)==36 & S(k)==2

```

```

        A(k)=25;
    elseif D(k)==36 & S(k)==3
        A(k)=26;
    elseif D(k)==36 & S(k)==4
        A(k)=27;
    elseif D(k)==36 & S(k)==5
        A(k)=28;
    elseif D(k)==37 & S(k)==0
        A(k)=23;
    elseif D(k)==37 & S(k)==1
        A(k)=24;
    elseif D(k)==37 & S(k)==2
        A(k)=25;
    elseif D(k)==37 & S(k)==3
        A(k)=26;
    elseif D(k)==37 & S(k)==4
        A(k)=27;
    else
        A(k)=28;
    end;
    end;
if c==11
    global tt
    tt=hist(A,28);
    clear mu0 u ud f w abs m k y S D t z s sold j k l i a;
    break;
end;
if c<=10
    global ha
    ha(c,:)=hist(A,28);
end;
clear A mu0 u ud f w abs m k y S D t z s sold j k l i a;
end;
global avg;
avg=mean(ha);                %%% computing the avg for ryan
spread=std(ha);              %%% computing the spread
w1=[0 -1; 0 1; 1 0; -1 0];
r=1;
for h=1:14
    g1(1)=avg(r+1)+ratio*spread(r+1); %%% calculating 1st layer biases for pairs 1 & 2,
3 & 4 ...27 & 28
    g1(2)=-(avg(r+1)-ratio*spread(r+1));
    g1(3)=-(avg(r)-ratio*spread(r));
    g1(4)=avg(r)+ratio*spread(r);
    b1(h,:)=g1;
    r=r+2;
    clear g1                %%% clear away the bias from previous values
end;
w2=[1 1 1 1];              % assign weights for layer 2 (fixed)

```



```

b2=[-4]; % assign bias for layer 2 (fixed)
rt=1;
for w=1:14 % this is the target wave file(tt)
    p(:,w)=[tt(rt);tt(rt+1)]; % pair them up [1 2] [3 4] .....
    rt=rt+2;
    v(:,w)=hardlim(w1*p(:,w)+b1(w,:)); % ouput from 1st layer perceptron note:
    b1(w,:) is the trained bias.
    total(w)=hardlim(w2*v(:,w)+b2); % ouput from 2nd layer perceptron
    Ryan=sum(total);
end;
if Ryan>=9
    target='Access Granted';
    if Ryan <=8
        target='Access Denied'
    end;
else target='Access Denied'
end;
set(handles.target_name,'String',sprintf('%c',target));
toc
timer=toc;
Ryan
close(br)
set(handles.timer,'String',sprintf(' Time taken: \n %f seconds',timer))
% -----
function varargout = File_Callback(h, eventdata, handles, varargin)
% -----
function varargout = Close_Callback(h, eventdata, handles, varargin)
clear
close(gcbf);
% -----
function varargout = Printer_Callback(h, eventdata, handles, varargin)
% -----
function varargout = Print_Callback(h, eventdata, handles, varargin)
print
% -----
function varargout = ViewFig_Callback(h, eventdata, handles, varargin)
% -----
function varargout = target_fig_Callback(h, eventdata, handles, varargin)
global u;
figure(1),plot(u);
% -----
function varargout = sound_recorder_Callback(h, eventdata, handles, varargin)
!SNDREC32.EXE
% -----
function varargout = Untitled_1_Callback(h, eventdata, handles, varargin)
global tt;
figure(2),bar(tt);
% -----
function varargout = text1_ButtonDownFcn(h, eventdata, handles, varargin)

```

```

% -----
function varargout = reference_Callback(h, eventdata, handles, varargin)
global avg
figure(3),bar(avg);
% -----
function varargout = drjohn_Callback(h, eventdata, handles, varargin)
% -----
function varargout = text6_ButtonDownFcn(h, eventdata, handles, varargin)
% -----
function varargout = text6_CreateFcn(h, eventdata, handles, varargin)
% -----
function varargout = saveas_Callback(h, eventdata, handles, varargin)
    [filename, pathname] = uiputfile( ...
        {'*.fig','*.mat'}, ...
        'Save as');
% -----
function varargout = pushbutton3_Callback(h, eventdata, handles, varargin)
pos_size = get(handles.figure1,'Position');
user_response = modaldlg([pos_size(1)+pos_size(3)/5 pos_size(2)+pos_size(4)/5]);
switch user_response
case {'no','cancel'}
    % take no action
case 'yes'
    % Prepare to close GUI application window
    %
    %
    %
    delete(handles.figure1)
end
% -----
function varargout = About_Callback(h, eventdata, handles, varargin)
% -----
function varargout = john_Callback(h, eventdata, handles, varargin)
user_response = drjohn
% -----
function varargout = marcus_Callback(h, eventdata, handles, varargin)
user_response = drmarcus

```

Program File (Dr_John.m)

```

function varargout = drjohn(varargin)
if nargin == 0 % LAUNCH GUI

    fig = openfig(mfilename,'reuse');

    % Use system color scheme for figure:
    set(fig,'Color',get(0,'defaultUicontrolBackgroundColor'));

    % Generate a structure of handles to pass to callbacks, and store it.

```

```

handles = guihandles(fig);
guidata(fig, handles);

if nargout > 0
    varargout{1} = fig;
end

elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK

    try
        if (nargout)
            [varargout{1:nargout}] = feval(varargin{:}); % FEVAL
switchyard
        else
            feval(varargin{:}); % FEVAL switchyard
        end
    catch
        disp(lasterr);
    end

end

% -----
function varargout = pushbutton1_Callback(h, eventdata, handles, varargin)
close(gcf)

```

Program File (Dr_Marcus.m)

```

function varargout = drmarcus(varargin)
if nargin == 0 % LAUNCH GUI

    fig = openfig(mfilename,'reuse');

    % Use system color scheme for figure:
    set(fig,'Color',get(0,'defaultUicontrolBackgroundColor'));

    % Generate a structure of handles to pass to callbacks, and store it.
    handles = guihandles(fig);
    guidata(fig, handles);

    if nargout > 0
        varargout{1} = fig;
    end

elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK

    try
        if (nargout)
            [varargout{1:nargout}] = feval(varargin{:}); % FEVAL
switchyard
        else
            feval(varargin{:}); % FEVAL switchyard
        end
    catch
        disp(lasterr);
    end

end

```

```

        else
            feval(varargin{:}); % FEVAL switchyard
        end
    catch
        disp(lasterr);
    end

end

% -----
function varargout = pushbutton1_Callback(h, eventdata, handles, varargin)
close(gcf)

```

Program File (modaldlg.m)

```

function answer = modaldlg(varargin)
error(nargchk(0,4,nargin)) % function takes only 0 or 4 argument
if nargin == 0 | isnumeric(varargin{1}) % LAUNCH GUI

    fig = openfig(mfilename,'reuse');

    % Use system color scheme for figure:
    set(fig,'Color',get(0,'defaultUicontrolBackgroundColor'));

    % Generate a structure of handles to pass to callbacks, and store it.
    handles = guihandles(fig);
    guidata(fig, handles);

    % Position figure
    if nargin == 1
        pos_size = get(fig,'Position');
        pos = varargin{1};
        if length(pos) ~= 2
            error('Input argument must be a 2-element vector')
        end
        new_pos = [pos(1) pos(2) pos_size(3) pos_size(4)];
        set(fig,'Position',new_pos,'Visible','on')
        figure(fig)
    end

    % Wait for callbacks to run and window to be dismissed:
    uiwait(fig);
    if ~ishandle(fig)
        answer = 'cancel';
    else
        handles = guidata(fig);
        answer = handles.answer;
        delete(fig);
    end
end

```

```

elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK

    try
        if (nargout)
            [varargout{1:nargout}] = feval(varargin{:}); % FEVAL
switchyard
        else
            feval(varargin{:}); % FEVAL switchyard
        end
    catch
        disp(lasterr);
    end
end
% -----
function varargout = noButton_Callback(h, eventdata, handles, varargin)
handles.answer = 'no';
guidata(h, handles);
uiresume(handles.figure1);
% -----
function varargout = yesButton_Callback(h, eventdata, handles, varargin)
handles.answer = 'yes';
exit

```