

# TECHNIQUES FOR HANDWRITTEN ARABIC NUMBERS RECOGNITION

By

Amjad Ahmad Hudaib

تعتمد كلية الدراسات العليا  
 هذه النسخة من الرسالة  
 التوقيع..... التاريخ: ١٠/١١/٩٩

Supervisor

Assoc. Prof. Dr. Ahmed Al-Jaber

Co-Supervisor

Asst. Prof. Dr. Mohammed Belal Al-Zoubi

Submitted in Partial Fulfillment of the Requirements

For the Degree of Master of Science in

Computer Science

Faculty of Graduate Studies

University of Jordan

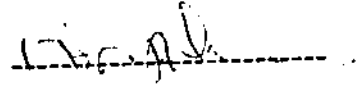
November 1999

This thesis was successfully defended and approved on 29/11/1999.

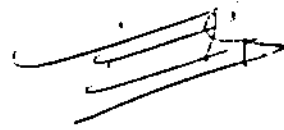
Examination Committee

Signature

Dr. Ahmed Al-Jaber, chairman



Assoc. Prof. of Algorithm Analysis

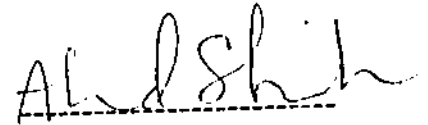


Dr. Mohammed Belal Al-Zoubi, member



Asst. Prof. of Graphics and Pattern Recognition

Dr. Ahmad Sharieh, member



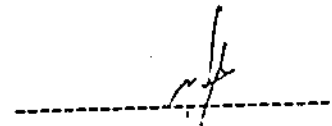
Asst. Prof. of Parallel Processing

Dr. Khalil Al-Hindi, member



Asst. Prof. of Artificial Intelligence

Dr. Abdulrauf Al-Hallaq, member



Asst. Prof. of Computer Networks

## ACKNOWLEDGMENT

I would like to express my gratitude to my supervisors Dr. Ahmed Al-Jaber and Dr. Mohammad Belal Al-Zoubi for their guidance and support.

I am grateful to all my professors, Dr. Ahmad Sharieh, Dr. Khalil Al-Hindi, Dr. Yahia Al-Halbi, Dr. Muneeb Qutaishat, Dr. Abdulrauf Al-Hallaq, and Dr. Sami Sarhan in computer science department at the University of Jordan. Also a special thanks to the University of Jordan for granted me a scholarship to attain the master degree in computer science.

I would like to take this opportunity to thank all my friends and colleagues for their support and motivation during my study especially Issa Qunbor, Hani Jaber, Mohammad Naser, Montaha Hamarsheh, Aseel Anani, Lubna Nasir, Bayan Abu-Shawer, Ansar Magdalena, Mohammad Al-Shraideh.

I am also in dept to Mr. Abdelmajeed Asha, Dr. Mohammad Al-Anani, Mr. Nasri Rawashdeh and his family for their help and support.

Finally, I would express my heart-felt gratitude to all my family especially my parents.

## TABLE OF CONTENTS

Chapter 1: Introduction	2
1.1 Importance of Handwritten Arabic Number Recognition	3
1.2 Artificial Neural Networks	4
1.3 Fuzzy Logic and Neuro Fuzzy Systems	6
1.4 Thesis Outline	8
Chapter 2: Literature Review	11
2.1 Pattern Recognition	11
2.1.1 Introduction	11
2.1.2 Definition of Pattern Recognition	12
2.1.3 Image representation	13
2.1.4 Classification	13
2.1.5 Pattern Recognition Approaches	15
2.1.5.1 The Conventional Pattern Recognition	16
Approaches	
2.1.5.1.1 Statistical Pattern Recognition	16
Approach	
2.1.5.1.2 Syntactic Pattern Recognition Approach	17
2.1.5.2 Artificial Intelligence Based approaches	20

2.1.6 Classifiers	21
2.1.6.1 Linear Discriminate Classifier	22
2.1.6.2 Quadratic Discriminate Classifier	23
2.1.6.3 Bayes Rule classifier	24
2.1.6.4 K-Nearest Neighbor Classifier	25
2.1.6.5 Artificial Neural Network Classifier	27
2.1.7 Handwritten Arabic Numbers Recognition	28
2.1.7.1 Historical Background	28
2.2 Artificial Neural Networks	32
2.2.1 Introduction	32
2.2.2 Historical Development of Artificial Neural Networks	33
2.2.3 Fundamentals of Neural Networks	35
2.2.3.1 Processing Units	35
2.2.3.2 Connections	36
2.2.3.3 Computation Procedures	36
2.2.3.4 Training Procedures	38
2.2.4 Taxonomy of Neural Networks	39
2.2.4.1 Supervised Learning	39
2.2.4.2 Unsupervised Learning	40
2.2.4.3 Reinforcement Learning	40

2.2.5 Backpropagation	41
2.2.6 Backpropagation Learning with Momentum Term	48
2.2.7 Resilient Propagation (RPROP)	49
2.3 Fuzzy Logic and Neuro Fuzzy Systems	52
2.3.1 Fuzzy Logic	53
2.3.1.1 Fuzzy Set Theory	53
2.3.1.2 Fuzzy set Operations	58
2.3.1.3 Fuzzy Rules	59
2.3.1.3.1 Linguistic Variable	60
2.3.1.3.2 IF-THEN-RULES	60
2.3.1.4 Fuzzy Reasoning	61
2.3.1.5 Fuzzy Systems	61
2.3.2 Neuro Fuzzy Systems	65
2.3.2.1 What is Neuro Fuzzy Systems?	65
2.3.2.2 Neuro Fuzzy Classification	67
2.3.2.3 Components of Neuro Fuzzy Systems	67
2.3.2.4 Neuro Fuzzy Learning	70
2.3.2.4.1 Training of Neuro Fuzzy Systems	71
2.3.2.4.2 Adjusting Fuzzy Parameters	72
Value	72

2.3.2.5 Neuro Fuzzy Modeling	74
Chapter 3: Handwritten Arabic Numbers Recognition Using Artificial Neural Networks	79
3.1 Introduction	79
3.2 Preprocessing	81
3.2.1 Segmentation Operation	81
3.2.2 Normalization Operation	82
3.2.3 Binarization Operation	83
3.2.4 Location operation	83
3.3 Recognition of Handwritten Arabic Numbers Using Artificial Neural Networks	84
3.3.1 Data Sets	84
3.3.1.1 Testing and Training Sets	86
3.3.2 The Neural Networks Architecture	87
3.3.3 Experimental Results	92
3.4 Conclusion	101
Chapter 4: Handwritten Arabic Number Recognition Using Neuro Fuzzy Systems	105
4.1 Introduction	105

4.2 Neuro Fuzzy System Architecture	106
4.3 Experimental Work	110
Chapter 5: Conclusions and Recommendation	117
5.1 Conclusions	118
5.2 Future Work	123
References	124
Appendices	132
Appendix A: Samples of Handwritten Digits	133
Appendix B: Source Code	134
Arabic Abstract	143



## TABEL OF TABLES

Table 3.1 Neural Network Experiments with (1000 Samples and 0.01 Error Goal)	93
Table 3.2 Neural Network Experiments with (2000 samples and 0.01 Error goal)	96
Table 3.3 NNW Result of Digits Recognition Using 2000 sample for Training	97
Table 3.4 Average Generalization Ratio of Recognition Each Digit Using Neural Networks	98
Table 3.5 Comparison between Neural Networks with 1000 samples and 2000 samples for Recognition Testing sets	100
Table 4.1 Neuro Fuzzy System Experiments	111
Table 4.2 Neuro Fuzzy System Result of Digits Recognition	112
Table 5.1 Comparison between Neural Network and Neuro Fuzzy Systems For Recognizing Training Sets	119
Table 5.2 Comparison between Neural Network and Neuro Fuzzy Systems For Recognizing Testing Sets	121

TABEL OF FIGURES

Figure 2.1 Classification Steps	14
Figure 2.2 Stages of Statistical Pattern Recognition Approach	17
Figure 2.3 Block Diagram of the Syntactic Pattern Recognition System	18
Figure 2.4 Liner Classifier in 2-dimension	23
Figure 2.5 Using Nearest Neighbor Algorithm for Classifying Point X and Y	26
Figure 2.6 Simple Artificial Neural Network	37
Figure 2.7 Typical Neural Network Activation Function	38
Figure 2.8 Backpropagation Network Architecture	41
Figure 2.9 Architecture of Backpropogation	42
Figure 2.10 MLFF Network Connection and Variable.	44
Figure 2.11 Fuzzy Set and Crisp Set	55
Figure 2.12 Illustration of Various Membership Functions	57
Figure 2.13 Fuzzy Rule Base System	64
Figure 2.14 Neuro Fuzzy Network	68
Figure 2.15 ANFIS Architecture for First-Order Sugeno Fuzzy Model	75
Figure 3.1 Steps of Handwritten Arabic Numbers Recognition Systems	80
Figure 3.2 Digit 0 in its Location	84
Figure 3.3 Samples of Handwritten Digits	85
Figure 3.4 Neural Networks Architecture	89
Figure 3.5 Convergence of Mean Square Error	91
Figure 3.6 Generalization Ratio Curves of Training and Testing Sets	94
Figure 3.7 Generalization Ration Curve of Recognition of Training and Testing Sets with (2000 samples)	97

Figure 3.8 Digits Recognition Using Neural Network	99
Figure 3.9 Comparison between NNW1 and NNW2 for Training Sets	101
Figure 3.10 Comparison between NNW1 and NNW2 for Testing Sets	102
Figure 4.1 Neuro fuzzy System Architecture	109
Figure 4.2 Digits Recognition using Neuro Fuzzy System	113
Figure 4.3 Generalization Ration of Training Data Using Neuro Fuzzy Systems	114
Figure 4.4 Generalization Ration of Testing Data Using Neuro Fuzzy Systems	115
Figure 5.1 Neural Network Vs Neuro Fuzzy Systems For Recognizing Training Sets	120
Figure 5.2 Neural Network Vs Neuro Fuzzy Systems For Recognizing Testing Sets	122

ABSTARCT

TECHNIQUES FOR HANDWRITTEN ARABIC NUMBERS  
RECOGNITION

By

Amjad Ahmad Hudaib

Supervisor

Assoc. Prof. Dr. Ahmed Al-Jaber

Co-Supervisor

Asst. Prof. Dr. Mohammed Belal Al-Zoubi

The automatic recognition of a handwritten Arabic numbers is an extremely important field that receives attention from researchers. Because of its importance, it became a benchmark problem for different approaches. This effort is certainly justified as there are numerous industrial, business applications, and many other where automated recognition will result in substantial cost.

This thesis is concerned with proposing techniques for recognition a handwritten Arabic numbers by using artificial neural network and neuro fuzzy systems techniques. These techniques input scanned images that contain the numbers. These images are then

passed through many preprocessing operations such as segmentation, binarization, normalization, and location, developed to prepare the image for the system that designed for recognition. The first technique is based on multi layer feed forward backpropagation neural network that has been tested on a data reached to 2000 samples collected from different persons. This technique produces generalization ratio reaches to 91.6 %. The second technique based on neuro fuzzy systems that designed and tested on the same data, the generalization ratio producing by using neuro fuzzy system reaches to 95.85%.

Consequently, results show that both neural network and neuro fuzzy system are suitable for handwritten Arabic numbers recognition, and neuro fuzzy system give more accurate results.

## Chapter 1

### Introduction

Computer power has increased over the years; one of the major goals of research into computer science has been to make computer easier to communicate with and to make their benefits available to much greater number of people. One of the major obstacles to the integration of a universal computer system is the fact that most useful data still written and stored on paper. In many situation it would be highly desirable to process the contents of these paper by computer. Computer handwritten recognition systems offer a new way of improving the human-computer interface and of enabling the computer to read and process the many handwritten documents especially numbers for the wide and sensitive role of numbers, as well as, business, banking, archiving, mathematics, physics, and so on.

The recognition of handwritten Arabic numbers is an extremely important subject that has received attention by many researchers in the past and in recent times. One of the very well known approaches to deal with this subject is mainly concentrated on scanning these numbers by a special machine like the scanner and then by the computer recognizing them as images. The drawback behind this approach is that a huge amount of storage spaces is needed to store the given images. The other problem is the large

amount of time required to process these numbers, as we cannot deal with this data as numbers and use this value; but just as image. Another big problem with several previous approaches is the quality. That is, the result of recognition may have a large error. In the attempt to solve these drawbacks, some techniques will be implemented in this thesis to recognize the handwritten Arabic numbers by recognizing the digits that the numbers consist of. Also in our approach, the quality of recognition is improved by having the error less than the errors obtained by many of the previous researchers.

### 1.1 Importance of Handwritten Arabic Number Recognition

A handwritten Arabic number recognition is an extremely important field; some of its importance can be summarized in the following points:

1. To recognize handwritten Arabic numbers while distinguishing the values of these numbers.
2. Reducing the storage requirement for scanned image files and replace them with numeric value. This is important for some applications such as electronic mailing and Internet services.
3. Automating and improving the speed and accuracy of the processing input pattern.
4. Arabic number handwritten recognition systems can be very useful in the domains in which numbers have very important and sensitive role like banking systems, and archiving.
5. Recognizing archiving documents.
6. Facilitating the editing process, and updating on the resulting files.

7. Reducing postal office's overhead costs by recognizing the handwritten zip code automatically.

In this thesis, we will implement techniques for recognizing handwritten Arabic numbers. These techniques are based on artificial intelligence methods like neural network. We will build a neural network system to recognize the digits that the numbers consists of. After that, we combine neural networks , fuzzy logic and fuzzy set, to build a neuro fuzzy system that will be used to recognize digits.

Artificial neural network and, neuro fuzzy system have powerful characteristics. These characteristics are summarized in the following section.

## 1.2 Artificial Neural Networks

Artificial neural networks field, was initially inspired by neurobiology, but it has since become a very interdisciplinary field, computer science, electrical engineering, mathematics, and other fields.

Many researchers, especially brain modelers, have been exploring artificial neural networks; they model the brain as a continuous-time nonlinear dynamic system in connectionist architectures that are expected to mimic brain mechanisms to simulate intelligent behavior. Such connectionism method replaces symbolically structured representations with distributed representation in the form of weights between a massive set of interconnected neurons (or processing units).



Much attention is now being focused on the general properties of neural computation, using simplified neural models. These properties include:

- Learning

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

- Generalization.

Networks don't just memorize the training data; rather, they learn to recognize the underlying patterns. This is essential in handwritten digit recognition, because input images of handwritten are never exactly the same.

- Nonlinearly:

Networks can compute nonlinear, nonparametric functions of their input, enabling them to perform arbitrarily complex transformation of data. This is useful, as recognizing handwritten numbers is a highly nonlinear process.

- Robustness

Networks continue to perform well when part of the network is disabled or damaged, or when presented pattern with noisy data. This is possible because the knowledge stored in neural network is distributed over many neuron and interconnection(Patterson,1996).

- Parallelism.

Networks are highly parallel in nature, so they are well suited to implementations on massively parallel computers. This will ultimately permit very fast processing of recognition digits.

### 1.3 Fuzzy Logic and Neuro Fuzzy Systems

The human brain interprets imprecise and incomplete information. Fuzzy set theory provides a systematic calculus to deal with such information linguistically, and it performs numerical computation by using linguistic labels stipulated by membership functions. Moreover, a selection of fuzzy if-then rules forms the key component of a fuzzy application. Unfortunately, fuzzy set theory lacks the adaptability to deal with changing external environments. Thus, we incorporate neural network learning concepts in fuzzy inference systems, resulting in neuro fuzzy systems. A neuro fuzzy system is a fuzzy system that uses a learning algorithm derived from or inspired by neural network theory to determine its parameters (fuzzy sets and fuzzy rules) by processing data samples. Therefore, the neuro fuzzy systems characteristics are:

- Neuro fuzzy human expertise utilizes human expertise in the form of fuzzy if-then rules, as well as in conventional knowledge representations, to solve practical problems.
- Artificial neural networks are employed extensively in neuro fuzzy systems to deal with pattern recognition, nonlinear regression and classification problems.

- Neuro fuzzy system computation relies mainly on numerical computation. Numerical computation, neuro fuzzy systems have therefore found a number of new application domains. These application domains are mostly intensive computation and include adaptive signal processing, adaptive control, nonlinear system identification, nonlinear regression, and pattern recognition.
- Model-free learning Neural networks and adaptive fuzzy inference systems have the ability to construct models using only target system sample data. Detailed insight into the target system helps set up the initial model structure, but it is not mandatory.
- Parallelism, neuro fuzzy systems are highly parallel, which will permit very fast processing of the problem like digit recognition.
- Intensive computations without assuming too much backgrounds knowledge of the problem being solved, neuro fuzzy and soft computing heavily on computation to find rules or regularity in data sets.
- Robustness and fault tolerance: both neural networks and fuzzy inference systems exhibit fault tolerance. The deletion of a neuron in a neural network, or a rule in fuzzy system, or presenting pattern with noise, does not effect the performance of the network significantly because of its parallel architecture and the knowledge stored in a distributed neurons and connections.

492631

## 1.4 Thesis Outline

This thesis investigates the recognition of handwritten Arabic numbers. The investigation is done by designing techniques and systems that can recognize digits that the numbers consists of.

In Chapter 2, we will present literature review of subjects that related to our thesis in three major sections.

In Section 2.1, we will present some basic concepts of pattern recognition, classification, and pattern recognition approaches such as conventional approaches and artificial intelligence approaches. Also we will introduce the classifier concepts and the most popular approaches used to implement the classifier.

In Section 2.2, a historical development artificial neural network, fundamentals, and taxonomy of artificial neural will be introduced. Also we will introduce well-known learning algorithms such as Backpropagation, Backpropagation learning with momentum, and Resilient Propagation.

In Section 2.3, we will present fuzzy logic and neuro fuzzy systems.

In fuzzy logic we will present detailed introduction to the theory and terminology of fuzzy disciplines, including fuzzy sets, fuzzy set operations, fuzzy rules, fuzzy reasoning and fuzzy systems.

In neuro fuzzy system, a variety of important neuro fuzzy system paradigms will be introduced by introducing advantages of combining the neural network and fuzzy logic theory. A neuro fuzzy classification, neuro fuzzy systems, learning procedures and, neuro fuzzy models will be introduced.

In Chapters 3,4,5 a detailed description of our work is given.

In Chapter 3, we will introduce the database set, and the preprocessing operations.

After that, we will introduce the artificial neural network that we designed for our experiments, including training and testing sets, and our neural network system by discussing the architecture, training and learning procedures that we use. After that, we will introduce and discuss the results that are obtained from our experiments.

In Chapter 4, a neuro fuzzy system that we designed to recognize digits will be introduced. After preprocessing operations, architecture of the neuro fuzzy system and the learning procedure will be discussed. Finally, we will discuss the experiments and their results.

Finally, Chapter 5 focuses on the results of our experiments. Comparisons between the various experiments and the results obtained are displayed. Suggestion for future work related to the experiments are put forward.

## Chapter 2

### Literature Review

## Chapter 2

# Literature Review

## 2.1 Pattern Recognition

In this section, we will present some basic concepts in the field of pattern recognition, which include definition of pattern recognition, image representation, and classification. Moreover, the pattern recognition approaches such as conventional approaches and artificial intelligence approaches, statistical approaches, and syntactic approaches are dealt with.

Also we will explain the classifier concepts and the most known function used to implement the classifier. Finally the importance and the historical background of a handwritten Arabic number recognition which is one of the most important application of pattern recognition will be introduced.

### 2.1.1 Introduction

One of the most basic and essential characteristics of human ability is to recognize and identify pattern such as faces of people, sounds, handwriting,..etc. This ability of recognizing patterns has motivated many researchers towards the discovery of mechanical and artificial methods for recognizing patterns. The results of pattern recognition researches developed in the early 1960's, as well as those developed

recently have been impressive: Numerous applications such as handwritten numbers recognition, character recognition, speech recognition, weather forecasting, medical analysis, satellite and aerial-photo interpolation, have been carried out.

The contributions of this development and the growth of researches have come from many disciplines: Statistics, communication theory, switching theory, control theory, operations research, biology, psychology, linguistics, and computer science.

### 2.1.2 Definition of Pattern Recognition

Pattern recognition can be defined as a process of identifying or discriminating objects in an image on the bases of information available about these objects (Morton and Eric,1993).

As we see, in pattern recognition there is the pattern term, which is a quantitative or structural description of an object or some other entity of interest in image, and there is the term recognition, which is to identify from knowledge or character of the pattern.

Hence there are classes of objects in pattern, that can be distinguished according to a certain feature or attributes. Pattern recognition can be also defined as a process of decision making in which the input pattern is recognized as a member of a class according to its attributes, or comparing its attributes to already known common attributes of that class.

Therefore, in order to build a pattern recognition system, a prior knowledge about the problem is necessary. This knowledge is known as the learning set (in the form of set),



this set is a set of features such as (length, area, ..., etc) and also may be raw data such as images of handwritten digits.

### 2.1.3 Image representation

Image can be identified as a two dimensional intensity function  $f(x, y)$  where  $x$  and  $y$  denote spatial coordinates and the value of the function ( $f$ ) at any point  $(x, y)$  is proportional to the brightness (or gray level in monochrome image) of the image at that point (Gonzales and Winter, 1987).

We may consider an image as a spatial representation of an object, represented by matrix with rows and columns, where a given point in the image or corresponding matrix elements value identifies the level of brightness at these point.

Therefore, image consists of elements that described by the spatial position in image and the intensity value associated with this position. Sometimes it is called picture elements, cell, or pixels (which is the most used).

### 2.1.4 Classification

Classification is the most related concept to pattern recognition, where the aim of classification is to partition a data set into subset or clusters of samples so that the degree of similarity crops up high among members of the same cluster and low between members belonging to different clusters.

A cluster is a homogeneous group of units or features which are very like one another. Likeness between units is usually determined by association, similarity or distance between measurement pattern associated with the units (Haralick and Shapiro, 1989).

Classification, on the other hand, is to assign an object or pattern into one of several prespecified classes based on the extraction of significant features and the processing or analysis of these features (Song, 1998), i.e. classification depends on the ability to discover common attribute among patterns. Figure 2.1 shows the steps to assign class for input image.

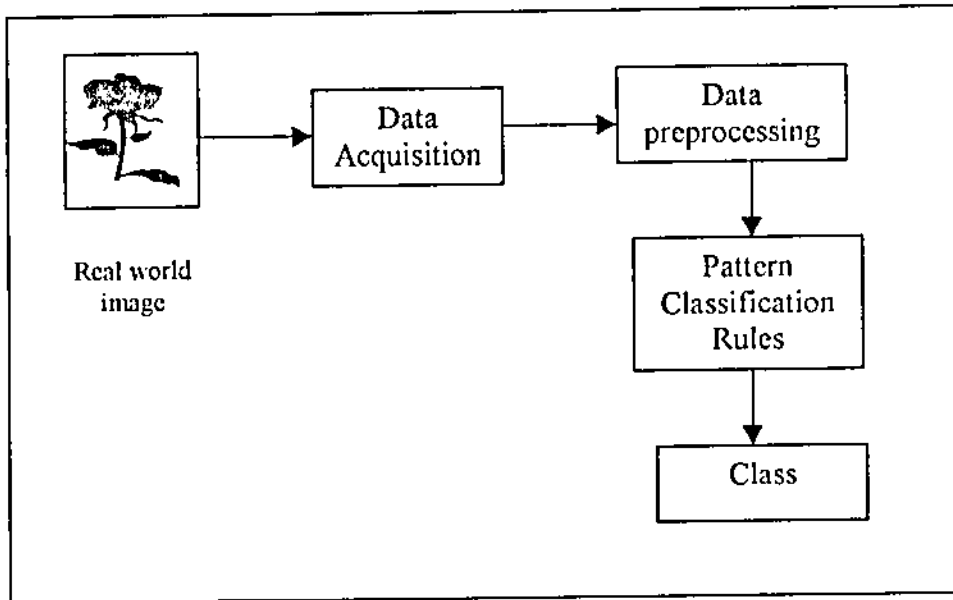


Figure 2.1 Classification Steps

As we see in this figure, we have three phases for entire real world image to be classified in its correct class. The first phase is data acquisition, which is concerned with collection analog data to be converted to a format suitable for computer processing. The second phase is concerned with the input data grouped into a set of characteristic features. The third phase is restricted with methods of classification and rules to be applied on the features of the pattern and the product in the class that the input pattern belongs to. This output can be produced by making cluster analysis. The Cluster analysis is a tool that attempts to assess the interaction among pattern by organizing them into groups or clusters such that, patterns within a cluster are more similar to each other than patterns belonging to a different cluster, the result of cluster analysis can be used to formulate hypotheses about the data, to classify new data (Augusteijn and Steck, 1998).

There have been numerous techniques investigated and developed for classification. We will discuss some of most popular used techniques

### 2.1.5 Pattern Recognition Approaches

Many techniques to pattern recognition have been developed. Generally these techniques can be grouped into two categories (Nandhakumar and Aggarwal, 1985), the first one is called the conventional pattern recognition approaches and the other is the artificial intelligence based approach.

### 2.1.5.1 The Conventional Pattern Recognition Approaches

The conventional pattern recognition approaches are divided into two approaches: statistical and structural.

#### 2.1.5.1.1 Statistical Pattern Recognition Approach

Statistical pattern recognition approach classifies the pattern into one of a finite number of classes, in which a multivariate probability distribution function is assumed to exist. A distribution function may be known in prior or it may be estimated from training set (Nandhakumar and Aggarwal, 1985).

A set of features is specified as being distinguishing features. These feature values are used to classify the pattern to its class with minimum error or minimum cost based on them. Therefore, statistical pattern recognition approach consists of two stages:

1. Extracting suitable features (distinguishing features) from the pattern by using suitable statistical method. These features are supposed to be invariant or less sensitive to variation and distortion (Al-Wailey and Ramzi, 1989).
2. Recognition or classifying the pattern into its class based on a set of selected features using suitable function with minimum error. These stages can be seen in Figure 2.2 where we can see feature extraction of the input pattern is performed, then according to these extracted features and the relevant classification rules, the input pattern is assigned to the class that it belongs to.

Statistical pattern recognition techniques are domain independent and the algorithms are easily transported to domains other than the ones they were initially developed for (Nandhakumar and Aggarwal, 1985).

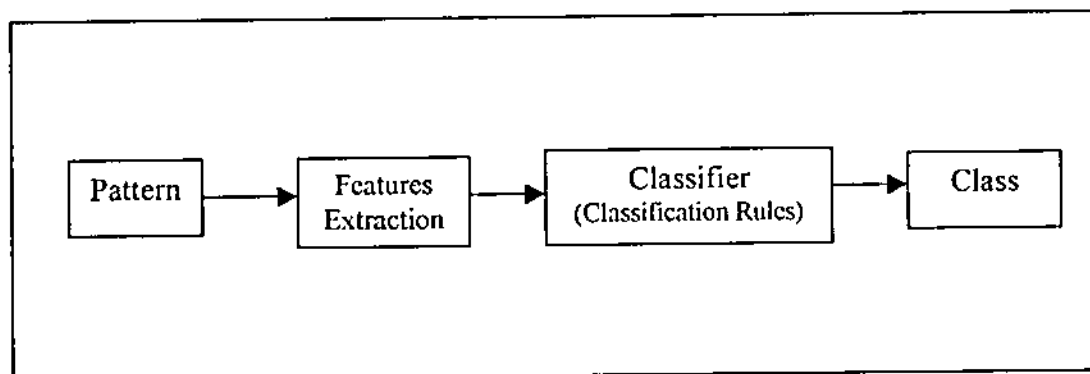


Figure 2.2 Stages of Statistical Pattern Recognition Approach

#### 2.1.5.1.2 Syntactic Pattern Recognition Approach

The pattern in syntactic approach (some times called linguistic or structural) is described in terms of its structural information, i.e. recognition in the form of sentences from the language of a phrase structure grammar (Haralick and Shapiro, 1989).

Syntactic pattern recognition is based on some concepts from the formal language theory whose origin traced to development of mathematical models of grammars by Noam Chomsky (Morton and Eric, 1993).

This approach decomposes the pattern into sub patterns of primitives; so the pattern is described in terms of simpler subpatterns which are much easier to recognize than the original pattern. The recognition of each pattern is usually made by parsing the pattern structure according to syntax rules, so we can use a hierarchical representation of the image.

Structural pattern recognition may be based on the “analysis by synthesis” principle (Nandhakumar and Aggarwal, 1985), this approach draws an analogy between the structure of the patterns and the syntax or grammar of natural languages.

The primitives or sub pattern contingent with each other by a sentence, a string, or tree which is specified by a grammar.

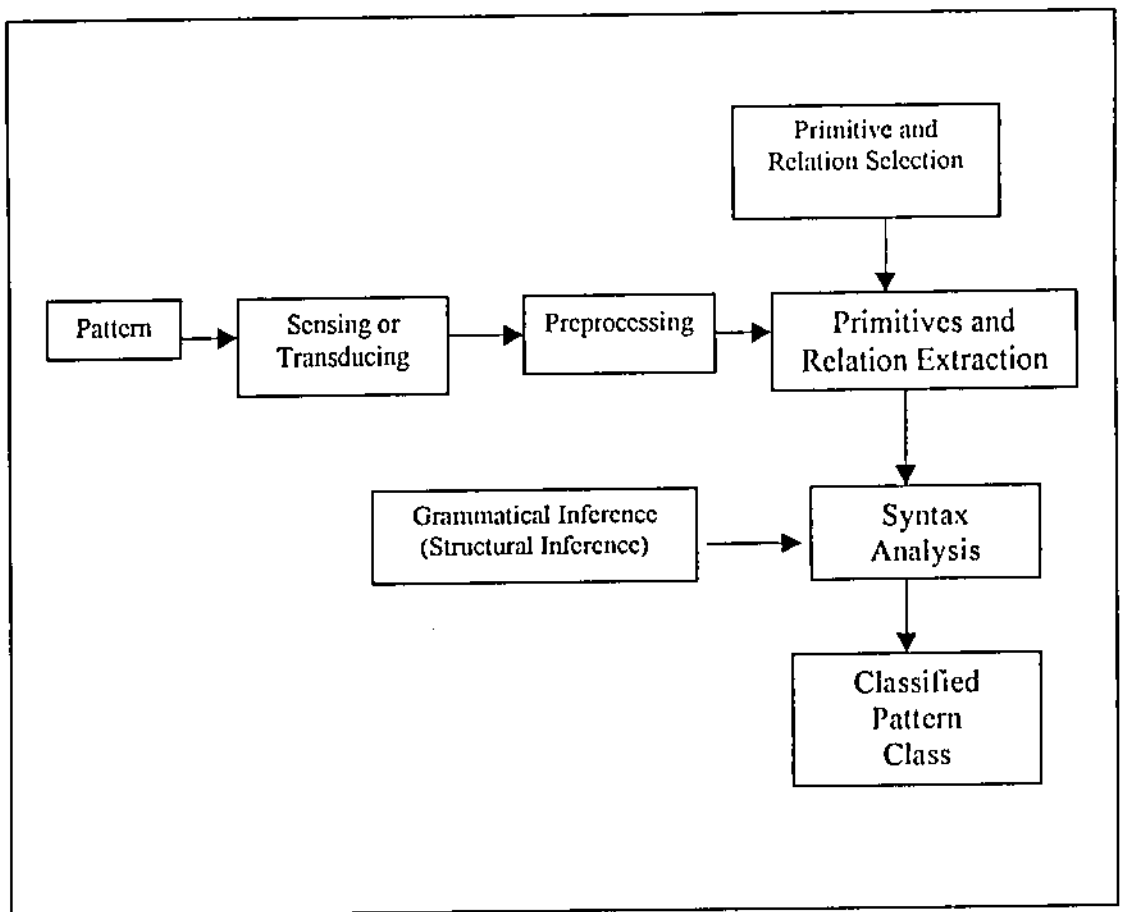


Figure 2.3 Block Diagram of the Syntactic Pattern Recognition System

Figure 2.3 shows a block diagram of syntactic approach of the pattern, where a given input pattern is converted to a suitable form for computer process; a transducing step

where the pattern is converted to a suitable form that can be used in the recognition steps. Then different operations can be applied to make the recognition process simpler. These operations are called preprocessing operations. After that each preprocessed pattern is segmented into sub patterns and pattern primitive according to the set of primitive elements selected before. These primitives are extracted and their relation are specified in order to be used with suitable grammar in the next step of the recognition problem. Finally, we assign the class that the pattern belongs to according to the grammar of primitives that we have.

Syntactic pattern recognition system can be considered as a composition of two parts: Analysis and recognition (Al-Wailey and Ramzi, 1989). The analysis part consists of primitive selection and grammatical or structural inference, i.e. takes sample patterns and select a set of primitives, which have a complete information about the pattern. These primitives are used to describe the pattern in terms of a specified structural relation. So input pattern is given and a suitable grammar is obtained at the output. The recognition part consists of preprocessing, primitive extraction including relation among primitives, and syntax analysis. So it takes the input pattern and gives a classified pattern at the output. i.e performing a syntax analysis to determine that the pattern is syntactically correct with respect to the specified grammar syntactic pattern recognition.

### 2.1.5.2 Artificial Intelligence Based approaches

One of the most commonly used techniques is to use artificial intelligence methods. These methods are either supervised where a "teacher" provides output targets for each input pattern and correct the network error explicitly through learning process, or unsupervised pattern recognition where there is no teacher to present target output through learning process.

The objective of artificial neural networks is to understand how the brain provides human beings with such abilities e.g. perceptual interpretations reasoning, learning, using these abilities to solve a complex application such as pattern recognition voice recognition, fuzzy pattern matching and nonlinear discrimination.

In artificial neural networks, self organizing and ability to learn through organizing and reorganizing in response to learning rule are achieved by producing which produce supervised and unsupervised learning.

In supervised learning we have a training set and testing set. Both sets have patterns and their targets, i.e. the particular class to which the pattern belongs. Through the learning, processing the training set will produce the output, which has to be the same form as the target. So we have guidance in the form of class information.

Unsupervised learning on the other hand has the possibility of exploring the structure of data without guides in the form of class information. It can often reveal features not previously expected or known about. These might include the division of data into a number of smaller groups. Each group has its separate identifiable properties More details will be given in section 2.3 about the artificial neural networks.



### 2.1.6 Classifiers

The aim of pattern recognition is to design classifiers, which is a mechanism or technique that takes features of input pattern and produce a value indication to which class that pattern belongs to. This object can be achieved from the set of information already given about a known class of given pattern and set will be called the learning set.

Mathematically, classifiers mapping the input pattern into a vector that illustrates to which class this pattern belongs within an acceptable amount of error, where the error is the difference between the calculated output and the real output. Formally:

Let

X: vector on n features that extracted from input pattern.

C: vector indicates the class to where that pattern belongs.

f: a function returns a value, that specifies the class, of the input vector. This value calculates according to the input vector. Therefore,

$$C = f(X) \quad (2.1)$$

This function assigns each input unit to a cluster or a class, in the basis of its corresponding features (i.e. f: measurement space  $\rightarrow$  set of cluster, mapping from measurement space to the set of clusters).

Classifiers can be divided into two types: parametric classifier and non-parametric classifier (Belkasim et al., 1992), where parametric classifier assumes a functional distribution of a given sample for example Bayes classifier. And the non-parametric

classifier does not assume any functional distribution of a given sample, for example K-nearest neighbor rule.

To implement a classifier, numerous functions can be used. We will discuss in the next subsections some of these classifier functions.

### 2.1.6.1 Linear Discriminate Classifier

A discriminant function  $f_i$  is a scalar function that separates its measurement space into two clusters, where the domain of  $f_i$  is usually measurement space and the range is usually real number. When  $f_i(p) > f_k(p)$ , for  $k=1,2, \dots, k$  then the decision rule assigns the  $i$ 'th category to the unit give rise to pattern ( $p$ ) (Haralick and Shapiro, 1989).

In other words, it is a function that separate the measurement space into more than one clusters, and assigns the input pattern to the cluster that belongs. Therefore, a linear discriminant function is a discriminant function that partitions measurement space using liner boundaries line in 2-dimensional space, for instance Figure 2.4 shows an example where the line that represent the given function divided the space into two partitions each one belongs to different class.

Formally, let the input pattern  $P$  represented by  $(x_1, x_2, \dots, x_n)$  then,

$$f(p) = \sum_{i=1}^n a_i x_i + a_0 \quad (2.2)$$

where:

$a_i$ : constant .

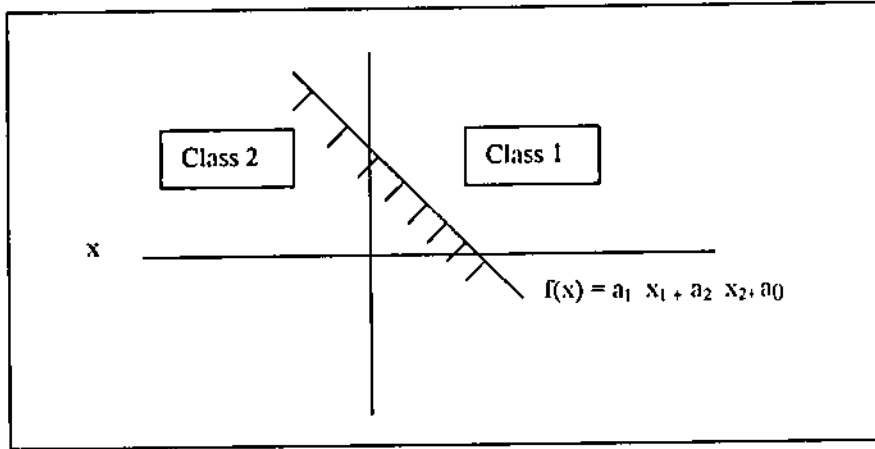


Figure 2.4 Liner Classifier in 2-dimension

Example of linear discriminant function is a well-known Fisher's linear discriminant function. It is the oldest classification procedure (1936) and most commonly used in computer packages (Michie et al., 1994). And its idea is to divide sample space by a sense of lines in two –dimensions, planes in three–dimensions...etc. the line divides two classes is drawn to bisect the line joining the center of those classes. Fisher viewed separation and classification together and used approach based on distance between groups.

### 2.1.6.2 Quadratic Discriminate Classifier

The quadratic discriminate classifier is simply defined as the logarithm of appropriate probability density function, so that quadratic discriminate classifier is calculated for each class (Michi et al., 1994). Therefore, it is similar to linear discriminate classifier but the region or space is divided by quadratic surface (circle, ellips, parabola...etc) instead of lines. Formally, the quadratic discriminate classifier is a discriminant function of the form

$$f(p) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} x_i x_j + \sum_{i=1}^n a_i x_i + a_0 \quad (2.3)$$

Where

P: the input pattern represented by  $(x_1, x_2 \dots x_n)$ ,

$a_i, a_{ij}$  : constants.

For example, the quadratic form with two variables:

$$f(p) = a_1 x_1^2 + a_2 x_1 x_2 + a_3 x_2^2 + a_4 x_1 + a_5 x_2 + a_6 \quad (2.4)$$

The quadratic discriminate classifier procedure starts by constructing an ellipse for each sample, for examples, centered on the center of gravity points, i.e. this ellipse contains patterns or points, which are likely hood all points with the ellipse and classified as one class.

### 2.1.6.3 Bayes Rule classifier

In classification statistical techniques can be used to obtain optimal functions for classification. Where, statistical pattern recognition we use information about random components and the deterministic component of the design set to obtain optimal discriminates, this information is in the form of the measurement vector in the design set.

Bayes Rule classifier is based on minimizing the total expected loss incurred by misclassification (Belkasim et al., 1992). In other words, the decision rule with least probability is to allocate the class with the highest probability of occurrence, the relative probability is the conditional probability  $P(A_i | x)$  which is given by:

$$P(A_i | x) = \frac{\prod_i p(A_i | x)}{\sum_{j=1}^n \prod_i p(A_i | x)} \quad (2.5)$$

#### 2.1.6.4 K-Nearest Neighbor Classifier

K-Nearest Neighbor classifier find (k-1) or more surfaces dividing the space into K or more regions such that, all the samples in one region form only one pattern set and every point in the decision surface is equidistant from two nearest samples.

The procedure started by finding K-neighbors from the closest class, then by using distance formula any K'th distance as a threshold, all classes with n distance exceeding that threshold are eliminated. The procedure is repeated until the threshold value becomes larger than n distance.

The distance  $D(X, Y)$  formula in 2-dimension is given in formula 2.6.

$$D(x, y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (2.6)$$

Where:

$$X = (x_1, y_1)$$

$$Y = (x_2, y_2)$$

Therefore, this algorithm classify unknown pattern  $(x,y)$  by computing the Euclidean distance from this point to every other known data points and find the nearest neighbor. Then assume the unknown has the same identity as its nearest neighbor. For instance, in Figure 2.5 we can assume that point marked as ( X ) forms class A, and the point marked as ( Y ) forms class B, where the space is divided up using 12 data points. In actual use, the more data points we have available, the better of classification accuracy will be (Donald, 1998).

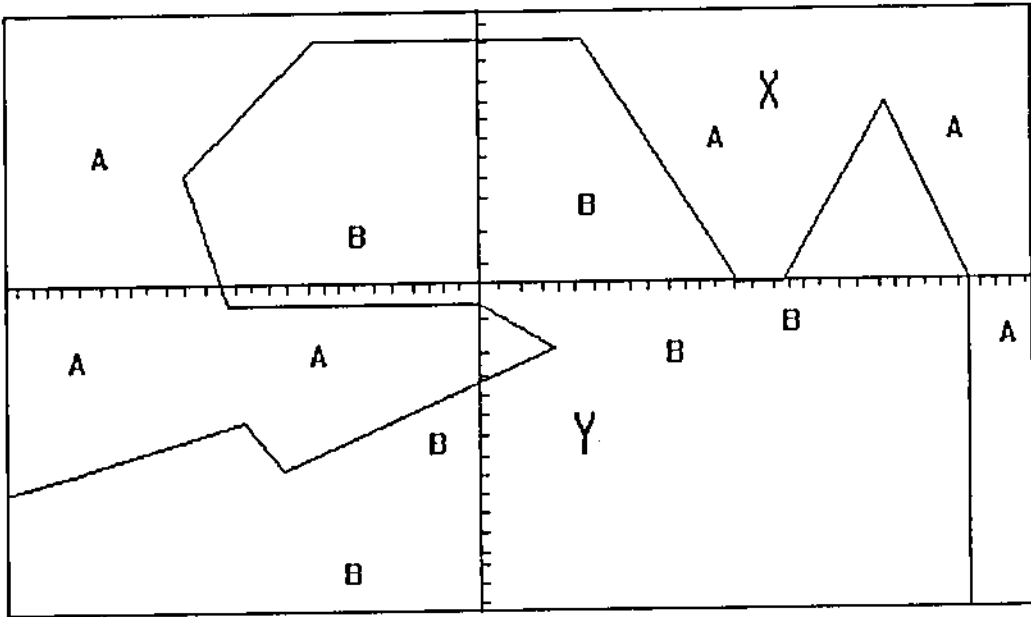


Figure 2.5 Using Nearest Neighbor Algorithm for Classifying Point X and Y

### 2.1.6.5 Artificial Neural Network Classifier

In Artificial Neural Network classifier, the input values are fed in parallel via input connections, each connection carries an analog value which may take on two values of binary inputs or may vary over a large range for continuous valued input (Lippman, 1987).

After classification is complete the most likely class will be on strongly or "high" value of output.

If the correct class is provided, then this information and classifier output can be fed back to the first stage of the classifier to update the parameters which called weights using stable learning algorithm (Atlas et. al., 1990). Adaptive will make a correct response more likely for succeeding input pattern that similar to current pattern.

Generally, The artificial neural network approach has many similarities with statistical pattern recognition concerning both the data representation and the classification principles, but the practical implementation is, however, very different. The analysis mode involves the configuration of a network of artificial neurons and the training of the net to determine how the individual neurons can effect each other. The recognition mode involves sending data through the net and evaluating which class got the highest score. This subject will be discussed in more detailed in section 2.2.

## 2.1.7 Handwritten Arabic Numbers Recognition

The recognition of a handwritten Arabic numbers is an extremely important subject and one of the most important applications of pattern recognition. This field receives the attention of many researchers in the past and recent time for its importance. Great efforts have been exerted to develop this field since it represents a successful solution to manipulate large quantity of data automatically.

### 2.1.7.1 Historical Background

Considerable research has been done in the field of recognition system in which the primarily concern is to recognize both machine printed and handwritten Arabic numbers and characters.

Guyon in 1991 (Guyon, 1991), introduced an application of neural networks to character recognition, he made a recognition of English character.

Al-Badr and Haralick in 1995 (Al-Badr and Haralick.,1995) introduced an application to recognize Arabic character with segmentation-free word recognition .

An automatic recognition of hand-printed Arabic characters using artificial neural networks, was introduced by Amin in 1996. In his research on hand-printed Arabic characters (Amin et al., 1996).



Adel and Ali 1997 introduced another research that made an evaluation of neuro-fuzzy approach to recognize Arabic handwritten character (Adel and Ali, 1997).

In 1980 Nouh, Sultan and Tolba, presented in their research a recognition of Arabic character using sequential tree search and improved correlation (Nouh et. al.,1980)

In 1980, Amin and others presented a simple on line real time recognition system for handwritten Arabic characters, written on graphic tablets. They used a structural classification method to recognize the isolated Arabic character, they used the distance between features and vectors for unknown character and compared it with all characters in the directory and determined the character. The recognition rate is (85.4%) (Amin et al, 1980).

In 1994, Murad presented a system that recognizes handwritten Arabic characters, the research includes two different methods for recognition, which are template matching and projection profile ( Murad, 1994).

In 1985, Yacu implemented a system to recognize isolated handwritten Arabic character. He depend on his approach on a syntactic approach by extracting a geometric feature of a pattern and then classifying the pattern by register a special feature of every character on a table (directory) and comparing the new pattern with this table of features to classify it. The recognition rate was (87%) for testing (unknown) data (Yacu,1985).

In 1994, Andrew William in his Ph.D. thesis implemented off-line cursive handwritten recognition recurrent network. Results obtained showed that the method of current error backpropagation network can be applied successfully to the task. An (88 %) recognition rate has been achieved on open vocabulary task (Senior, 1994).

In 1990 Edelmanent developed a handwritten reader on the alignment of letter prototype. Features of digits as end points, top, left, or right are found. These features are tested against a set of prototype curves, (81%) recognition rate on the training set and around (50 %) on the test sets (Edelmanent et al., 1990).

In 1988, Fukushima and others, presented in their paper, a neural network for visual pattern recognition, he simulates a self-organizing network for classification the pattern. Fukushima et al. Presented in their paper a system has used band-extraction layer which can detect bend points and points of lines, correctly. They used techniques of dual threshold for feature extraction (Fukushima et al., 1998).

In 1998, Song proposed a structural adaptive intelligent neural tree for classification digits. They partition a hierarchically input pattern such as digits using tree structured network, which composed of sub-networks with topology preserving mapping ability (Song and Lee, 1998).

Cun et al. 1998 presented an application of backpropagation network to handwritten digit recognition. The inputs of the network consist of normalized images of isolated digits.

Cun showed in simple digit images that the architecture of the neural network strongly influences the networks generalized ability (Cun et al., 1998).

In 1996, Michael and Brijesh., described a method of recognition a handwritten digits by fitting generative models that are built from de formable B-Spline with Gaussian space along the length of the spline (Michael and Brijesh, 1996).

## 2.2 Artificial Neural Networks

In this section we introduce many concepts from the artificial neural network (ANN) which includes the historical development, the fundamentals concepts such as processing units, connections, computational procedure, and training procedure. Also the taxonomy of artificial neural network according to its learning procedure (supervised, unsupervised and reinforcement) has been discussed. Finally, we will describe Backpropagation, Backpropagation learning with momentum, and resilient propagation learning algorithm, with their derivation.

### 2.2.1 Introduction

Artificial neural network systems are simplified or simulated of the central nervous system of human. It models the brain as a continuous time nonlinear dynamic system in connectionist architectures that are expected to mimic brain mechanisms to simulate intelligent behavior.

Therefore, artificial neural network is an interconnected network of simple processing elements communication between processing elements occurs along paths of variable connection strengths, by changing the value of these connection strengths. The network can collectively produce complex overall behavior. So artificial neural network is a parallel-distributed information processing elements that are connected through directional links.

## 2.2.2 Historical Development of Artificial Neural Networks

Meculloh and pitts (1943) proposed the first computational model of neuron. They describe property of simple two-state binary threshold where the type of neuron in one whose output was either 0 or 1, depending on its processing of net input exceeded a given threshold.

Donlad Hebb (1949) was one of the first researchers who suggested a plausible process for neural and one of the first suggestions of the connection architecture.

The first computer simulation of artificial neural network was reported by Rochester and colleagues (1956) recognized by the official beginning date of Artificial Intelligence (AI) (Patterson, 1996) in conducting simulation of Hebb's model.

In (1959) ADALINE (ADApTive Linear NEuron) was conceived by Bernard widrow. It is a sing threshold logic neuron with bipolar output of  $-1$ , and  $+1$ , and he also developed a network of ADALINE called MADLINE (Multiple ADALINEs).

Rosenblatt (1962) developed several variations of the networks. He called them single-layer perceptions. He discovered an iterative learning procedure of this type and proved that the learning procedure always converged to a set of weights that produce the desired function.

In spite of Rosenblatt's work, Minsky and Papert (1969) showed that a single-layer perception's computing functions are quite limited in their computational power and they expressed multi-layer perceptions.

During the 1970s, a number of investigations were conducted on associative memory by researchers such as Kohonen, Anderson and others.

In 1983 Cohen and Grossberg developed an important theorem on the global convergence of dynamic networks.

Grossberg is best known for the highly successful adaptive resonance theory network (ART networks) which he invented.

In 1982 Hopfield suggested that the network can be analyzed in terms of an energy function, he showed that an energy function decreases and converges to a minimum and remains there.

Rumelhart (1986) discovered a learning algorithm to adjust the weights in multilayer feed forward networks, the algorithm is known as back propagation since the weights are adjusted from output layer backward to reduce the error.

Although artificial neural networks have had an interesting history, it is still one of the subjects that researchers focus on.

## 2.2.3 Fundamentals of Neural Networks

Artificial neural network is a composition of four major components, the first one is a set of processing units, the second is a set of connections, the third is a computation procedure, and the fourth is a training procedure. In the next subsections we will investigate those parts in more detailed.

### 2.2.3.1 Processing Units

Processing units are also known as Processing Elements (PEs) cells. Every artificial neural network is composed potentially of a large number of simple PEs simulating in neuron in the brain. These processing elements supporting massive parallelism and all computational work simultaneously on the system performed by these units. Functionally, there are three types of processing units (input, output and hidden units), these units compute a value that relates to arbitrary concepts.

- Input units: receive data from environment synchronous or asynchronous in time. Input processing elements are denoted by  $n$ -dimensional vector  $X$  which corresponds to a set of object feature values.
- Output units: represent control or decisions, producing  $m$ -dimensional vector  $Y$ .
- Hidden units: are connected to either input, output or other interior (hidden) units and compute their values and transform (passed) it to other further processing.

### 2.2.3.2 Connections

The units of the network are organized by a set of connection; these connections usually have associated weight, which is a real value that describes how much influence a unit has to its neighbor. Positive values of weights have an excitatory influence (one unit excite another) while negative values have inhibitory influence zero value corresponds to no connections. Weights are usually one directional links from input toward output units, and these connections determine the net topology for neural network. Weight encodes long-term memory or knowledge of the network and their value determines the computation reaction of the input pattern. Weights can change as a result of training, but change slowly because accumulated knowledge of the network changes slowly.

### 2.2.3.3 Computation Procedures

As the input pattern presented to the network, the neuron behaves as computation or activation function ( $f$ ) producing an output  $y = f(\text{net})$  where net is the weighted sum of input stimuli (Michie et al., 1994) this can be seen in Figure 2.6.

$$\text{net} = \sum_{i=1}^n w_i x_i + \theta \quad (2.7)$$

Where

$n$ : number of inputs

$x_i$  : associated external input signal or stimulus  $x_i$

$w_i$  : corresponding weight.

$\theta$  : Bias term, it acts as threshold value for the weighted sum.



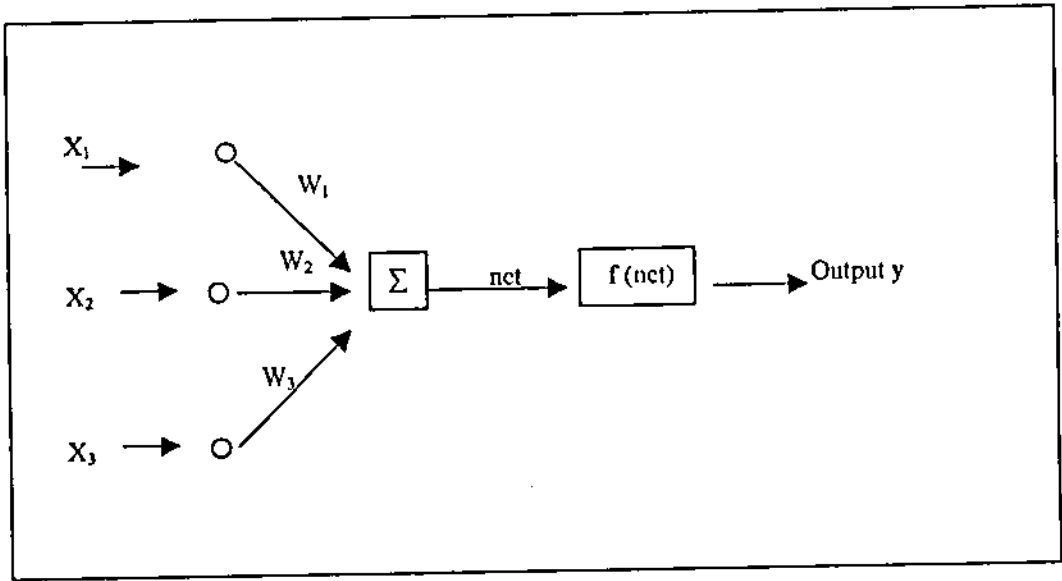


Figure 2.6 Simple Artificial Neural Network

Activation functions are bounded, differentiable and continuous, and the most popular activation functions are: linear, step, tanh (tan sigmoid) and sigmoid. Figure 2.7 shows these functions and the most commonly used is sigmoidal function (logistic function) (Patterson, 1996) (Tabliskis, 1995).

Sigmoidal function is continuous, differentiable and bonded between (0 and 1)

It's general form:

$$f(x) = \frac{1}{1 + e^{-B \cdot \text{net}}} \quad (2.8)$$

Where:

B: is a constant that determines the value of the shaped curve.

net: Weighted Sum of Input.

### 2.2.3.4 Training Procedures

In general, training means adapting the connections so that the network exhibits the desired computation behavior for all input patterns (Tabliskis, 1995)

Adaptive Learning is an essential property of a system that must function in dynamic environments or which compensate for variations in input stimuli (such as handwriting patterns generated by different people) (Patterson, 1996)

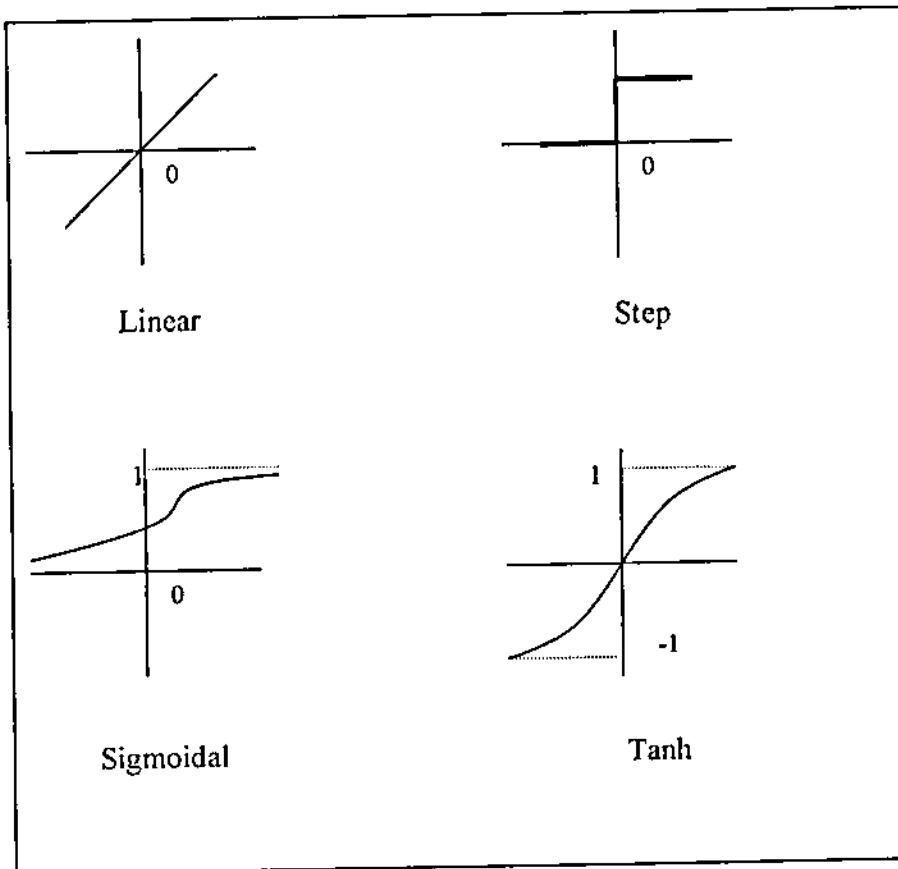


Figure 2.7 Typical Neural Network Activation Function

Learning is accomplished by the establishment of connections between nodes, after that adjustment the weight values the threshold values, or combinations of these operations in order to get the output that we satisfy.

The general learning problem is to find a weight matrix  $w$  that satisfies the vector equation (Patterson, 1996)

$$y^p = F(x^p, w) \quad (2.9)$$

$x^p$  : Input pattern,  $p= 1,2, \dots P$

$F$  : vector function.

## 2.2.4 Taxonomy of Neural Networks

Artificial neural networks can be classified in term of learning procedure to three main classes: Supervised, Unsupervised and reinforcement (Patterson, 1996)(Tabliskis, 1995)

### 2.2.4.1 Supervised Learning

In supervised Learning, a “teacher” provides output targets for each input pattern and corrects the network error explicitly. Learning process includes presentation of a “teacher” through process. During learning process comparison between computed output and target output is done to determine the error (E) according to the following formula.

$$E^p = y^p - t^p \quad (2.10)$$

Where E error

Y: Calculated output.

T: Target output.

P: P pairs of patterns in training set.

Error can be used to adjust the individual weight to reduce the error. Learning has been achieved when the error for training patterns has been reduced to some acceptable level for all new patterns.

#### 2.2.4.2 Unsupervised Learning

In unsupervised learning, there is no teacher to present the target outputs, and the network must find regularities in the training data by itself. Such self-organizing network can be used for clustering, quantizing, classifying or mapping input data. Such learning may be accomplished by strengthening selected weights to match central prototypical training pattern that are representative of a similar cluster (Patterson, 1996).

#### 2.2.4.3 Reinforcement Learning

In Reinforcement Learning (also called semi-supervised learning (Tabliskis, 1995), or graded (Jang et al., 1997), an external teacher doesn't provide explicit target for network but only evaluate the network's behavior (computed output) where it is right or wrong, and then use this information to improve the performance of the network. This

improved done by reinforcing weights on units, which give right answer and reduce the weight value on the units giving the wrong answer. This learning method can be considered as a simple way of adjusting behavior and can be found in animal learning skills and coping with physical environment, it also matches common sense ideas (Jang et al., 1997).

### 2.2.5 Backpropagation

Backpropagation (BP) also known as error backpropagation, the Generalized Delta Rule (MSC, 1995), or multi layer perceptron, is the most widely used supervised learning algorithm for NN. Bp is a very powerful tool, with applications of pattern recognition, dynamic modeling and control of systems overtime, among other tools. (WerBos, 1990) The backpropagation algorithm was first proposed by Paul Werbos in 1973. However, it wasn't known and become widely used until it was discovered by Rnnelhart and Mcclelland in 1986.

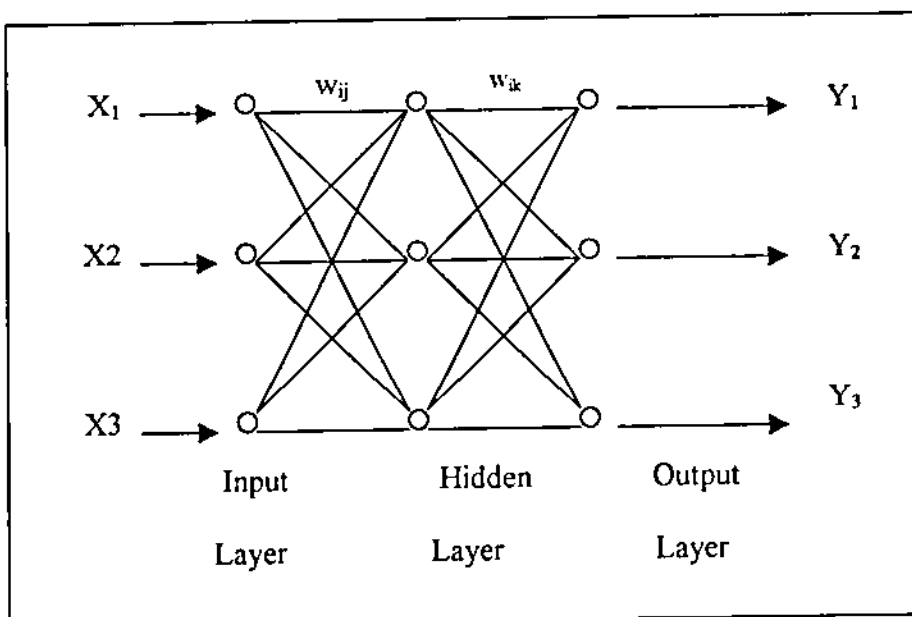


Figure 2.8 Backpropagation Network Architecture

Backpropagation partitions the network into layers, input, where inputs are distributed to the first hidden layer without computation, hidden layers, which make computational process and feed forward through network.

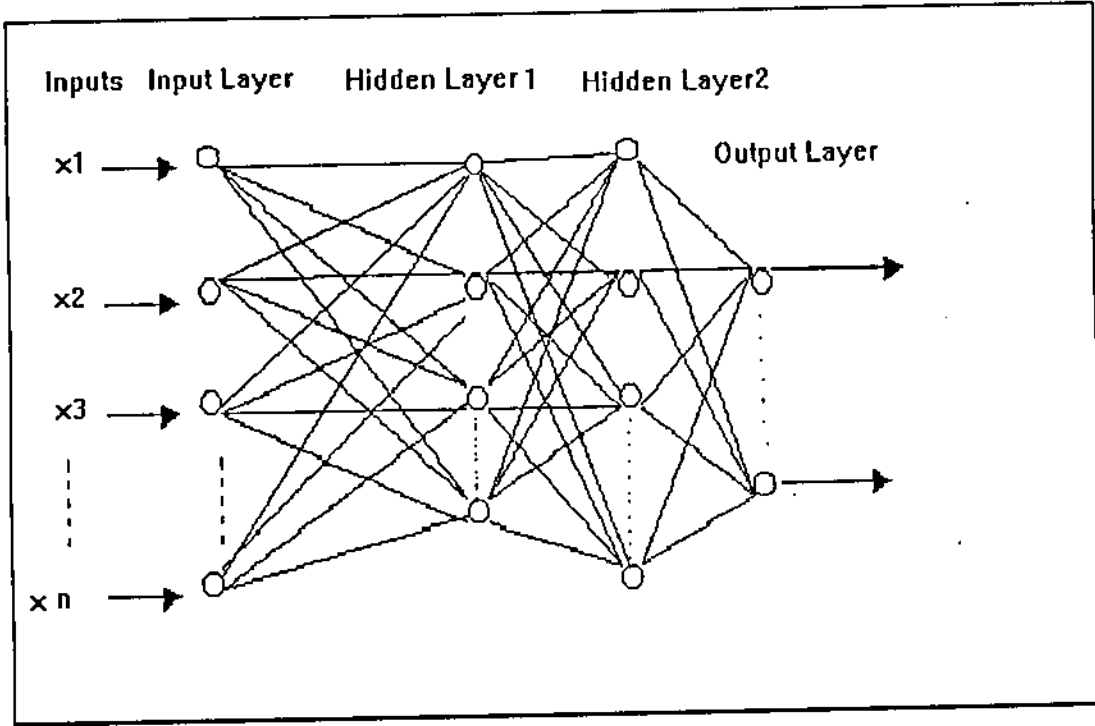


Figure 2.9 Architecture of Backpropagation

Output layer, calculates the result for a given input. Hidden layers and output layer receive weighted sum of outputs from previous layer as shown in equation (2.11), and compute its output using activation function (sigmoid function), for example as shown in equation (2.12)

$$net_{jk} = \sum out_{(j-1)i} w_{ik} + w_0 \quad (2.11)$$

Where:

$Out_{j-1}$ : is the input activation from unit  $i$ , from previous layer.

$W_{ik}$  : the weighted connecting unit  $i$  to the unit  $k$ .

$W_0$ : base.

$$f(net) = \frac{1}{1 + e^{-net}} \quad (2.12)$$

Learning in BP networks can be considered as an optimization procedure based on gradient descent that adjusts weights to reduce the system error. The training pattern is presented to the network and propagated forward layer by layer to the output layer where computed output is computed. The computed output is compared with a target output and error value is computed. An error value is back propagated through the network in order to adjust the weights. This learning process is repeated until the network responds for each input vector with an computed output vector is sufficiently closed to the desired one i.e. reach an acceptable error. In order to derive the backpropagation learning rule let us use Multi Layer Feed Forward (MLFF) network having a single hidden layer which has units that are fully connected with input layer and output layer units.

The input layer is consisting of  $n$  units, each unit fully connected with units in hidden layer which consist of  $h$  units: weigh connection between an input layer unit  $i$  and the hidden layer unit  $j$ .  $W_{ji}$  where  $i=1,2,\dots,n$  and  $j=1,2,\dots,h$ . Output layer consist of  $m$  units fully connected to hidden layers. Weight connecting hidden layer unit  $j$  with output unit layer  $k$   $w_{kj}$  where  $k=1,2,\dots,m$ . The input training pattern  $p$  is denoted as  $x^p$ ,  $p=1,2,\dots,p$ . The output from unit  $k$  in output layer to the input  $x^p$  is denoted by  $y^p_k$  and the target

$$H_j = \sum_{i=1}^n w_{ji} x_i$$

output is  $t^p_k$ . The net input to hidden layer unit  $j$  by using formula 3.4 is denoting by  $H_j$  where and the net input to the output layer unit  $k$  is denoted by  $I_k$

$$\text{Where } I_j = \sum_{j=1}^n w_{kj} z_j$$

where  $Z_k = f(H_j)$  using sigmoid function formula (2.12) The output  $y_k = f(I_k)$ . Figure 2.10 specifies this network, therefore,

$$\begin{aligned} y_k &= f(I_k) = \\ &= f\left(\sum_j w_{kj} \cdot z_j\right) \\ &= f\left(\sum_j w_{kj} \cdot f(H_j)\right) \\ &= f\left(\sum_j w_{kj} \cdot f\left(\sum_j w_{\mu j} x_i\right)\right) \end{aligned} \quad (2.13)$$

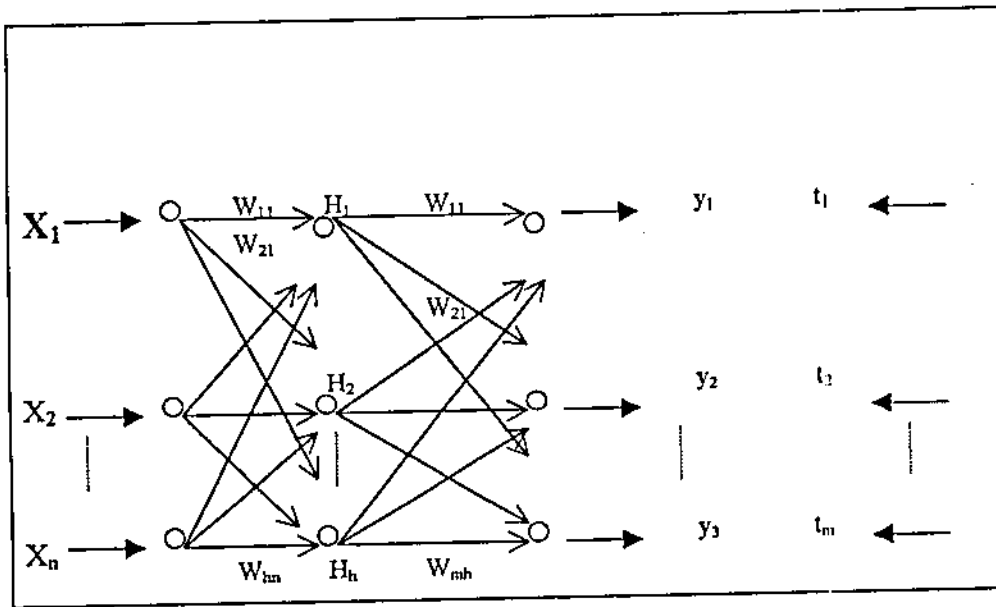


Figure 2.10 MLFF Network Connection and Variable.



System error  $E_{t,t}$  is the average of the output errors over all training pattern errors  $E^p$ , which defined by the mean square error and calculated by formula 2.14.

$$E_{t,t} = \lim_{p \rightarrow \infty} \frac{1}{p} \sum_{p=1}^p E^p \quad (2.14)$$

We will use a correction procedure that adjusts the weight in proportion to reduce the error. This can be accomplished if the weights are adjusted in proportion to the negative of the error gradient. Thus at step  $s+1$  of the training process, the weight adjustment should be proportion to the derivative of error mean compute in steps, this can be written as

$$\nabla W(s+1) = -\eta \frac{\partial E^p}{\partial w(s)} \quad (2.15)$$

Where  $\eta$  is a constant learning rate, scales the derivative, it has an important effect on the time needed until convergence is reached where too small value, too many steps are needed to reach acceptable solution. On the contrary a large learning rate will possibly lead to oscillation (Riedmler, 1994).

Hence the gradient of the total system error is given by

$$\frac{\partial E}{\partial W} = \frac{1}{p} \sum_{p=1}^p \frac{\partial E^p}{\partial W} \quad (2.16)$$

We will use the mean square error (MSE formula 2.17) to calculate the error function  $E^p$  since it is one of the most commonly used measures. It is differentiable, decreasing function of the difference between the computed outputs and target outputs.

$$E^p = \frac{1}{2} \sum_{k=1}^n (t_k^p - Y_k^p)^2 \quad (2.17)$$

We partially derive  $E^p$  with respect to  $W_{kj}$  and  $W_{ji}$  to find an expression for the weight adjustment (2.17) where:

$$W_{kj}(s+1) = W_{kj}(s) + \Delta W_{kj} \quad (2.18)$$

$$= -\eta \frac{\partial E}{\partial W_{kj}} \quad (2.19)$$

$$\frac{\partial E}{\partial W_{kj}} = \frac{\partial E}{\partial I_k} \cdot \frac{\partial I_k}{\partial W_{kj}} \quad (2.20)$$

Therefore: Chain rule is used to do so as follows:

$$= \frac{\partial E}{\partial I_k} \cdot \left( \frac{\partial}{\partial W_{kj}} \cdot \sum_j^n Z_j \cdot W_{kj} \right) \quad (2.21)$$

$$\frac{\partial E}{\partial W_{kj}} = \left( \frac{\partial E}{\partial Y_k} \cdot \frac{\partial Y_k}{\partial I_k} \right) \cdot \frac{\partial I_k}{\partial W_{kj}} \quad (2.22)$$

Using chaining rule again:

$$= -(t_k^p - y_k^p) \cdot f'(I_k) \cdot Z_j \quad (2.23)$$

$$\text{Let } \delta_k = (t_k - y_k^p) \cdot f'(I_k) \quad (2.24)$$

To adjust  $W_{ji}$  we don't have target value, instead we use output error to adjust the input to hidden layer weight  $W_{ji}$

Therefore

$$\Delta W_{kj} = \eta \cdot \delta_k \cdot z_j \quad (2.25)$$

$$\Delta W_{ji} = -\eta \frac{\partial E}{\partial W_{ji}} \quad (2.26)$$

Chain rule is performed to partially derive error function  $E$  according to  $W_{ji}$ .

$$\frac{\partial E}{\partial W_{ji}} = \frac{\partial E}{\partial H_j} \cdot \frac{\partial H_j}{\partial W_{ji}} \quad (2.27)$$

Chain rule again to partially derive  $E$  according to weighted sum from input  $I_j$ .

$$\frac{\partial E}{\partial W_{ji}} = \frac{\partial E}{\partial Z_j} \cdot \frac{\partial Z_j}{\partial H_j} \cdot \frac{\partial H_j}{\partial W_{ji}} \quad (2.28)$$

$$= -\sum_{k=1}^m (t_k - y_k) \cdot f'(I_k) \cdot W_{kj} \cdot f'(H_j) \cdot x_i \quad (2.29)$$

where  $\delta_k$  computed in (2.24)  $\delta_k = (t_k - y_k) \cdot f'(I_k)$

So

$$\frac{\partial E}{\partial W_{ji}} = \delta_j \cdot x_i \quad (2.30)$$

Therefore by using (2.26), (2.30) and (2.31)

where  $\delta_k$  computed in (2.24)

$$\delta_k = (t_k - y_k) \cdot f'(I_k) \quad (3.31)$$

$$\frac{\partial E}{\partial W_{ji}} = \delta_j \cdot x_i \quad (2.32)$$

$$\Delta W_{ji} = \eta \delta_j \cdot x_i$$

## 2.2.6 Backpropagation Learning with Momentum Term

Another possible way to improve the rate of convergence is by adding a momentum to gradient expression. This can be done by adding a fraction of the previous weight change to the current weight change. It acts as an averaging effect which smooths the trajectory of the gradient as it moves downhill (Patterson, 1996).

A commonly used update rule introduced by Rumelhart and others in 1986 includes such a momentum term. The update equation they used is defined by.

$$\Delta W_{ji}(t+1) = -\eta \frac{\partial E}{\partial W_{ji}(t)} + \mu \Delta W_{ji}(t). \quad (2.33)$$

Where  $\mu$ : is the momentum coefficient.

The value of  $\mu$  should be positive and less than 1, typical value in range (0.5, 0.9) (Patterson, 1996) but the optimal value of  $\mu$  is equally problem dependent.

Momentum is believed to render the learning procedure more stable and to accelerate convergence in regions of the error function (Riedmiller and Braun, 1993).

### 2.2.7 Resilient Propagation (RPROP)

RPROP stands for (Resilient PROPagation) and it's an efficient learning scheme that performs a direct adaptation of the weight step based on local gradient information. The effort of adaptation is blurred by gradient behavior whatsoever.

To achieve this, for each weight its individual updated value  $\Delta_{ij}$ , which solely determines the size of the weight-update.

This adaptive update-value evolve during the learning process based on its sign on the error function E, according to the following learning rule:

$$\Delta_{ij}(t) = \begin{cases} +\eta * \Delta_{ij}(t-1), & \text{if } (\delta E / \delta W_{ij}(t-1)) * (\delta E / \delta W_{ij}(t)) > 0 \\ -\eta * \Delta_{ij}(t-1), & \text{if } (\delta E / \delta W_{ij}(t-1)) * (\delta E / \delta W_{ij}(t)) < 0 \\ \Delta_{ij}(t-1), & \text{else} \end{cases} \quad (2.34)$$

Where  $0 < -\eta < 1 < +\eta$

Which means, every time the partial derivative of the corresponding weight  $W_{ij}$  change it's sign, which indicates that the last update was too big and the algorithm jumped over a local minimum, the update value  $\Delta_{ij}$  is decreased by the factor  $-\eta$ .

If the derivative return it's sign, the update-value is slightly increased in order to accelerate convergence in shallow regions.

Once the update value for each weight is adapted, the weight update itself follows a very simple rule:

If the derivative is positive (increasing error) the weight is decreased by its update-value.

If the derivative is negative (decreasing error) the update-value is added.

$$\Delta W_{ij} = \left\{ \begin{array}{l} -\Delta ij(t), \text{ if } (\partial E / \partial W_{ij}(t)) > 0 \\ +\Delta ij(t), \text{ if } (\partial E / \partial W_{ij}(t)) < 0 \\ 0, \text{ else} \end{array} \right\} \quad (2.35)$$

$$\Delta ij(t) = \left\{ \begin{array}{l} +\eta * \Delta ij(t-1), \text{ if } (\delta E / \delta W_{ij}(t-1)) * (\delta E / \delta W_{ij}(t)) > 0 \\ -\eta * \Delta ij(t-1), \text{ if } (\delta E / \delta W_{ij}(t-1)) * (\delta E / \delta W_{ij}(t)) < 0 \\ \Delta ij(t-1), \text{ else} \end{array} \right\} \quad (2.36)$$

Where  $0 < -\eta < 1, +\eta$

If the derivative is negative (decreasing error) the update-value is added.

$$\Delta ij(t) = \left\{ \begin{array}{l} -\Delta ij(t), \text{ if } (\delta E / \delta W_{ij}(t)) > 0 \\ +\Delta ij(t), \text{ if } (\delta E / \delta W_{ij}(t)) < 0 \\ 0, \text{ else} \end{array} \right\} \quad (2.37)$$

$$w_{ij}(t+1) = w_{ij}(t) - \Delta w_{ij}(t)$$

There is one exception: if the partial derivative change sign, i.e. the previous step was too large and minimum missed. The previous weight-update is reverted:

$$\Delta W_{ij}(t) = -\Delta ij(t-1), \text{ if } (\delta E / \delta W_{ij}(t-1)) \cdot (\delta E / \delta W_{ij}(t)) < 0 \quad (2.38)$$

## 2.3 Fuzzy Logic and Neuro Fuzzy Systems

This section consists of two main sections: Fuzzy logic and neuro fuzzy systems.

In fuzzy logic section, we will introduce the field of fuzzy logic, by introducing the fuzzy logic and the differences between fuzzy logic and crisp logic. Then we will explain the fuzzy set theory including the definition of the membership function and the different types of membership functions. Also we will discuss the set operation and the fuzzy rules including (linguistic variable and if-then-rule is). Finally the concepts of fuzzy reasoning and fuzzy system are given.

In the next section, we introduce a neuro fuzzy system by discussing the advantages of combining the neural network and fuzzy logic theory, then we present the neuro fuzzy classification. The component of neuro fuzzy system, neuro fuzzy learning and how training the neuro fuzzy system and adjusting fuzzy parameters value of neuro fuzzy system will be discussed next. Finally, neuro fuzzy model and adaptive neuro fuzzy inference rule have been given in this section.



### 2.3.1 Fuzzy Logic

Fuzzy logic is a superset of conventional (Boolean) logic that has been extended to handle the concept of partial truth, truth values between “completely true” and “completely false” (Jang et al., 1997).

Fuzzy logic is based on fuzzy set theory that was proposed in 1965 by Lotfi Zadeh, as an attempt to generalize classical set theory. Zadeh says that, rather than regarding fuzzy theory as a single theory, we should regard the process of “fuzzification” as methodology to generalize any specific theory from crisp (classical) discrete to a continuous fuzzy form. Also, fuzzy logic representation try to capture the way that human represent and reason real world situation, it links languages with computing (reasoning) through linguistic variables and quantifier, linguistic variable such as “tall”, “age”, can assume words such as “short”, “young”, which are quantifiable through membership functions.

#### 2.3.1.1 Fuzzy Set Theory

An ordinary set that divides the universe into those items that are completely in the set (member of a set) by assigning the value 1, and those that are completely outside the set (not member of the set) by assigning the value 0. On the other hand fuzzy set is a set without a crisp boundary that is gradially belonging to a set or not. In other words it allows the possibility of degree of membership, that is, any of the value greater than or equals to 0 and less than or equals 1. This value called membership value (usually

denoted by  $\mu$ ), and the function that assigns this value called membership function. Membership function (MF) maps the value from subset of the universe into the interval  $[0,1]$ .

For example, in a crisp set, if the height of person is greater than or equals 170 then he is tall, otherwise he is short. But if a women with a quarter or half centimeter less than 170, is she tall? In crisp logic the answer is not, on the other hand fuzzy logic allow the statement "she is a tall women" to have a range of truth fullness, depending on the height of person. This value is assigns to membership function. Figure 2.11 illustrates this example of fuzzy logic and traditional (crisp) logic.

The membership function gives fuzzy set flexibility in modeling commonly used the linguistic expression such as "water is hot", "temperature is high"... etc. We can define membership function as follows:

**Definition 2.1** Fuzzy set and membership:

Let  $X$  denotes universe of objects, then a fuzzy set  $A$  in  $X$  is defined as a set of ordered pairs:

$$A = \{ (x, \mu_A(x) \mid x \in X \} \quad (2.39)$$

Where  $0 \leq \mu_A(x) \leq 1$

$\mu_A(x)$  is called membership function for fuzzy set  $A$ .

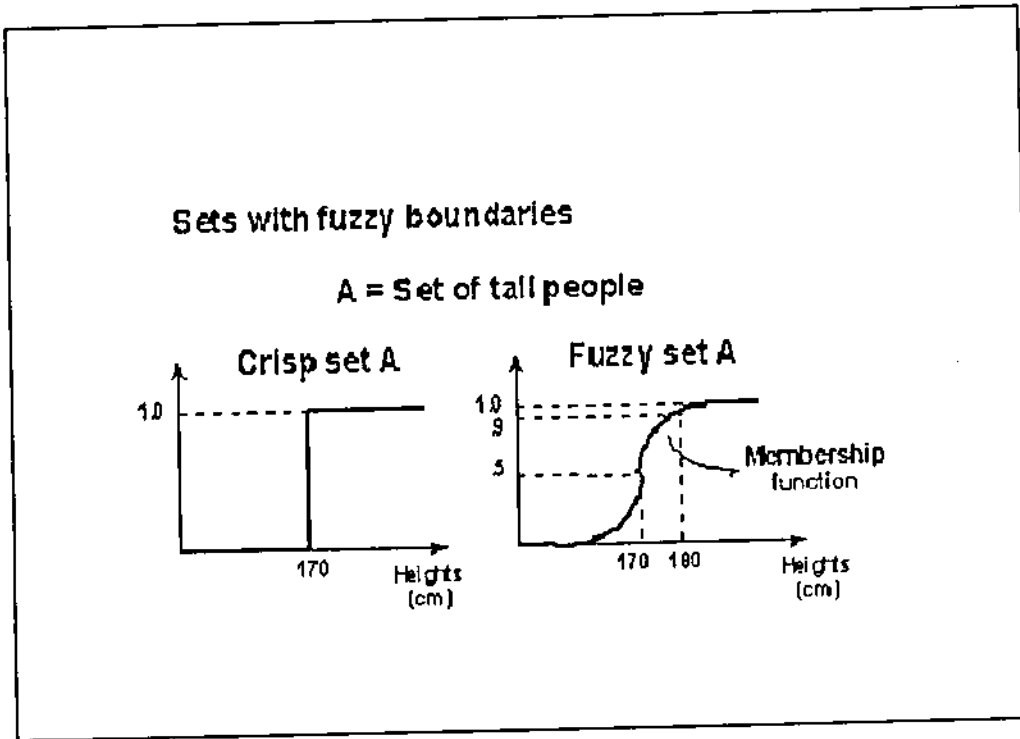


Figure 2.11 Fuzzy Set and Crisp Set

There are many types of membership function, where the most commonly used functions are:

- Triangular Membership Function:

Let  $a$ ,  $b$ , and  $c$ , be three parameters with  $a < b < c$ , then by using minimum (min), and maximum (max) operations a triangular membership function is given by the following formula:

$$\text{triangle}(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \quad (2.40)$$

Which is an equivalent to

$$\text{triangle}(x; a, b, c) = \begin{cases} 0, & x \leq a \\ \frac{x-a}{b-a}, & a \leq x \leq b \\ \frac{c-x}{c-b}, & b \leq x \leq c \\ 0, & \text{otherwise} \end{cases} \quad (2.41)$$

- Trapezoidal Membership Function

Let  $a, b, c,$  and  $d,$  be four parameters with  $a < b < c < d,$  then by using minimum (min), and maximum (max) operations a triangular membership function is given by the following formula:

$$\text{trapezoidal}(x; a, b, c, d) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \quad (2.42)$$

We have alternative expression for the preceding equation:

$$\text{trapezoidal}(x; a, b, c, d) = \begin{cases} 0, & x \leq a \\ \frac{x-a}{b-a}, & a \leq x \leq b \\ 1, & b \leq x \leq c \\ \frac{d-x}{d-c}, & c \leq x \leq d \\ 0, & \text{otherwise} \end{cases} \quad (2.43)$$

- Gaussian Membership Function

A gaussian membership function is specified by two parameters  $(a, b)$  as follows:

$$\text{gaussian}(x; a, b) = e^{-\frac{1}{2}\left(\frac{x-a}{b}\right)^2} \quad (2.44)$$

- Generalized Bell Membership Function

A generalized bell MF is specified by three parameters (a, b, c) as follows:

$$bell(x, a, b, c) = \frac{1}{1 + \left| \frac{x-c}{a} \right|^{2b}} \quad (2.45)$$

These membership functions can be seen in Figure 2.12

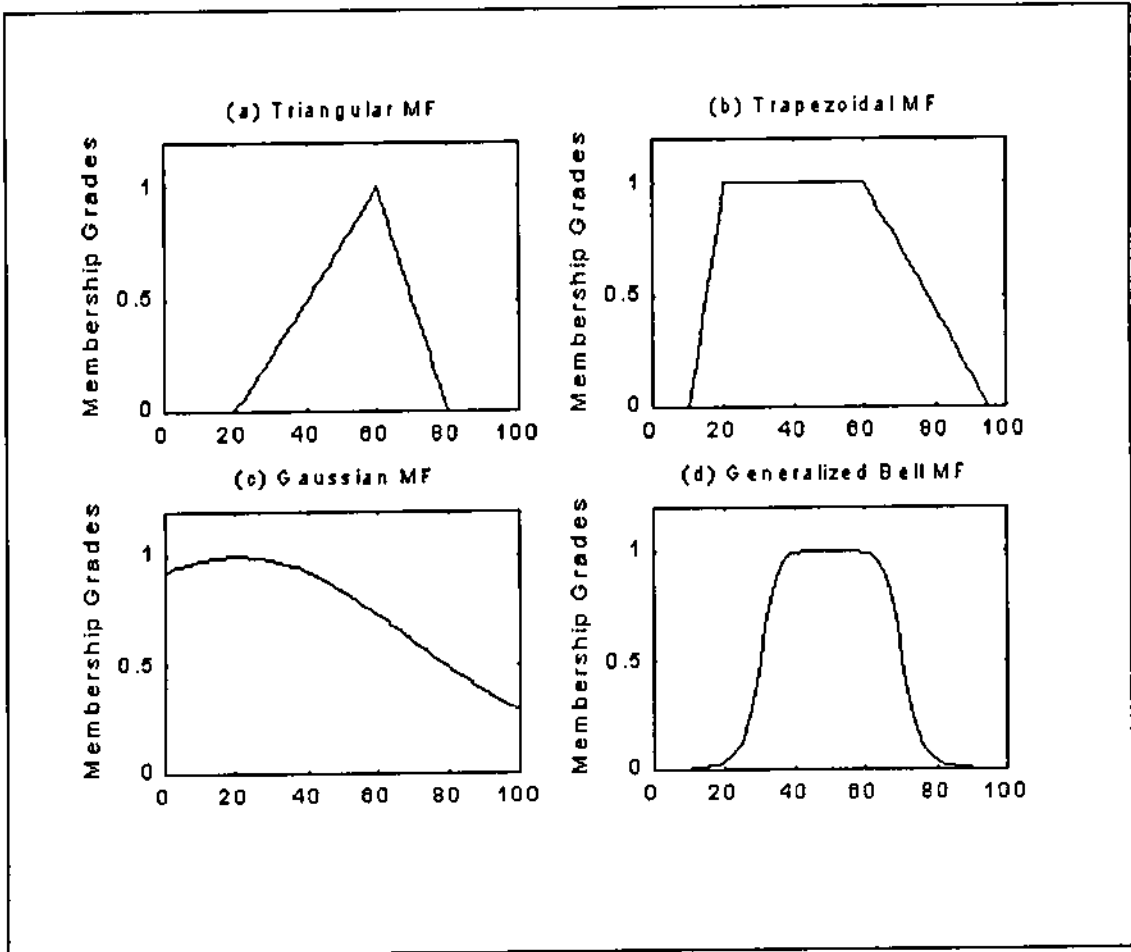


Figure 2.12 Illustration of Various Membership Functions

### 2.3.1.2 Fuzzy set Operations

There are several operations on fuzzy sets such as intersection, union, ... etc. These are somewhat similar to operations on crisp sets. The following are the most important operations:

#### 1. Containment

Fuzzy set A is contained in a fuzzy set B (or A is subset of B, or A is similar to B)

$$A \subseteq B \Leftrightarrow \mu_A(x) \leq \mu_B(x) \quad (2.46)$$

Where X denote the universe of objects, and  $\mu$  is a membership function for fuzzy sets A and B.

#### 2. Union (disjunction)

The union of two fuzzy sets A and B is a fuzzy set C, whose membership function is related to those of A and B by:

$$\mu_C(x) = \max(\mu_A(x), \mu_B(x)) \quad (2.47)$$

$$= (\mu_A(x) \vee \mu_B(x)) \quad (2.48)$$

#### 3. Intersection (Conjunction)

The intersection of two fuzzy sets A and B is a fuzzy set C, whose membership function is related to those of A and B by:

$$\mu_C(x) = \min(\mu_A(x), \mu_B(x)) \quad (2.49)$$

$$= (\mu_A(x) \wedge \mu_B(x)) \quad (2.50)$$

#### 4. Complement

The complement of fuzzy set  $A$  is denoted by  $\bar{A}$  ( $\neg A$ , not  $A$ ), and its membership function is defined as:

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x) \quad (2.51)$$

#### 5. Cartesian Product

Fuzzy sets  $A$  and  $B$  are fuzzy sets in  $X, Y$ , respectively. The Cartesian product of  $A$  and  $B$  is denoted by  $A \times B$  is a fuzzy set in the product space  $X \times Y$  with the membership function

$$\mu_{A \times B}(x) = \min(\mu_A(x), \mu_B(x)) \quad (2.52)$$

Similarly Cartesian co-product of  $A$  and  $B$ , which is denoted by  $A + B$ , is a fuzzy set in the product space  $X \times Y$  with the MF

$$\mu_{A+B}(x) = \max(\mu_A(x), \mu_B(x)) \quad (2.53)$$

### 2.3.1.3 Fuzzy Rules

Fuzzy rules combine two or more input fuzzy sets (called antecedent sets), and associate an output sets (called consequent sets) with them.

First, we are going to explain linguistic variable and then discuss the most important rule if-then-rule.

### 2.3.1.3.1 Linguistic Variable

Principle of linguistic was proposed by Zadeh, as an alternative to model human thinking. Conventional techniques for system analysis are unsuited for dealing with humanistic system, whose behavior is strongly influenced by human judgment, perception, and emotion (Jang et al., 1997). This what is called incompatibility.

As the complexity of a system increases, our ability to make precise and significant statements about its behavior diminishes until a threshold is reached beyond which precision and significant become almost mutually exclusive characteristics. (Zadeh,1988).

### 2.3.1.3.2 IF-THEN-RULES

Fuzzy if-then-rule is used to model and analyze the fuzzy system. The form of if-then-rule is:

$$R_n : \text{If } X \text{ is } A \text{ then } Y \text{ is } B$$

Where

$R_n$  is the rule number.

X is an input.

Y is an action.

A, and B are linguistic value defined by the fuzzy set.

“X is A” is called premise or antecedent, while “Y is B “ is called conclusion or consequence.



### 2.3.1.4 Fuzzy Reasoning

Fuzzy reasoning is an inference procedure that derives conclusions from a set of fuzzy if-then-rules and known facts. The basic inference rule is modus ponens, which can infer the truth of B (the form of if-then-rule) from the truth of A, and the implication  $A \rightarrow B$ .

### 2.3.1.5 Fuzzy Systems

A fuzzy system is a system which makes use of a fuzzy sets or fuzzy logic, and the corresponding mathematical framework. The basic fuzzy system fuzzy rule based has three major components: fuzzification, inference mechanism, and defuzzification.

#### 1. Fuzzification (Crisp to Fuzzy)

Convert input crisp variables into fuzzy set variables i.e.(into degrees of membership).

Functions like triangular, trapezoidal, sigmoidal, ...etc, that specified in fuzzy set theory can be used to convert crisp values into fuzzy sets.

#### 2. Inference (Rule evaluation)

Inference mechanism performs inference procedure that derives reasonable output or conclusion upon the rules and the given facts.

#### 3. Defuzzification (Fuzzy to Crisp)

In many applications, a crisp output is desired to obtain crisp value, the output fuzzy set must be defuzzified, which means, converts fuzzy actions into crisp actions, and combines them into a single executable action. That can be done by extract a crisp value from a fuzzy set.

There are five methods for defuzzifying a fuzzy set A of a universe of discourse X (Jang et al., 1997):

### 1. Centroid Of Area (COA) $X_{coa}$

It is the most widely strategy, which depends on the calculation of expected values of probability distributed (some times called the center of the gravity) and it is calculated by formula (2.54). This method computes the X-coordinate of the center of the area under the fuzzy set A.

$$X_{COA} = \frac{\int_x \mu_A(x) x dx}{\int_x \mu_A(x) dx} \quad (2.54)$$

Where:

$X_{coa}$ : X- coordinates of the center of area

$\mu_A(x)$  : a fuzzy set A .

A continuous domain X thus must be discreteized to be able to compute the center of area, this can be calculated by formula (2.55).

$$X_{COA} = \frac{\sum_{i=1}^n \mu_A(x_i) x_i}{\sum_{i=1}^n \mu_A(x_i)} \quad (2.55)$$

Where  $n$  is the number of element  $x_i$  in  $X$ .

Example 2.1: Consider the output fuzzy set  $X = (0.2, 0.2, 0.3, 0.9, 1)$ , and the output domain is  $Y = (0, 25, 50, 75, 100)$ . The defuzzified output obtained by applying formula (4.16) is:

$$\begin{aligned} Y' &= (0.2*0 + 0.2*25 + 0.3*50 + 0.9*75 + 1*100) / (0.2 + 0.2 + 0.3 + 0.9 + 1) \\ &= 72.12 \end{aligned}$$

The output of the system, computed by the fuzzy model, is thus 72.12. Which is a crisp value.

## 2. Bisector Of Area (BOA) $X_{BOA}$

That is the X-coordinate of the vertical line that partitions the region of the given fuzzy sets into two regions with the same area.

## 3. Mean Of Maximum

It is the average of maximizing  $X$  at which the membership function reaches the maximum  $\mu$

#### 4. Smallest Of Maximum

It is the minimum X-coordinate of the given maximizing membership function.

#### 5. Largest Of Maximum

It is the largest X-coordinate of the given maximizing membership function.

These components of a fuzzy inference system with crisp output are shown in Figure 2.13.

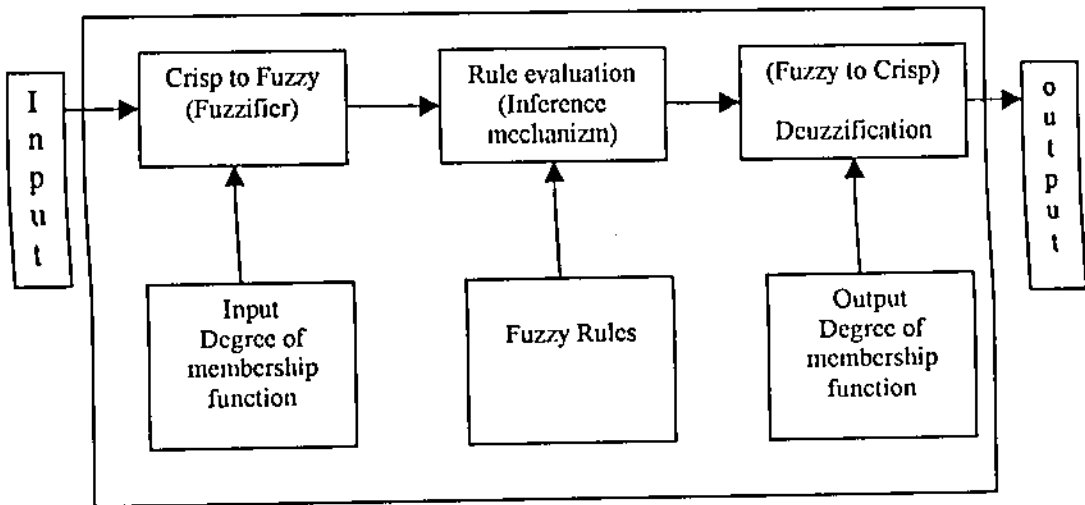


Figure 2.13 Fuzzy Rule Base System

## 2.3.2 Neuro Fuzzy Systems

In the conventional fuzzy approach, the model is designed according to a priori knowledge that fixes the membership functions and the consequent model. If this knowledge is not available but a set of input and output data is observed from the process, the component of the fuzzy system (membership and consequent models) can be represented in a parametric form and the parameters turned by learning procedure.

In this case, the fuzzy system turns into a neuro fuzzy system, which is a powerful tool with the trade off in terms of readability and efficiency between human – like representation of the model and learning method.

### 2.3.2.1 What is Neuro Fuzzy Systems?

A fuzzy logic provides a close link between natural language and “approximate computational reasoning”; fuzzy computing methods do not include the ability to learn adaptively (Patterson, 1996). Where as neural networks have many capabilities that needs for tasks such as, perception, learning, and predictive behavioral response.

Combining the capabilities of neural networks and fuzzy logic, into an integrated system will have the advantages of neural networks (such as learning ability, optimizing, connection structure, and the most prominent of advantage is their ability for massive parallel processing (Patterson, 1996)), and fuzzy abilities such as human like if-then rules thinking and the ease of incorporating expert knowledge (Meganti et al., 1998).

492631

Therefore, it is only natural that they can be used jointly and even combined in a type of Neuro-Fuzzy System. Zadeh called such a joint “soft computing”.

In summary, neural network and fuzzy logic can be applied concurrently within the same system to achieve complementary objectives to improve the performance of a system. They complement each other in the following ways:

1. Fuzzy logic can express qualitative values of human logic well, and provide smooth actions through continuous membership function.
2. Fuzzy logic rules can express a wide range of condition and action relationships thereby requiring fewer rules than conventional logic-based expert systems.
3. Neural networks are good for unstructured tasks such as pattern recognition like handwritten digit recognition, and they can determine membership relations.
4. Neural networks can learn to formulate complex nonlinear functions from training examples.
5. Neural networks can learn various tasks from training examples.

In general, a neuro fuzzy system is a fuzzy system that uses a learning algorithm derived from neural network computation to determine its parameters (fuzzy set and fuzzy rules) by processing data samples.

Therefor, a neuro-fuzzy system can be interpreted as a system of fuzzy rules. It is possible to create the system out of training data from scratch, and it is possible also to initialize systems by prior knowledge in form of fuzzy rules.

### 2.3.2.2 Neuro Fuzzy Classification

Classification of data is an area of application where a neuro-fuzzy can be used with rules like:

If  $X_1$  is  $m_1$  and  $X_2$  is  $m_2$  and . . .  $X_n$  is  $m_n$  Then  $P_i$  belongs to class $_j$  and

Where

$m_1, m_2, \dots, m_n$  : is membership function of fuzzy sets.

$P_i$  : Pattern consist of  $(x_1, x_1, \dots, x_n)$  features.

$i$  : 1,2,...,P number of patterns.

A neuro- fuzzy classifier has a different way of achieving the goal of classification i.e. to classify the input pattern into its class. If a decision is made for a neuro-fuzzy classifier usually the following advantages are considered:

- Prior knowledge can be used.
- The classifier is interpretable in form of linguistic rule.

These advantages are discussed in more details in sections 2.3.2 and 2.3.2.1.

A neuro fuzzy classifier usually derived from data and is not specified directly, it can be created from data by heuristic learning procedure.

### 2.3.2.3 Components of Neuro Fuzzy Systems

The neuro fuzzy system has five layers of neuro with a selected feedforward inter connection, as illustrate in Figure 2.14. Each node performs a particular function (node function) on incoming signals. The formulas for the node functions may vary from node to node.

The first layer (A and B nodes) represents input linguistic variable values, which corresponds to conjuncts in the antecedents of fuzzy rules. Each node is connected to a few units in the first hidden layer.

The second layer: antecedent fuzzy sets nodes divided into subgroups, connected to the second hidden layer units.

The third layer: fuzzy rule units in which each input from the antecedent fuzzy sets is connected to a rule to conjunct in the rules antecedent and produce the consequent of the rule.

The fourth layer: every unit takes the output (consequent) of the third layer from the units which are in fuzzy domain group, that connected with and make its computation.

The result is passes to final layer.

The fifth layer is the conclusion, the output fuzzy value.

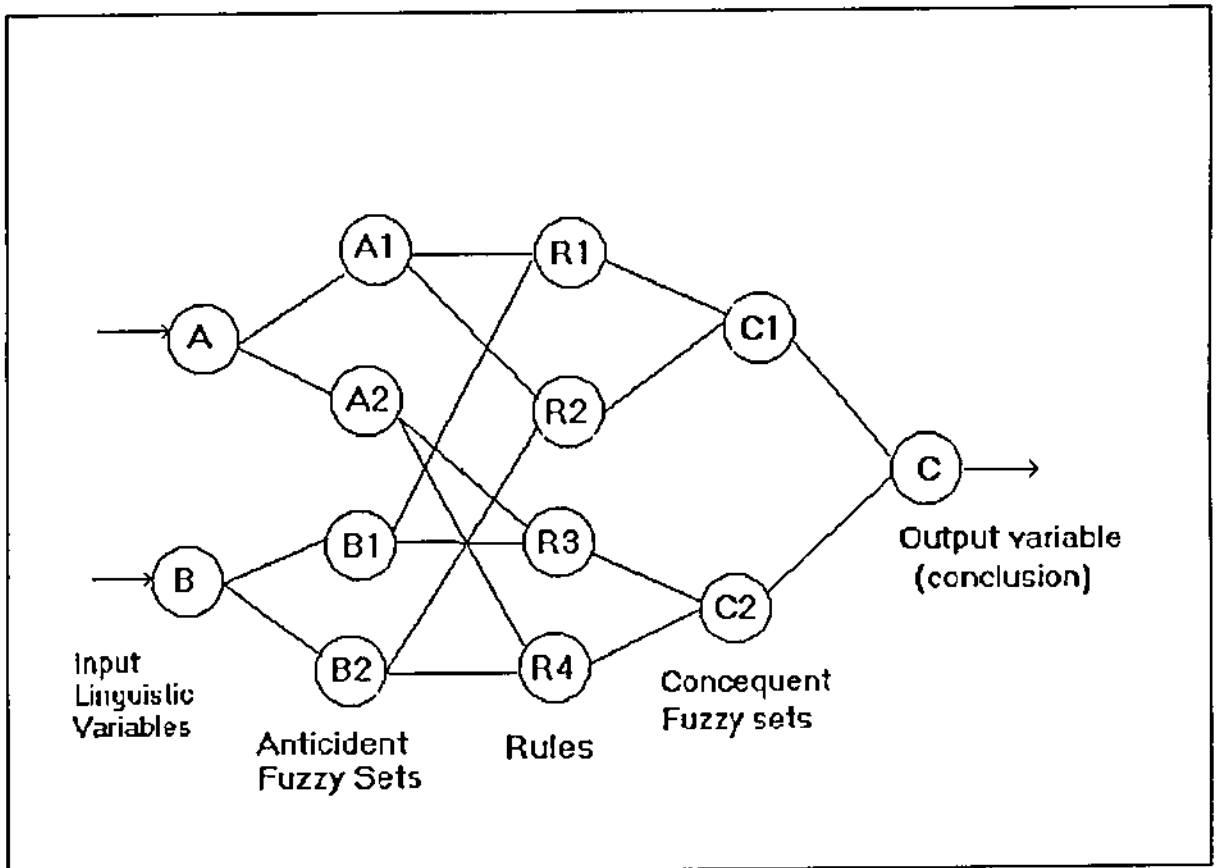


Figure 2.14 Neuro Fuzzy Network



The four fuzzy rules in the Figure 4.14 can be interpreted as follows:

$R_1$ : if A is  $A_1$  and B is  $B_1$  then C is  $C_1$ .

$R_2$ : if A is  $A_1$  and B is  $B_2$  then C is  $C_1$ .

$R_3$ : if A is  $A_2$  and B is  $B_1$  then C is  $C_2$ .

$R_4$ : if A is  $A_2$  and B is  $B_2$  then C is  $C_2$ .

The normalized inputs to the network are passed to the different antecedent set units through the input to antecedent unit connections. The outputs of the antecedent units are the corresponding membership function values for the input fuzzy sets.

Each rule unit has weight values equal  $1/k$ , where  $k$  is the number of input values to the rule. Minimum operation or dot product operation for example can be performed with these input weights. Rule weights are applied to complete rules: The antecedent of the rule is evaluated to determine the degree of the fulfillment, which is then multiplied by rule based. Rule weights are applied only to the consequent part of rule, the degree of the fulfillment of a rule is used to compute the conclusion which is then multiplied by the rule weights. Using rule weights gives rise to some semantically problem, they are sometimes interpreted as measure of "important", "influence", or "reliability".

Likewise, each consequent set unit has weight values set equal to one, using the activation function defined in formula (2.56), each consequent set results in a computation of the max operation for example.

$$F(x) = \begin{cases} 0, & \text{for } x \leq 1 - 1/n \\ n(x + 1/n - 1), & \text{for } (1 - 1/n) < x < 1 \\ 1, & \text{for } x \geq 1 \end{cases} \quad (2.56)$$

The output units compute the final defuzzified result in accordance with the formula (2.57), which is centeroid method for defuzzification.

$$\text{output} = \frac{\sum_{i=1}^n \mu(x_i) x_i}{\sum_{i=1}^n \mu(x_i)} \quad (2.57)$$

Where  $\mu(x)$  is the degree of the fulfillment of the  $i$ 'th rule.

In training the network a form of gradient descent error backpropagation is used to update the two layers of weights.

#### 2.3.2.4 Neuro Fuzzy Learning

To use the advantages of neural network and fuzzy logic which are discussed in sections 2.3.2 and 2.3.2.1 we have to develop a fuzzy learning algorithm. The Backpropagation algorithm provides a method for adjusting crisp values of the weights of a neural network, which minimizes a value provided by an error function which computes the difference between the actual and the desired output of the network. In order to compute the new fuzziness of the weights we propagate the desired fuzziness of the output data directly to the preceding layer instead of using an error signal. So backward pass is divided into three parts:

First the desired fuzziness is passed back through a neuron. Then we have to distribute the fuzziness among the input connections. We have to adjust the new fuzziness of the

weights and to compute the desired output of the previous neuron. Finally different desired output values have to be combined.

In the following sections we will talk about training of the neuro-fuzzy model value and the adjusting of the fuzziness.

#### 2.3.2.4.1 Training of Neuro Fuzzy Systems

First we will use the most popular membership function, the triangular function which can be specified by three parameters.

$$a = \text{triangle}(a_l, a_m, a_r) \quad (2.58)$$

Where

$a$  : model value.

$a_l, a_m, a_r$  : nonnegative real values represent (left, middle, right) fuzziness respectively.

We do not use the difference between system's values of the desired and the network output, but we compute the fuzzy difference between both fuzzy numbers for adjusting values of the weights. This fuzzy difference is a fuzzy number too, but for adjusting the system values of the weights a real number is needed. We interpret triangular fuzzy number as a probabilistic object. This way of seeing fuzzy numbers is not based on fuzzy set theory, but we can add a process of Backpropagation algorithm, which can be profitable for the learning process.

The algorithm uses this meaning of fuzzy numbers and guesses possible values for the left and the right side of a fuzzy weight. This is done corresponding to the probabilistic distribution of the fuzzy the fuzzy number. The obtained values are used to calculate

the weight changing by common Backpropagation algorithm (specified in section 2.2) and to apply it to the modal value of the weight.

#### 2.3.2.4.2 Adjusting Fuzzy Parameters Value

An error function can be used for adjusting the fuzziness of the weights. To see how the fuzziness can be trained we look at the information we already have. The forward pass delivers an output vector, comparing it with the desired output, the fuzziness is either smaller or larger as fuzziness of the required output.

In the first case (fuzziness is smaller) the weights that influence the appropriate output neuron have to increase, in the second case (fuzziness is larger) they have to decrease. To reach this goal we distribute the fuzziness error of the left and right side within the backward pass too. Within an output triangle ( $O_l, O_m, O_r$ ) of an output neuron and the corresponding desired output triangle ( $T_l, T_m, T_o$ ) the fuzziness error is given by:

$$E_t = O_l - T_l \quad (2.59)$$

$$E_r = O_r - T_r \quad (2.60)$$

The fuzziness error is passed back through the neurons. A neuron located in a hidden layer has stronger influence than any output neuron. Therefore it receives several different error. Because a neuron produces only one output, we have to compute the

average of the errors. We used the arithmetic average to minimize the computational expense.

If the fuzziness error is positive, the fuzziness of the desired output is too small in the average. So fuzziness of this weight has to be increased. So we can formulate the following learning rules for the left and the right fuzziness of the weights.

$$W_R^{new} = W_R^{old} + E_R * \eta \quad (2.61)$$

$$W_l^{new} = W_l^{old} + E_l * \eta \quad (2.62)$$

Where:

$\eta$  : is some positive step size regulating to what extent proceeding that direction, and called learning rate of the fuzziness.

(2.63)

$$W_c^{new} = \frac{W_r + W_l}{2}$$

During learning for each output unit  $c$ , we determine the delta value

$$\delta_c = t_c - o_c \quad (2.64)$$

For each rule unit  $R$  we first determine the delta value

$$\delta_R = O_R(1 - O_R) \sum W(R, C_R) \delta_e \quad (2.65)$$

Where

$W(R, C_R)$  is the fuzzy weight on the connection from rule unit R to output unit  $C_i$

### 2.3.2.5 Neuro Fuzzy Modeling

We will propose a class of Adaptive Neuro Fuzzy Inference System (ANFIS), which acts functionally as inference system.

ANFIS architecture:

For simplicity, we assume that the fuzzy inference system has two inputs x and y and one output z. A common rules for the first-order Sugeno fuzzy model is as follows:

Rule 1: If x is A1 and y is B1 then  $f_1 = p_1x + q_1y + r_1$

Rule 2: If x is A2 and y is B2 then  $f_2 = p_2x + q_2y + r_2$

We can see reasoning mechanism for this model in Figure 2.15.

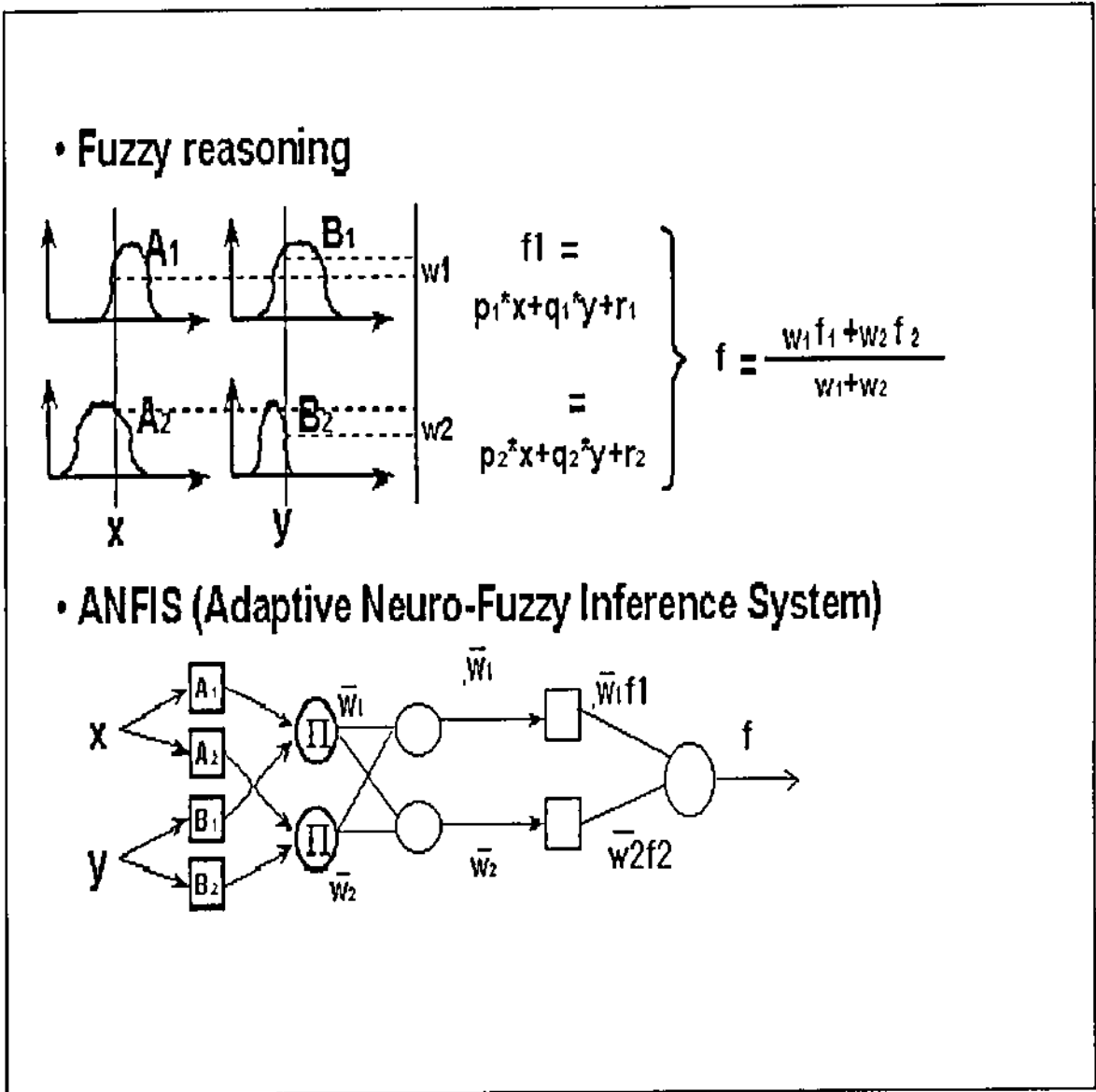


Figure 2.15 ANFIS Architecture for First-Order Sugeno Fuzzy Model

ANFIS architecture for first-order Sugeno fuzzy model consists of five layers.

Layer 1: every node<sub>i</sub> in this layer is an adaptive node with a node function

$$O_{1,i} = \mu_{A_i}(x) \quad , \text{for } i = 1, 2 \quad (2.66)$$

$$O_{1,i} = \mu_{B_{i-2}}(y) \quad , \text{for } i = 3, 4 \quad (2.67)$$

Layer 2:

Every node in this layer produces an output which is the product of the incoming signal. We can use (AND), (minimum) operations instead of dot product operation.

$$O_{2,i} = w_i = \mu_{A_i}(x) \cdot \mu_{B_i}(y), \quad i=1,2. \quad (2.68)$$

Layer 3:

Every node in this layer calculate the ratio of the i'th rule strength (weight) to the sum of all rule strengths.

$$O_{3,i} = \varpi_i = \frac{w_i}{w_1 + w_2}, \quad i=1,2 \quad (2.69)$$

Layer 4:

Every node in this layer is an adaptive node with a node function

$$\begin{aligned} O_{4,i} &= \varpi_i f_i \\ &= \varpi_i (p_i x + q_i y + r_i) \end{aligned} \quad (2.70)$$

Parameters in this layer (p, q, r) are referred to as consequent parameters.

Layer 5:

The node in this layer computes the overall output as summation of all incoming signals.



## Chapter 3

# Handwritten Arabic Numbers Recognition Using Artificial Neural Networks

## Chapter 3

# Handwritten Arabic Numbers Recognition Using Artificial Neural Networks

### 3.1 Introduction

The area of the handwritten Arabic numbers recognition received much attention over the past decade. Indeed, because of its importance, it has become a benchmark problem for different approaches. This effort is certainly justified, as there are numerous industrial and business applications where automated recognition will result in substantial cost savings.

Therefore, this chapter and chapter 4 will introduce systems designed to recognize handwritten Arabic numbers. The first system based on artificial neural networks that used for recognition. The second system based on neuro fuzzy systems. Before recognition process, many preprocessing operations have been developed and designed. These preprocessing operations such as: Segmentation, binarization, normalization and location operations prepare the input image for recognition systems. These recognition steps can be seen in Figure 3.1.

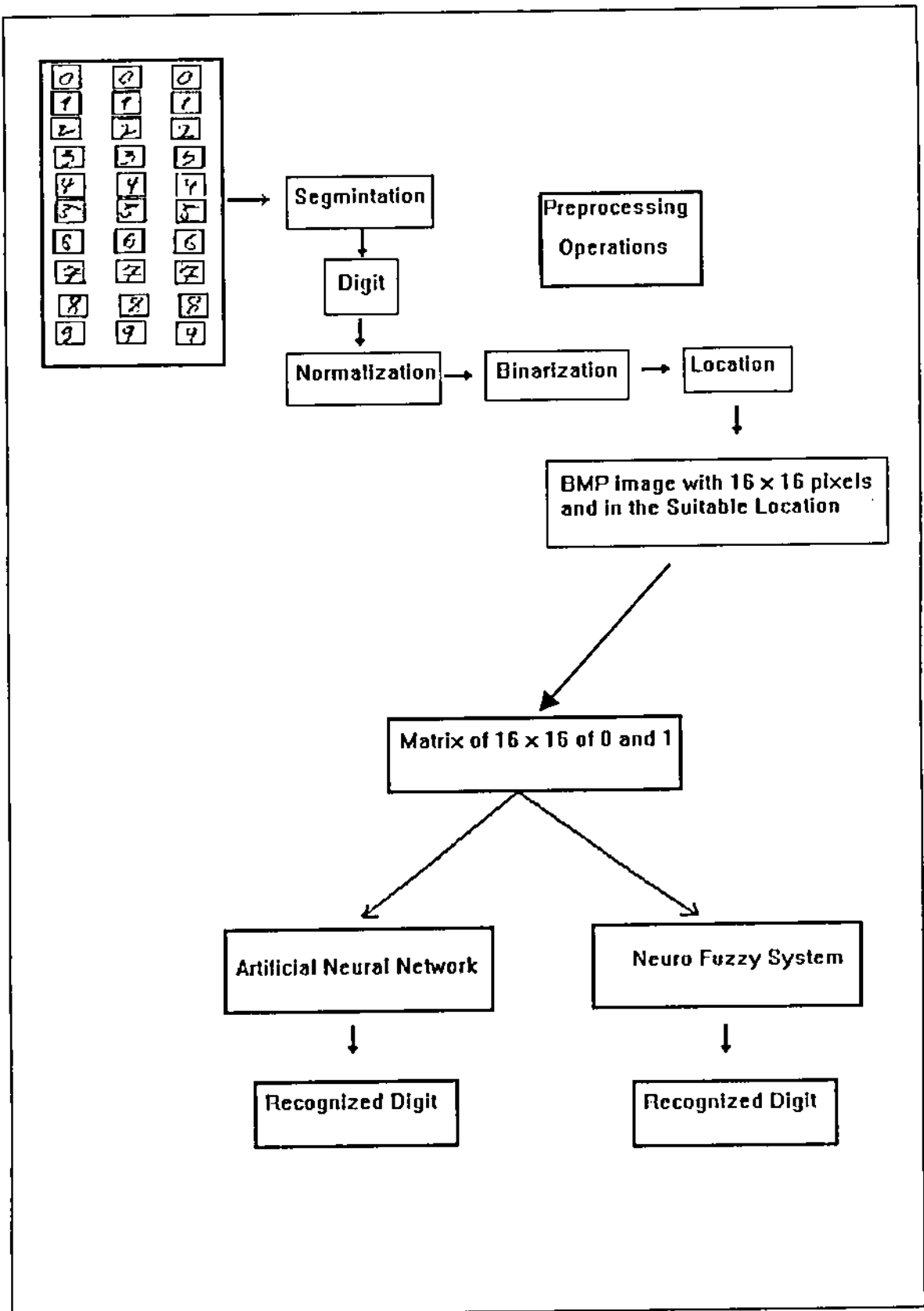


Figure 3.1 Steps of Handwritten Arabic Numbers Recognition Systems

## 3.2 Preprocessing

A preprocessing is operations applied before pattern classification or identification is presented. Producing from the input images a representation form that facilitates the recognition process.

The preprocessing is necessary for classification and must satisfy two basic requirements: The first requirement is to locate each digit on a file in order to make sense in the coded file which will eventually result from recognition. The second requirement presents each isolated digit to the recognition algorithm in a suitable form. Segmentation, normalization, binarization and location simplify the preprocessing operations.

### 3.2.1 Segmentation Operation

Segmentation means, isolating and extracting each digit (0,1,2,...,9) from numbers.

In order to recognize a handwritten number, the image of the number or the digit needs to be segmented into images of individual digit. There are two major approaches of segmentation:

1. Have the writers or the peoples who fill the form which is used for recognition to write isolated digits into a preprinted boxes, and extract them based on a geometric forms model i.e. spatial location of the boxes. Such approach is easy to implement, and fast. However, printed boxes require more space on forms. This approach will be used in our work.

2. Unconstraint fields consist of a single blank space for a complete response. Such approach requires more computation and in inherently more unresolvable ambiguities.

Both approaches have their place in practice, printed digits boxes are best reserved for responses that are difficult to verify, examples are bank check amounts, social security numbers and, telephone numbers.

After facilitating the segmentation operation we get files each containing an isolated digit.

### 3.2.2 Normalization Operation

Some normalization is needed to standerize the size of the digits. The normalized image size is usually taken to be 16x16 pixels. Because the digits can be written in many different sizes, we have to force the image of the digit to be located in a fixed size window without changing the meaning of the digit. In our work we use 16x16 pixel images. This size has been used after making many experiments on different sizes as 25x25 pixels, 10 x 8 pixels. The image size of 16 x 16 pixels is the most suitable size for our work. Also it is recommended by many researcher in this field see (Claus, 1998), and (Patterson, 1996).

### 3.2.3 Binarization Operation

The image of each digit should be binarized, which means that individual pixel are set to either zero or one. As the image of the segmented digit was saved in a monochrome bitmap file this means that each element of the image will be 0 or 1.

### 3.2.4 Location operation

The digit has to be located in a fixed location in its file. The Suitable place in our case is to put the digit in the upper and left most corner boundary of the image. This can be done by, finding the first upper row that has pixel or pixels that belong to the digit and make it the first row of the image. Determine the most left column that has a pixel or pixels belong to the digit and shift the digit to the left most column of the image.

These two steps were done without changing the shape or meaning of the digit.

We can see an example of presenting digit 0 in Figure 3.2, where in (a) digit zero in the location of the image as it taken after segmentation operation. In (b) the location of digit zero after processing location operation.

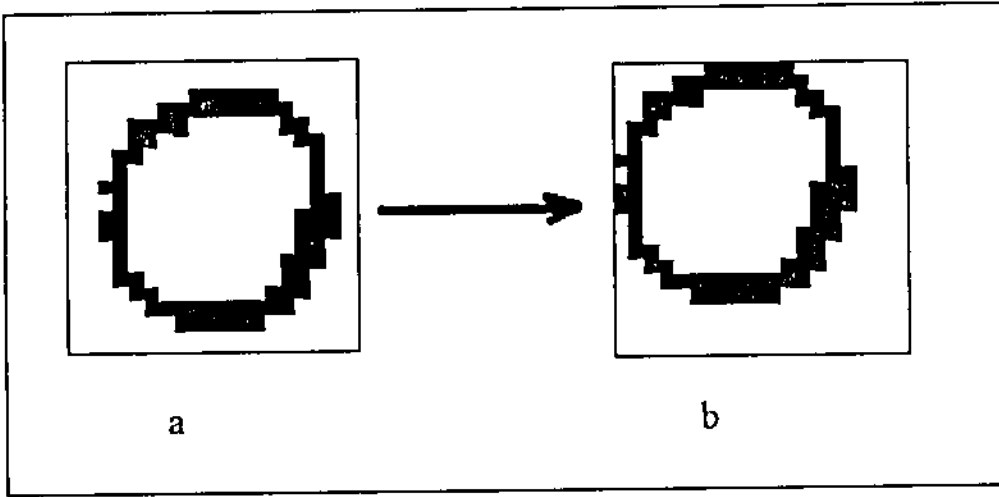


Figure 3.2 Digit 0 in its Location

### 3.3 Recognition of Handwritten Arabic Numbers Using Artificial Neural Networks

#### 3.3.1 Data Sets

The experiments are performed using data sets of 2000 samples, 200 samples for each digit. These digits were extracted from images that obtained using a scanner. The binary image of each number was written to monochrome bitmap file. The conversion of the pattern of black and white area constitutes an entire file into the set of smaller files. Each file has a fixed size of 16x16 pixels. And serve as the input to digit classification or recognition algorithm. These small files are needed for recognition to locate each digit on an isolated file. Presenting each isolated digit to the recognition algorithm in a

suitable form. Which means separating each isolated digit in a monochrome Windows Bitmap file of size 16 x 16 pixels.

A handwritten number database samples are collected from 220 persons. These samples are acquired from students, faculty members, teachers, employees of many universities, school students, and teachers from different schools. Their handwriting was sampled on A4 size paper. These samples were scanned using a HP scanner to produce scanned images. These scanned images were saved in a Window Bitmap format. Sample of collected digits can be seen in Figure 3.3, which display a randomly chosen digits from the collected samples.

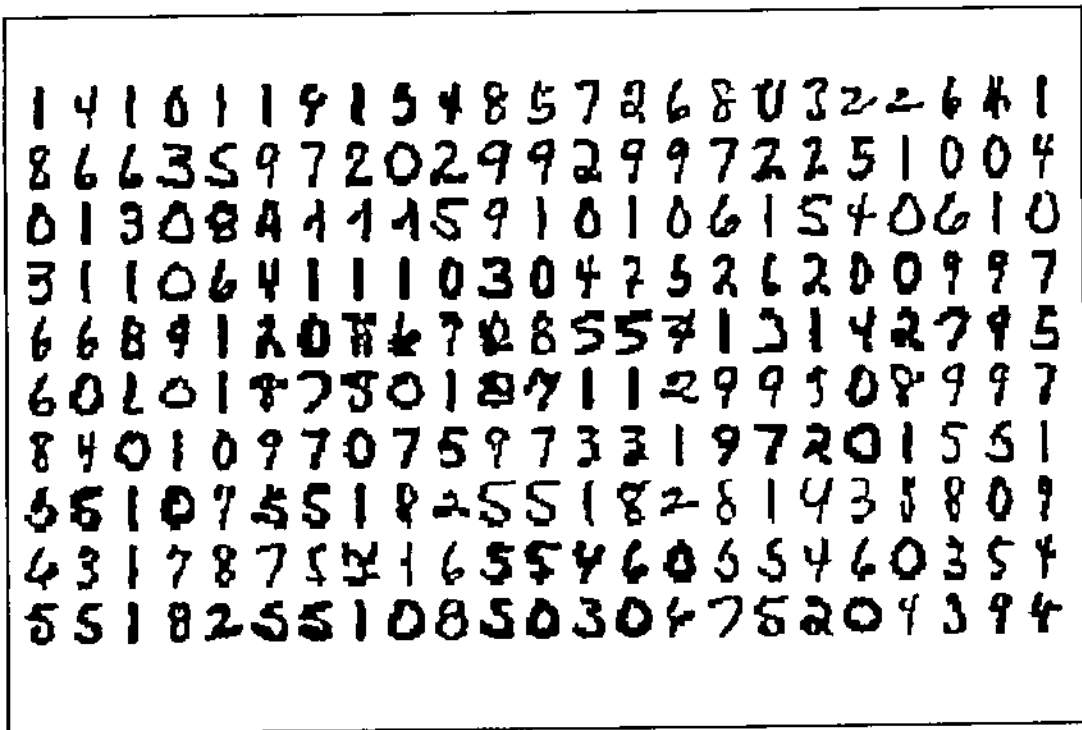


Figure 3.3 Samples of Handwritten Digits



After the preprocessing operations, We get Windows Bitmap files of size  $16 \times 16$  pixels; each file contains image of isolated digit in its suitable location.

In order to use these files of images we have to transfer the image into a two-dimension matrix of 16 rows and 16 columns. This matrix is then transferred to a vector of size ( $16 \times 16 = 256$ ) that can serve as an input to a neural network.

### 3.3.1.1 Testing and Training Sets

In our experiments we have 2000 samples, 200 samples for each digit. These samples are divided into two groups or sets: training and testing set.

#### 1. Training set:

The training set is the set that has the samples used for training the neural network. The training set represents 90% of the whole samples, i.e. every digit has 180 samples for training. Every sample has its target used for training, which is the correct or real value of the sample.

This training set used to train the network by providing pairs of sample and its correct target.

#### 2. Testing set.

The testing set is the set that has the samples used to test the neural network. The testing set includes 10% of the whole samples, i.e. every digit has 20 samples for testing and the neural network does not see these samples through training. Each sample has its

target, which is used to check whether the neural network can correctly recognize the samples or not.

The training and testing sets are changing through every experiment. That means, in every experiment the samples that representing the training set, and the testing set are changing. At the first experiment we take the first 10 % of the samples to be the testing set and the remainder 90% to be the training set. At the second we take the next 10 % of the samples to be the testing set and the other 90% to be the training set. Therefore, 10 experiments are needed to cover all samples i. e. to make all samples be in training set and in one experiment to be in the testing set.

### 3.3.2 The Neural Networks Architecture

In our experiments, we use multi layer feed forward neural network, with Resilient Propagation (RPROP) learning algorithm. This neural network consists of 4 layers.

The first layer, is the input layer, consists of 256 nodes, which take their input values from the training or testing set. Every node is fully connected and feed forward with all nodes in the next layer.

The second layer (first hidden layer) is consisting of 100 nodes. Every node finds its output value by calculating its input weighted sum and substitutes the result in the activation function that used. This output acts as the input to the next layer nodes.

Every node is fully connected and to all nodes in the next layer. The weighted sum is calculated by formula 3.1.

$$net_j = \sum_i w_{ji} x_i + \theta \quad (3.1)$$

Where

$i = 1, 2, \dots, n$  number of inputs

$j$ : node number

$x_{ji}$  : external input.

$w_i$  : corresponding weight.

$\theta$  : Bias term, it acts as a threshold value for the weighted sum.

The third layer (second hidden layer) consists of 10 nodes. Every node takes its input value from the output of the previous layer with its weight, then calculating weighted sum, and then substitute this weighted sum with sigmoidal function to produce its output. This output represents the input value to the final layer.

The fourth layer is the output layer which consists of 10 nodes. The 10 nodes represent the outputs. Every node calculates its output value from the nodes in previous hidden layers.

As we see 10 outputs are calculated. To determine the suitable output we take the maximum calculated output of all nodes. The node that produces the determined output represents the class that the input digit belongs to. That means, if node 0 produces the maximum output between all 10 output nodes (node1, node2, ..., node10) the input pattern (digits) is zero.

The architecture of the neural networks shown in Figure 3.4, where every layer with its node is displayed.

According to the activation function that used to produce the output of every node, we use a Sigmoidal function. The Sigmoidal function is continuous, differentiable and bonded between 0 and 1. It's general form is given in formula 3.2.

$$f(x) = \frac{1}{1 + e^{-B \cdot \text{net}}} \quad (3.2)$$

Where:

B: a constant that determines the value of the shaped curve.

net: Weighted Sum of Input.

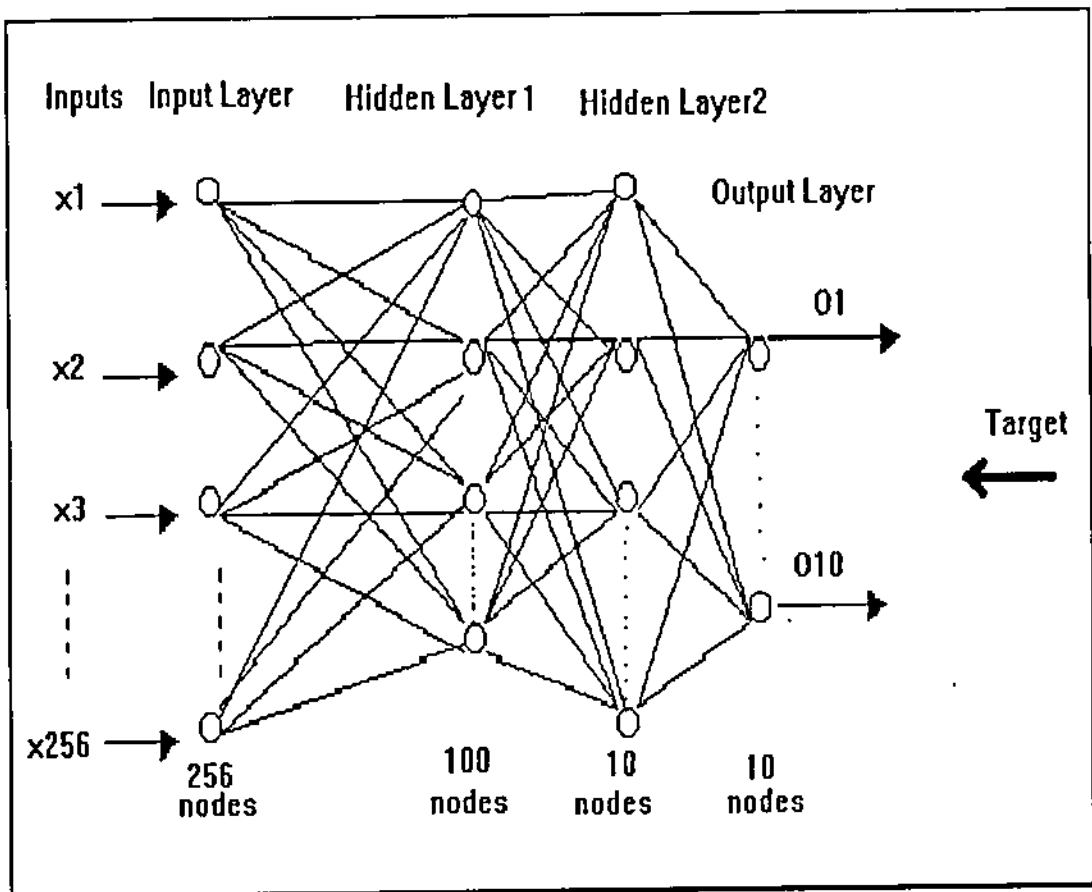


Figure 3.4 Neural Networks Architecture

The training of the neural network is achieved by presenting the training set to the neural network. The presentation is done by taking the samples of all digits in a training set and the target of each digit, one by one. The computational output of the neural networks is calculated and compared with target output. If the calculated output is the same of the target output that means, the neural network recognize the input image successfully. Otherwise mean square error (MSE) method is used to find the error with acceptable accuracy. If the calculated error is less than or equals the acceptable error value or accuracy, the training is terminated. This acceptable value of error is called training error goal. In our experiments, the value of the training error goal is equal 0.01. This value is determined by making many experiments using different values of error goal, until we get this value 0.01.

When the error goal of training is reached the training of the neural network is terminated. There is another method for termination of training. When the number of the training iteration (epochs) is reached the maximum input given number, the training of the neural networks is terminated. In our experiments we use the first method, i. e. when error goal is reached.

The error goal curve between error value and number of epochs must be converging to reach the error goal. Figure 3.5 displays the curve of error goal. The X-axis determines the number of epochs and the Y-axis determines the error value, and the curve specifies the relation between them with error goal equal 0.01.

This curve in Figure 3.5 is the curve of training neural network that we specified its architecture in section 3.3.2. The training of the neural network has been performed using 1000 samples. We can mention that as soon as the number of the epochs is increasing the error value is decreasing, which means that the curve of error is

converging. This training terminated by one of the two ways that mentioned, and in this case it terminated by reaching its goal.

When the training of the neural networks terminated the generalization ratio of recognition of training and testing set is evaluated. The generalization ratio is evaluated by calculating the number of the correctly recognized digit by the trained neural networks to the whole number of set. This means, the calculated output from the trained networks for every digit in the training set is compared with its target output. If they are the same then the neural network can recognize the digit correctly.

The procedure for finding the generalization ratio of recognizing digit is also used for testing set.

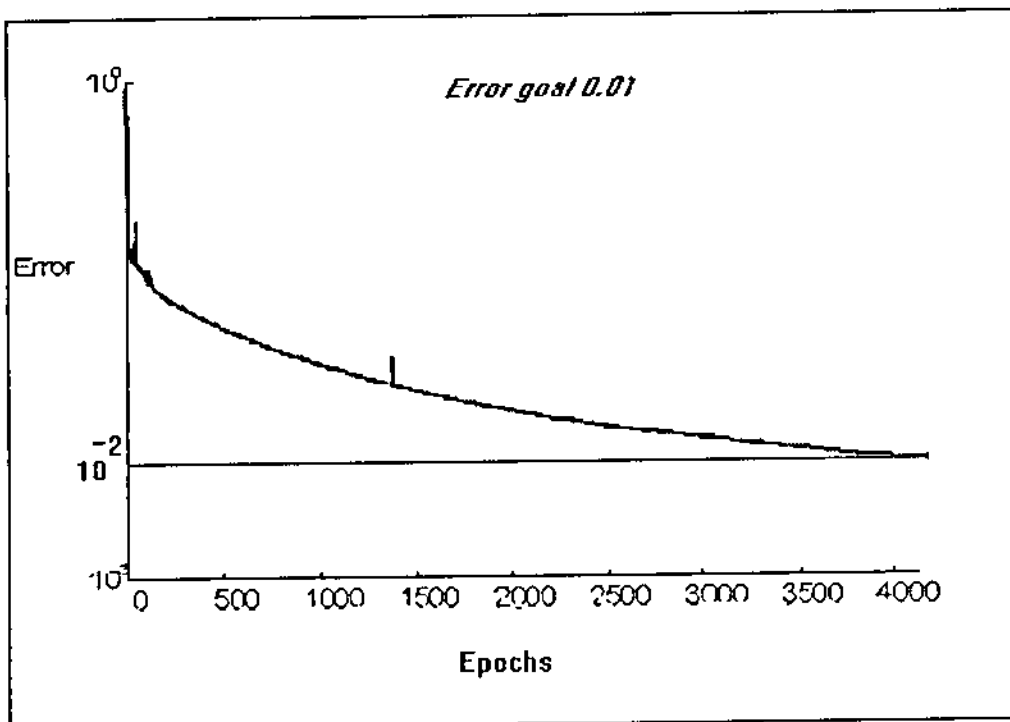


Figure 3.5 Convergence of Mean Square Error

### 3.3.3 Experimental Results

The experiments of neural networks are divided into two groups. Each group of experiment contains 10 experiments. Each group represents the whole data because every time we change the testing set and training set until we cover the whole samples. The 10 experiments in each group are needed because we change the testing set that represents 10% in every experiment. Therefore, we need 10 experiments to cover all samples i.e. to be checked as testing samples.

The first group of experiments is done by using 1000 samples 100 for each digit. Therefore, for each set, 90 samples for training set which represent 90%, and 10 for testing set, which represent 10 % of the samples.

We trained the neural network by using multi layer feed forward neural networks using resilient propagation learning algorithm. The results of these experiments can be seen in Table 3.1. The first column displays the experiment number. In exp.1 the first 10% of the data is used as testing set and the remainder is used as training set. In exp.2 the second 10% of the data is used as a testing set, and the remainder as training set and so on. The second column represents the generalization ration of recognition the digit in training set with error goal 0.01. The generalization ration of digit recognition is calculated as follows:

Calculate the number of the digits that the system can recognize it correctly (calculated output = target output for every sample). After training process is complete, the digit recognition is performed. Divide this number by the number of the whole samples in the sets either training or testing; and the multiply the result by 100%.

The third column represents the generalization ratio of recognition for the testing set.

Experiments results can be seen in Table 3.1 where the average generalization ratio of recognition digits in training set is 95.1667, and average generalization ratio of recognition digits in the testing set is 83.65. We shall focus on the result of testing set because we want the system to recognize digits that it never see it before or during training. And also we try to increase this ratio.

Table 3.1 Neural Network Experiments with (1000 Samples and 0.01 Error Goal)

Experiment Number	Generalization Ratio of Training Set	Generalization Ratio of Testing Set
Exp. 1	94.111	84.5
Exp. 2	96.3889	84
Exp. 3	95.111	86.5
Exp. 4	95.944	82
Exp. 5	94.833	75.5
Exp. 6	96.5556	89.5
Exp. 7	94.115	84
Exp. 8	95.3889	85.5
Exp. 9	98.556	86
Exp. 10	95.1667	79
-----	-----	-----
Average	95.61701	83.65



Figure 3.6 shows the percentage average of recognition of training and testing sets. The upper curve represents the training set recognition and the lower curve represents testing set. The generalization ratio curve of training is always upper than generalization ratio recognition curve of testing set because the neural network parameters has been justified according to the learning procedure that performed on a training set. The generalization ratio for both is not constant, but it is varying. The reason is that, when the training is complete the neural networks produce its parameters like weight values. These parameters produced according to the giving input and the error goal and its learning algorithm. The polynomials that can be used them for recognizing is determined upon these parameters. So these polynomial result varying according to the mentioned parameters. Therefore, the results for all experiments is not constant.

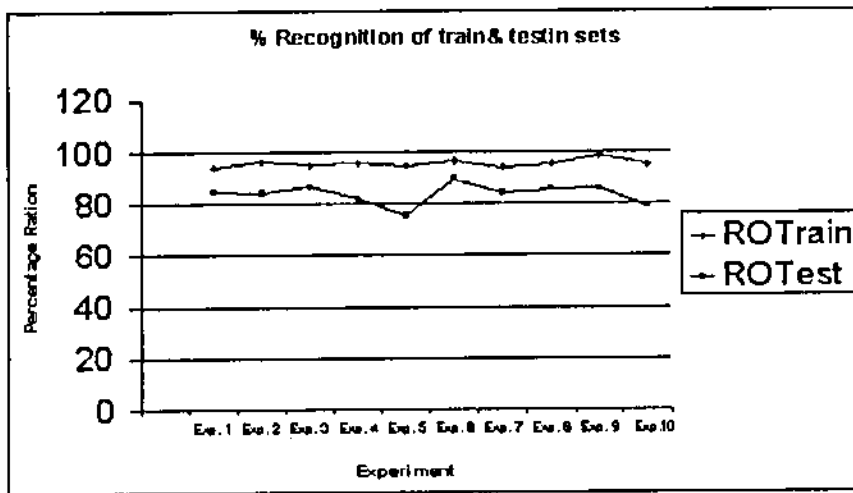


Figure 3.6 Generalization Ratio Curves of Training and Testing Sets

To improve the ratio of recognition for testing value we increase the number of the samples used for training and testing. In our experiments we use 2000 samples, 200 for each digit. Therefore, the size of training set in every experiment was 1800 samples, 180 for each digit which represent 90%, and 20 for testing set which represent 10 % of the samples.

We trained the neural network by using the same neural network i.e. multi layer feed forward neural networks using resilient propagation learning algorithm, and 0.01 error goal.

The results of these experiments can be seen in Table 3.2. The first column displays the experiment number. The second column represents the generalization ration of recognition the digit in training set with error goal 0.01 and the third column displays the generalization ratio of recognition the testing sets.

Experiments result can be seen in Table 3.2. The average generalization ratio of recognition digits in training set is 97.32268 and for testing set 91.6.

These results and the relations between training and testing curve of the ten experiment illustrated in Figure 3.7. The curve of percentage average of recognition ratio for the training set is upper the curve of percentage average of recognition for testing sets. The reason is that the neural network calculates its parameter according to the training set. And also the generalization ration for both is not constant but it is varying. The reason is when the training is complete the neural network produce calculate its parameters according depending on the giving input. We can mention from all these experiment that the pattern recognition is case dependent, which mean that every case or every sample has its situation. i.e. recognizing some digits does not mean recognizing all

digits in the same way, but some time it can recognize some digits and it cannot recognize the others.

Now according to the testing set we focus on, the average of the obtained results of the testing set is 91.6. We will focus on this neural network and its result. In Table 3.3, we can see the results of recognition the digits. Column one represents the experiments. Columns two to column 11 represent the digits from 0 to 9. Column 12 represents the summation of unrecognized digits. Finally, column 13 represents the generalization ratio of recognizing the digits through experiments. The last row represents the average generalization ratio of recognition of every digit by taken from all experiments.

Table 3.2 Neural Network Experiments with (2000 samples and 0.01 Error goal)

Experiment Number	Generalization Ratio of Training Set	Generalization Ratio of Testing Set
Exp. 1	95.1111	90.5
Exp. 2	98.3889	95
Exp. 3	97.5566	94.5
Exp. 4	97.8333	89.5
Exp. 5	98.5556	89
Exp. 6	97.5556	93.5
Exp. 7	96.1115	89
Exp. 8	95.3889	89.5
Exp. 9	98.5566	90
Exp. 10	98.1667	95.5
Average	97.32268	91.6

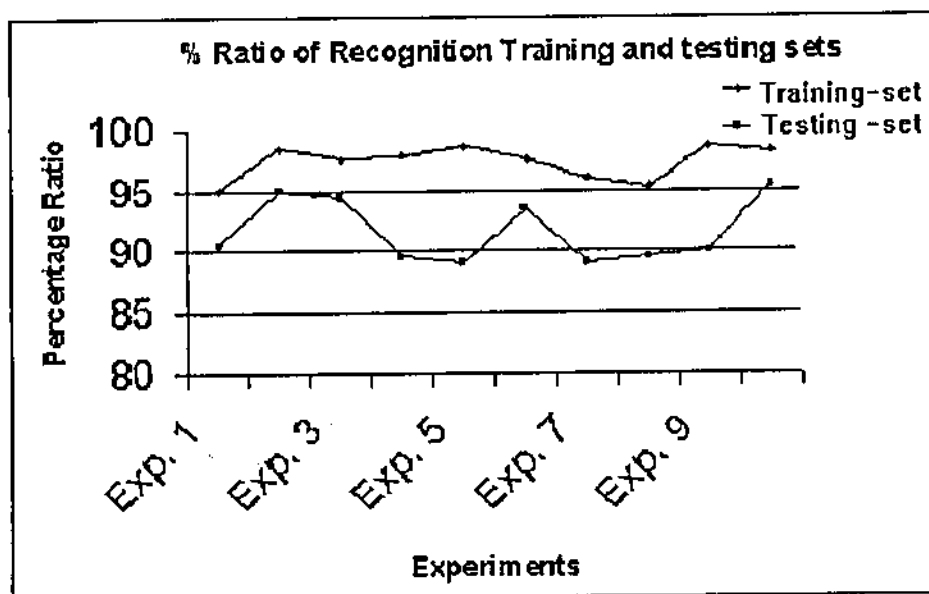


Figure 3.7 Generalization Ration Curve of Recognition  
of Training and Testing Sets with (2000 samples)

Table 3.3 NNW Result of Digits Recognition

Using 2000 sample for Training

Exp. /Digit	0	1	2	3	4	5	6	7	8	9	Total	% 100
Exp1	2	1	5	0	0	6	1	1	2	1	19	90.5
Exp2	1	0	2	0	0	1	2	1	1	2	10	95
Exp3	0	0	0	4	0	0	3	0	3	1	11	94.5
Exp4	1	1	2	3	1	2	2	1	5	3	21	89.5
Exp5	0	0	3	4	1	3	2	3	4	2	22	89
Exp6	0	0	1	1	0	1	3	1	3	3	13	93.5
Exp7	2	0	2	2	1	2	3	1	4	5	22	89
Exp8	1	1	3	3	0	3	3	2	2	3	21	89.5
Exp9	1	0	3	3	2	3	2	1	2	3	20	90
Exp10	0	0	2	1	0	0	1	0	1	0	5	97.5
Total	8	3	23	21	5	21	22	11	27	23	164	91.8

The experiments show that digits 1 can be recognized in 98.5 %. It is the highest generalization ratio of recognition the digits. Digit 8 is the lowest generalization ratio of recognition that equals to 86.5. This can be seen in Table 3.4. Other digits recognition ratio located between these two values. We can return this result to the complexity of writing digit 8 according to writing digit 8, hence varying of the samples of digit 1 is smaller than samples of digit 8. Also digit 8 is some home closer to digit 9, digit 3, digit 2, in the shape. Which means, its shape like the shape of the mentioned digits. That make the percentage recognition of these digits is lower than the percentage recognition of digit 1.

Table 3.4 Average Generalization Ratio of Recognition Each Digit  
Using Neural Networks

Exp. /Digit	0	1	2	3	4	5	6	7	8	9
Generalization ratio of recognition digits	96	98.5	88.5	89.5	97.5	89.5	89	94.5	86.5	88.5

Figure 3.8 shows the representation of the digits and the generalization ratio of recognition every digit. Where the x-axis represents the digits and the y-axis represent the generalization ratio of recognition the digit.

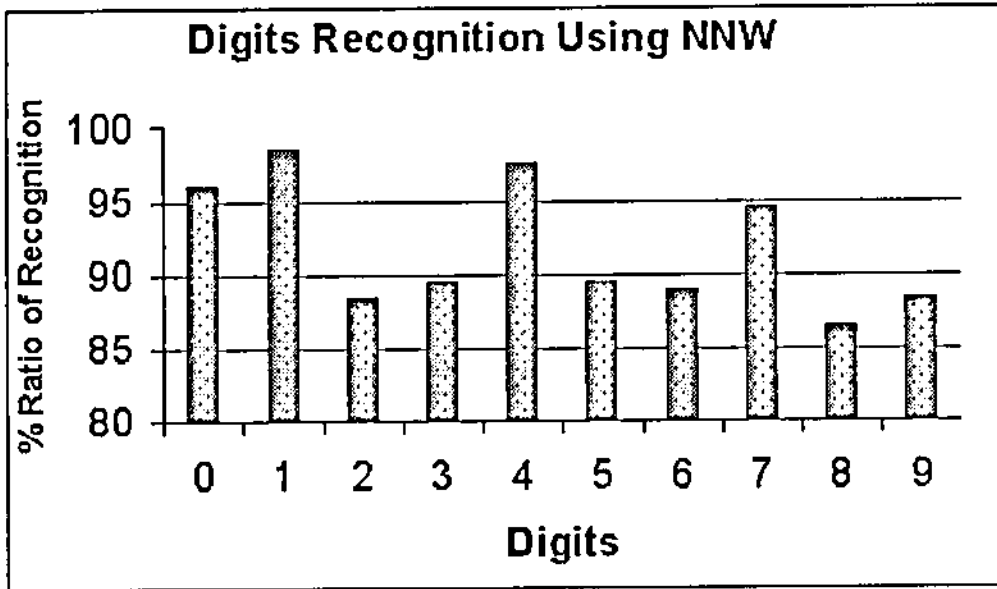


Figure 3.8 Digits Recognition Using Neural Network

We have two neural networks with their results. To compare between these neural networks Table 3.5 shows that the recognition of the digits using the first neural networks (that trained using 1000 samples) is 83.65 and using the second neural networks (that trained using 2000 samples) is 91.6. Therefore, we can see that as the number of sample increase the generalization ratio of recognition is increasing. According to this results we shall use the second neural network in the following comparisons.

Table 3.5 Comparison between Neural Networks with  
1000 samples and 2000 samples for Recognition Testing sets

Experiment number	Generalization Ratio of Neural network (with 1000 samples)	Generalization Ratio of Neural network (with 2000 samples)
Exp. 1	84.5	90.5
Exp. 2	84	95
Exp. 3	86.5	94.5
Exp. 4	82	89.5
Exp. 5	75.5	89
Exp. 6	89.5	93.5
Exp. 7	84	89
Exp. 8	85.5	89.5
Exp. 9	86	90
Exp. 10	79	95.5
=====	=====	=====
Average	83.65	91.6

Figure 3.9 shows the curve of the ten experiment for first neural network (NNW1) that use 1000 samples and the second neural network (NNW2) that use 2000 samples. From this curve, it is clear that NNW2 gives better results among the 10 experiments.

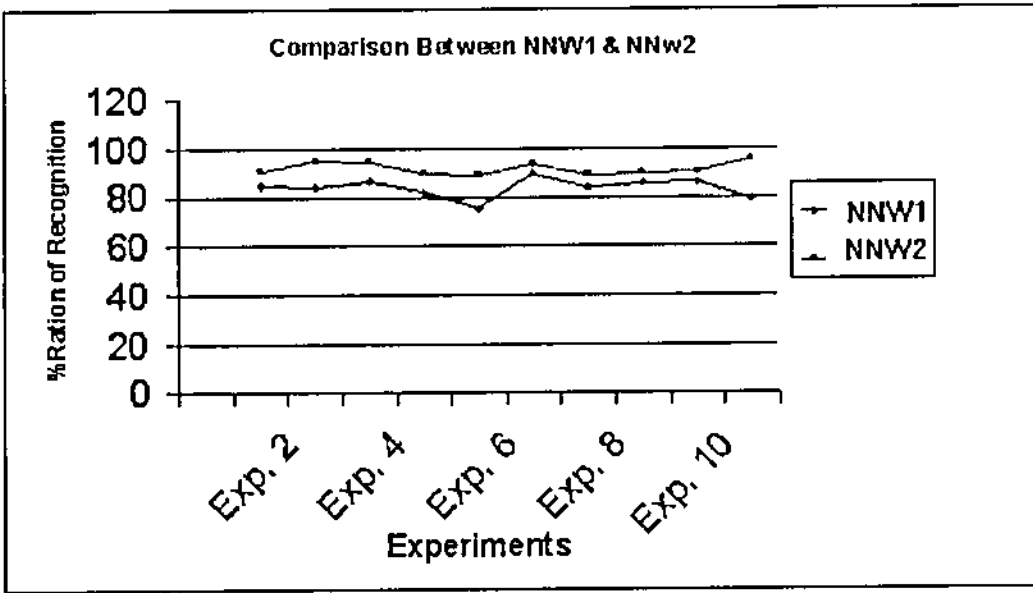


Figure 3.9 Comparison between NNW1 and NNW2 for Training Sets

The percentage average for recognition of the ten experiments can be seen in Figure 3.10. The X-axis represents the first neural network (NNW1) that use 1000 samples and the second neural networks (NNW2) that use 2000 samples. The Y-axis represents the percentage average of generalization ratio of recognition digits. This Figure shows that the second neural network gives better results than the first neural network. This means, as the size of the training set increases the generalization ratio of recognition of the digit increases.



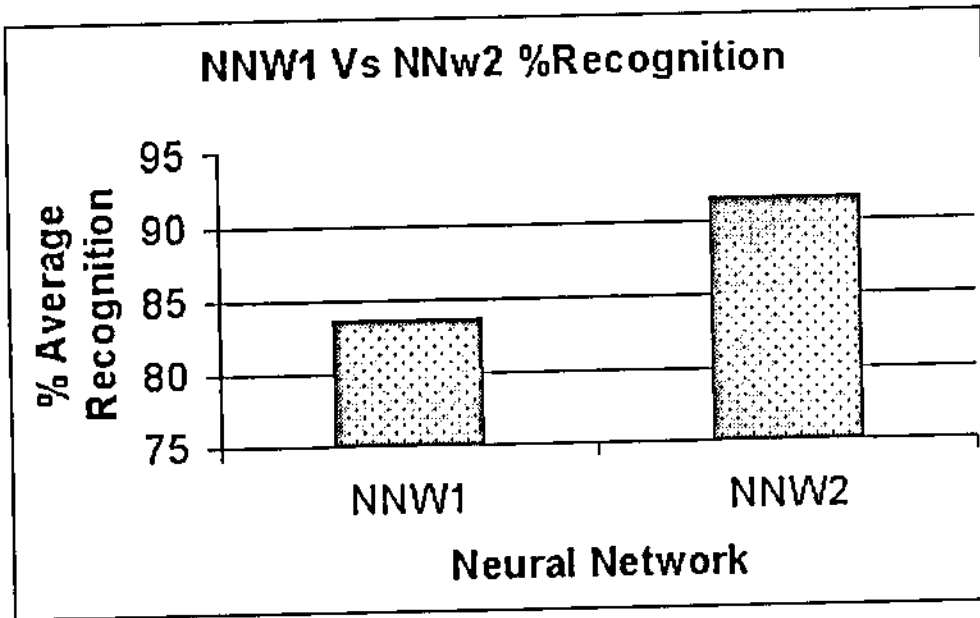


Figure 3.10 Comparison between NNW1 and NNW2 for Testing Sets

### 3.4 Conclusion

Our goal was to design a neural network system that can recognize the handwritten digits. Alternative techniques used to obtain this goal. We designed a multi layer feed forward neural network, with Resilient Propagation (RPROP) learning algorithm. This neural network consists of 4 layers: 256 nodes for input layer, 100 node for first hidden layer, 10 nodes for second hidden layer, and 10 for output layer. We made experiments using two different sets of different sizes of training sets. The first experiment used 1000 samples, and the second used 2000 samples. The results discussed in this chapter show that, as we increase the size of the training set we improve the results of recognition digits using neural network. Our final result of generalization ratio of

recognition of the digits for testing set was 83.65 by using 1000 samples and 91.6 for testing set by using 2000 samples.

The goal was achieved by designing a neural network that can recognize digits with an average generalization ratio of recognizing 91.6 for digits that the neural never saw during training.

## Chapter 4

# Handwritten Arabic Numbers Recognition Using Neuro Fuzzy Systems

## Chapter 4

# Handwritten Arabic Number Recognition Using Neuro Fuzzy Systems

### 4.1 Introduction

In this chapter we design a neuro fuzzy system to recognize Arabic numbers. After performing the preprocessing operations such as: Segmentation, binarization, normalization, and location, we mentioned before, we obtained Windows Bitmap files of size 16 x 16 pixels, each containing an image of isolated digit in a suitable place. These data files were separated into two sets training and testing sets, where 90% of the data for training, and 10% for testing.

The Neuro fuzzy system is a system that uses neural network properties such as computational power parallel facility, fault tolerance, and learning capability. Fuzzy logic and fuzzy set where remain as means as for representing, manipulation and utilizing uncertain information and to provide a framework handling uncertainties associated with real world problems such as a handwritten recognition.

## 4.2 Neuro Fuzzy System Architecture

The goal of our experiments is to design a neuro fuzzy system that can recognize handwritten numbers. This achieve by derive fuzzy rule from the input pattern and their targets, hence we are using a supervised learning. The input samples in our case are represented in a crisp mode, and are used to derive rule for recognizing the digit. That means to classify each input pattern to the class that belongs to, using supervised learning algorithm with feed forward multi layer fuzzy error backpropagation.

The fuzzy rules we want to derive from data to recognize the digits, of the form of fuzzy if-then-rule as:

R: If  $X_1$  is  $\mu_1$  and  $X_2$  is  $\mu_2$  and ...  $X_n$  is  $\mu_n$  Then pattern  $(X_1, X_2 \dots X_n)$  belongs to  $D_i$ .

Where:

R: Rule.

$X_1, X_2, \dots X_n$  : Pattern features.

$\mu_1, \mu_2, \dots \mu_n$  : Fuzzy set .

$D_i$ : Digit class. i.e the class that the pattern belongs to.

So, we have to determine these rules, using supervised learning algorithm with fuzzy backpropagation. We need to determine the membership function that related to each input, i.e. fuzzy sets member, to determine the correct class of the input pattern.

As we see in this rules we have input pattern which consists of  $(X_1, X_2, \dots, X_n)$  features. These features represent the input crisp vector to the neuro fuzzy system. We have  $D_i$  which represents the crisp class of the pattern (i.e. the digit that the input pattern belongs to). We have also fuzzy sets  $(\mu_1, \mu_2, \dots, \mu_n)$  in between. Therefore, we need a processing between crisp to fuzzy (fuzzification) and from fuzzy to crisp (defuzzification).

In our experiments, we have three layers feed forward backpropagation :

The first layer is the input layer that takes the features that represent each pattern. In this layer, every input feature  $(x)$  is associate with a fuzzy set. We use Triangular membership function that is specified by three parameters  $(a, b, c)$  as follows:

$$\mu(X) = \text{triangle}(x; a, b, c) = \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a \leq x < b \\ \frac{c-x}{c-b}, & b \leq x \leq c \\ 0, & x > c \end{cases} \quad (4.1)$$

Where

$a, b, c$  are parameters with  $(a < b < c)$ .

We have alternative expression for the formula (4.1) using minimum (min) and maximum (max) operations:

$$\text{triangle}(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \quad (4.2)$$

The second layer is the hidden layer, which holds the units, where each unit represents the fuzzy rules.

In this layer, the output of every units including in this layer, is calculated by the following formula:

$$a_{R_i} = \min(W(x_1, R_i), W(x_2, R_i), \dots, W(x_n, R_i)) \quad (4.3)$$

Where

$a_{R_i}$ : the output of the  $i$ 'th rule unit.

$x_1, x_2, \dots, x_n$ : input pattern features.

$R_i$ :  $i$ 'th rule.

$W(x_j, R_i)$ : fuzzy weight on the connection between  $x_j$  and  $R_i$ , where  $j=1, 2, \dots, n$ .

The fuzzy set and linguistic rule in the first and second layer will be obtained from a training set.

The third layer represents the output layer. This layer consists of units that determine the digit that the input pattern belongs to. The output of each unit is calculated by:

$$O_i = \frac{\sum W(R_j, ou) a_{R_j}}{\sum W(R_j, ou)} \quad (4.4)$$

Where

$a_{R_j}$ : activation output of the rule unit that is connected with output unit  $i$ ,

$W(R_j, ou)$ : the weight on the connection from the rule unit  $j$  to the output unit  $i$ .

The architecture of the neuro fuzzy system can be seen on Figure 4.1 with the three layers: input layer, hidden layer, and the output layer.

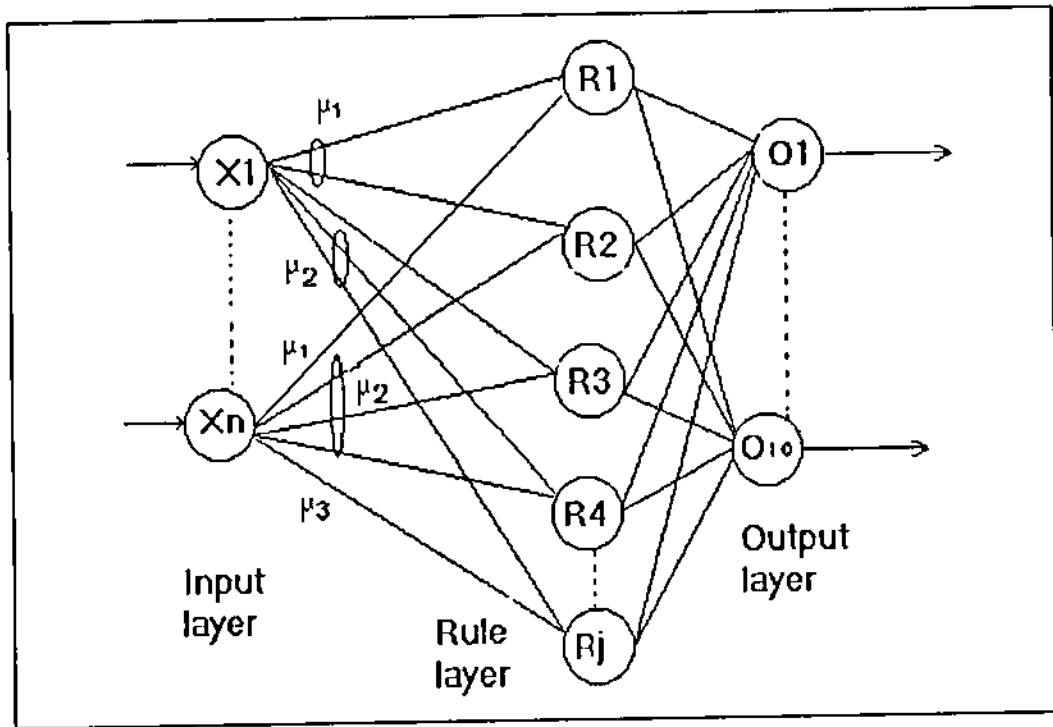


Figure 4.1 Neuro fuzzy System Architecture

According to the output, we calculate every unit's output. The unit that has the highest value will be the unit that represents the class where the input pattern belongs.

After we select a pattern which consists of  $(x_1, x_2, \dots, x_n)$ , and its target from a training set, we find the membership function of each  $x_i$  where  $i = 1, 2, \dots, n$ . After that we determine the membership function of the rule using formula (4.3). Then connecting the rule nodes to the output unit and computing the output value.



If the calculated error is greater than the accuracy that we specify, then by using backpropagation algorithm, we update the value of the parameters of the triangular (a, b, c) and apply the changes of each element in the fuzzy set. Repeat this procedure until we get the accuracy that we specify.

### 4.3 Experimental Work

In our experiments, 2000 samples are used. For each digit, 200 samples are used. The size of training set in every experiment is 1800 samples, 180 for each digit which represent 90%. And 200 for testing set which represent 10 % of the samples.

The results of this system can be seen in Table 4.1. The first column displays the experiment number. The second column represents the generalization ration of recognition the digit in training set with error goal 0.01. The third column displays the generalization ratio of recognition the testing sets. The table shows the percentage average of the generalization ratio of recognition digits in training set is 98.2777 and for testing set 98.85.

Table 4.1 Neuro Fuzzy System Experiments

Experiment Number	Generalization Ratio of Training Set	Generalization Ratio of Testing Set
Exp. 1	97.22222	95
Exp. 2	98.88889	97.5
Exp. 3	98.33333	97
Exp. 4	97.77778	94.5
Exp. 5	98.83333	94.5
Exp. 6	97.82222	95.5
Exp. 7	98.33333	94
Exp. 8	98.88889	95
Exp. 9	98.88889	97.5
Exp. 10	98.88889	98
Average	98.2777	95.85

Table 4.2, shows the result of recognition the digits by the neuro fuzzy system. Column one represents the experiments. Columns two through eleven represent the digits from 0 to 9. Column twelve represents the summation of unrecognized digits. Finally, column 13 represents the generalization ratio of recognizing the digits through experiments. The last row represents the percentage average of generalization ratio of the recognition of every digit taken from all experiments.

Digit 1 has the highest average generalization ratio of recognition taken from all experiments. Figure 4.2 shows the results where we can see the representation of the

digits and the average generalization ratio of recognition every digit taken from all experiments. The X-axis represents the digits. The Y-axis represents percentage averages of generalization ratio of recognition the digit. Digit 1 has the highest percentage of generalization ratio of recognition that equals to 99, where as, digit 8 has the lowest percentage of generalization ratio and equals to 93.

Table 4.2 Neuro Fuzzy System Result of Digits Recognition

Exp./Digit	0	1	2	3	4	5	6	7	8	9		Total	Average
Exp1	0	0	4	0	0	4	0	0	2	0		10	95
Exp2	0	0	1	0	0	1	1	1	0	1		5	97.5
Exp3	0	0	0	2	0	0	1	0	2	1		6	97
Exp4	0	1	1	2	1	1	2	0	2	1		11	94.5
Exp5	0	0	2	3	1	1	1	1	1	1		11	94.5
Exp6	0	0	0	1	0	1	2	0	3	2		9	95.5
Exp7	1	0	1	1	1	1	1	1	3	2		12	94
Exp8	1	1	1	2	0	2	1	0	0	2		10	95
Exp9	1	0	0	0	2	1	0	0	0	1		5	97.5
Exp10	0	0	1	1	0	0	1	0	1	0		4	98
=====	===	==	==	==	==	==	==	==	==	==	=	==	===
<b>Total</b>	3	2	11	12	5	12	10	3	14	11		83	95.85
=====	===	==	==	==	==	==	==	==	==	==	=	==	===
<b>Average generalization ratio</b>	98.5	99	94.5	94	97.5	94	95	98.5	93	94.5		8.3	95.85

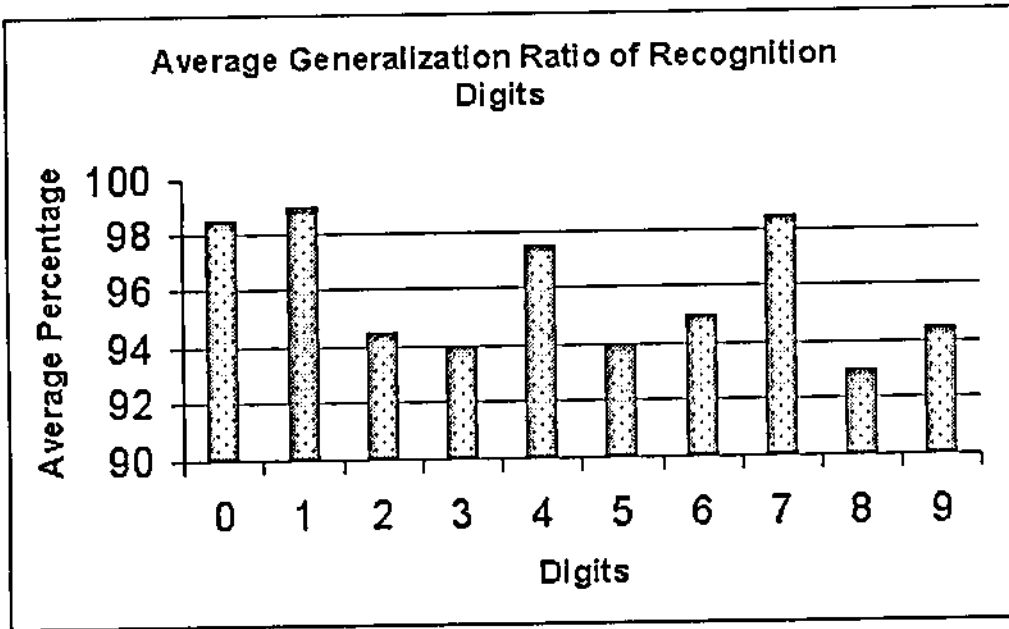


Figure 4.2 Digits Recognition using Neuro Fuzzy System

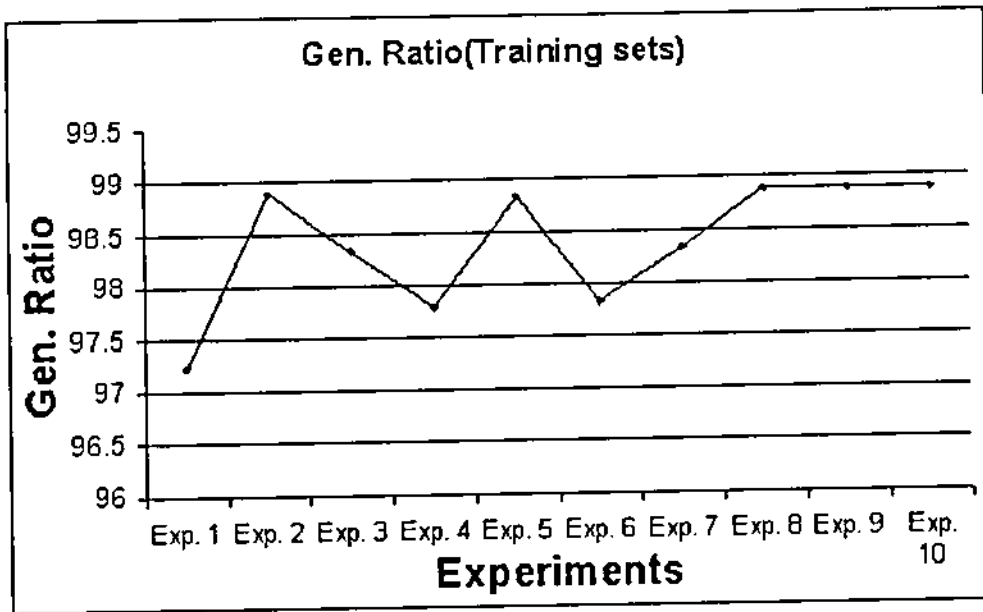


Figure 4.3 Generalization Ration of Training Data Using

Neuro Fuzzy Systems

Figure 4.3 shows the representation of the digits and the generalization ratio of recognition the training set for every experiment. The X-axis represents the experiments. The Y-axis represents the generalization ratio of recognition for every experiment.

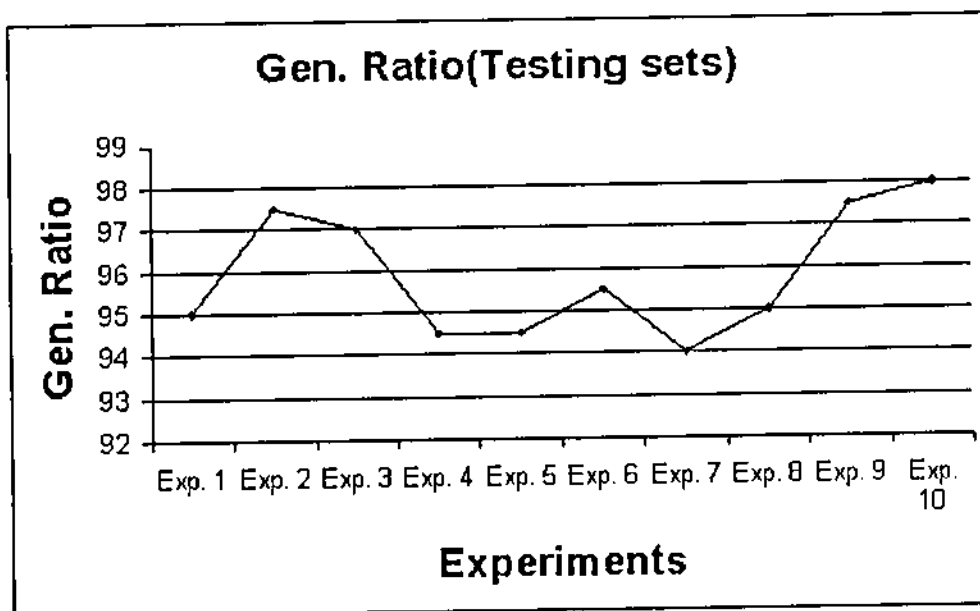


Figure 4.4 Generalization Ration of Testing Data Using  
Neuro Fuzzy Systems

Figure 4.4 shows the graph of the generalization ratio of recognition the testing set for every experiment. The X-axis represents the experiments. The Y-axis represents the generalization ratio of recognition for every experiment.

## Chapter 5

### Conclusion and Recommendation

## Chapter 5

### Conclusions and Recommendation

In this thesis, we have investigated two main techniques for handwritten Arabic number recognition. The first technique is concerned with multi layer feed forward neural network with resilient propagation and the second is concerned with neuro fuzzy systems.

In both techniques, we have 2000 samples. Different steps of preprocessing operations have prepared these samples. Preprocessing operations such as Segmentation, Normalization, Binarization, and Location.

By using multi layer feed forward neural network with resilient propagation, the following results have been obtained: The percentage average of generalization ratio for recognizing the training sets is 97.32278 and for testing sets is 91.6. The second technique focuses on neuro fuzzy systems; the percentage average of generalization ratio for recognizing the training sets is 98.2777 and for testing sets is 95.85.



## 5.1 Conclusions

Results have been obtained from the comparisons made between the neural networks and neuro fuzzy systems for recognizing the digits.

Table 5.1 shows the comparison between neural networks and neuro fuzzy systems for recognizing the training sets. In every experiment, the neuro fuzzy system yields results better than neural network in recognizing the training sets. This means that the neuro fuzzy systems can recognize the training sets more accurately than the neural networks. The percentage average of generalization ratio for recognizing the training sets is 97.32278 using neural networks and the percentage average of generalization ratio for recognizing the training sets is 98.2777 using neuro fuzzy systems. Therefore, the neuro fuzzy systems can recognize the training sets better than the neural network. This can be demonstrated in Figure 5.1. The curve of recognition using neuro fuzzy systems is consistently located upper the curve of neural network.

Table 5.1 Comparison between Neural Network and Neuro Fuzzy Systems  
For Recognizing Training Sets

Experiment number	Generalization Ratio of Neural Network	Generalization Ratio of Neuro Fuzzy Systems
Exp. 1	95.1111	97.22222
Exp. 2	98.3889	98.88889
Exp. 3	97.5566	98.33333
Exp. 4	97.6333	97.77778
Exp. 5	98.5556	98.83333
Exp. 6	97.5556	97.82222
Exp. 7	96.1115	98.33333
Exp. 8	95.3889	98.88889
Exp. 9	98.5566	98.88889
Exp. 10	98.1667	98.88889
Average	97.32268	98.2777

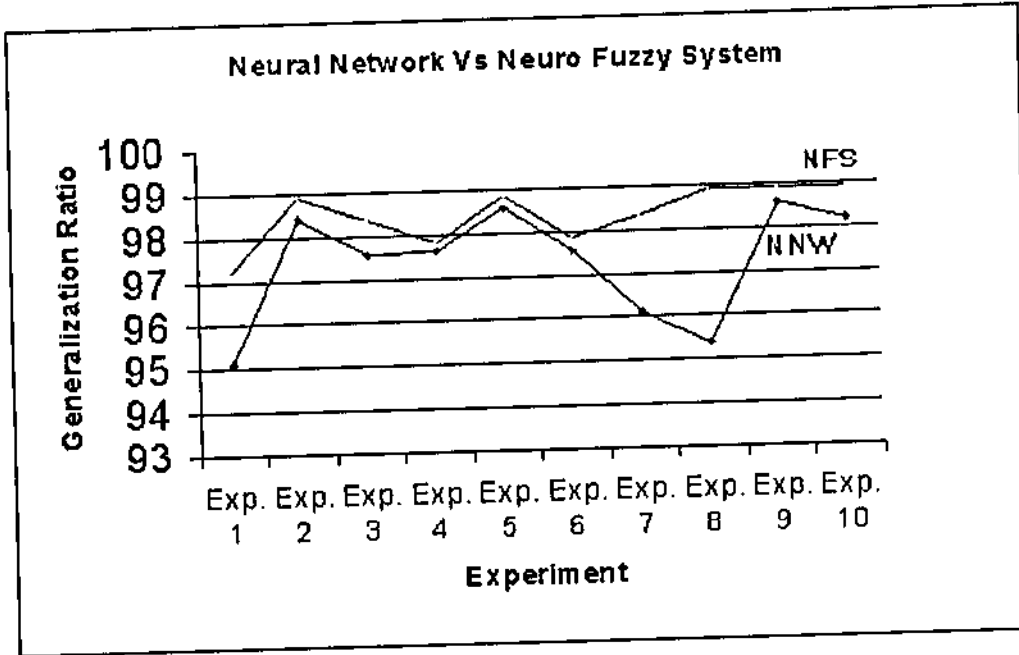


Figure 5.1 Neural Network Vs Neuro Fuzzy Systems  
For Recognizing Training Sets

On the other hand, the comparison of recognizing the testing sets between neural network and neuro fuzzy systems can be seen in Table 5.2. The neuro fuzzy systems seem more accurate for recognizing testing sets than neural network. Table 5.2 shows clearly that the results obtained by using neuro fuzzy systems for recognizing testing sets are relatively greater than the results obtained by using neural networks. Therefore, the percentage average of generalization ratio for recognizing digits using neuro fuzzy systems is 95.85. On the other hand, the percentage average of generalization ratio for recognizing digits using neural network is 91.6. These results can be seen in Figure 5.2. The curve of recognition using neuro fuzzy systems is located above the curve of neural network.

This means that neuro fuzzy systems can recognize the testing sets better than neural network.

Obviously, recognition of training sets and recognition of testing sets, when using neuro fuzzy systems for recognition digits gives better and more accurate results than using neural network.

Table 5.2 Comparison between Neural Network and Neuro Fuzzy Systems  
For Recognizing Testing Sets

Experiment Number	Generalization Ratio of Neural Network	Generalization Ratio of Neuro Fuzzy Systems
Exp. 1	90.5	95
Exp. 2	95	97.5
Exp. 3	94.5	97
Exp. 4	89.5	94.5
Exp. 5	89	94.5
Exp. 6	93.5	95.5
Exp. 7	89	94
Exp. 8	89.5	95
Exp. 9	90	97.5
Exp. 10	95.5	98
Average	91.6	95.85

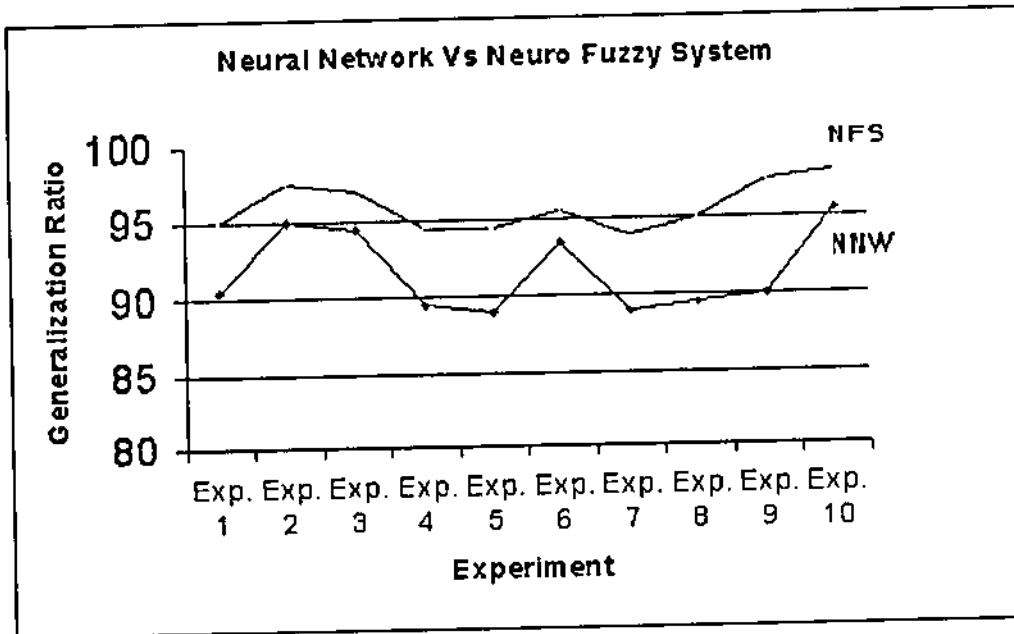


Figure 5.2 Neural Network Vs Neuro Fuzzy Systems  
For Recognizing Testing Sets

## 5.2 Future Work

In designing a complete handwritten recognition system, one must face the problem of where effort can be most efficiently applied to increase the performance. In order to increase the generalization ratio for recognition, research can be conducted in future in this field.

Increasing the number of samples that are used in training and testing sets will enhance the result of recognition.

Using genetic algorithm and neural networks that are both optimization algorithm in which genetic algorithm seeks to find one or more members of a population that represents a solution to the found set of (weights) that minimize the number of incorrect recognition, will improve the performance of recognition.

The application of the system on handwritten numbers could extend the ability to recognize handwritten characters.

Developing the preprocessing operations where the preprocessing operations can improve the results of recognition, especially segmentation algorithm for Arabic handwritten characters recognition, will be a field of interest for research in future.

Finally, more complex images, including color images, could be applied to the systems.

## References

- Adel, M., M. Ali. 1997. An Evolutionary Neuro-Fuzzy Approach to Recognize On-Line Arabic Handwriting, 4th International Conference Document Analysis and Recognition (ICDAR '97), Institute of Electrical and Electronics Engineers, Inc.
- Al-Badr, B., and R.M. Haralick. 1992. Recognition without Segmentation: Using Mathematical Morphology to Recognize Printed Arabic. Proceedings 13th National Computer Conference, Riyadh, Saudi Arabia, pp. 813—829.
- Al-Badr, B. , and R.M. Haralick. 1994. Symbol Recognition Without Prior Segmentation. Proceedings of the IS&T/SPIE Symposium on Electronic Imaging Science and Technology: Document Recognition, San Jose, Vol. 2181, pp. 303—314.
- Al-Badr, B. , and R.M. Haralick. 1995. Segmentation-Free Word Recognition with Application to Arabic, ICDAR'95: Third International Conference on Document Analysis and Recognition, Montreal, Canada .
- Al-Waily, Ali, and Salim Ramzi. 1989. A Study on Processing and Syntactic Recognition of Hand-Written Arabic Character, 1986, Master Thesis, University of Basra, Iraqi.

- Amin, A., H. Al-Sadoun, S. Fischer. 1996. Hand Printed Arabic Character Recognition Using an Artificial Neural Network. *Pattern Recognition*, 29(4), pp 663-675.
- Anil, K. Jain. 1989. *Fundamental of Image Processing*. Prentice Hall, Inc.
- Atlas, Les, Ronald Cole, Yeshwant Muthusamy, and, Alan Lippman.1990. A Performance Comparison of Trained Multiplier Perceptrons and Trained Classifier Tree, *Proc. IEEE* Vol. 78 No. 10, pp. 1614-1619.
- Augusteijn, M.F, and U.J. Steck. 1998. Supervised Adaptive Clustering: Ahybrid Neural Network Clustering Algorithm, *Neural Computing and Application*, Vol. 7, pp. 78-89.
- Bart, Kosko. 1992. *Neural Networks and Fuzzy Systems*. Prentice-Hall Inc Englewood Cliffs, New Jersey.
- Belkasim, O. S., M. Sharidhar and M. Ahmad. 1992. Pattern Classification Using An Efficient KNNR , *Pattern Recognition* Vol. 25 No.10, pp1269-1274,. Printed in Grait Britain.
- Clause, Neubauer. 1998. Evaluation of Convolutional Neural Networks for Visual Recognition, *IEEE Transaction on Neural Network*, Vol. 9, No.4, pp. 685-396.



- Cun, Y. Lee, B. Boser, J. S. Denker, and D. Henderson. 1998. Handwritten Digit Recognition With A Back-Propagation Neural Network, IEEE Transaction on Neural Network, Vol. 9, No. 2.
- Donald, A. 1998. Pattern Recognition, Prentice-Hall Inc., USA.
- Edelman, M., George N. Reeke, JR., and Olaf Sporns. 1990. Synthetic Neural Modeling: The Darwin Series of Recognition Automata, Proc. IEEE, Vol. 78, No. 9, pp. 1498-1530.
- Finn, G.D. 1999. Learning Fuzzy Rules From Data, Neural Computing and Application, Vol. 8, pp. 9-24.
- Fukushima, K., E. Kimura and Shouno. 1998. Neocognitron with improved Bend-Extractors: Recognition of Handwritten Digits in the Real World, Neural Computing and Application, Vol. 7, pp. 260-272.
- Gonzales, Rofael, and Bool Wenter. 1987. Digital Image Processing , 2nd edition. Addison-Wesley Publishing Company .USA.
- Gonzalez, Rafael C., and T. Tou Julius. 1981. Pattern recognition Principles. 4'Th Printing, Addison-Wesley Publish Company, Inc.

- Guyon, I. 1991. Application of Neural Networks to Character Recognition. *International Journal of Pattern Recognition and Artificial Intelligence*. Vol. 5, pp 353-382.
- Harlick, Robert M. and Linda G. Shapiro. 1989. Glossary of Computer Vision Terms, *Pattern Recognition*, Vol. 24, No. 1, pp 96-93. Printed in Great Britain.
- Haykin, Simon. 1999. *Neural Networks*, 2nd edition, Prentice Hall, Inc., USA.
- Hsin-Chia, Fu, Yeoung Yuh, and Xu. 1998. Multilinguistic Handwritten Character Recognition by Bayesian Decision-Based Neural Networks, *IEEE Transaction on Signal Processing*, Vol. 46, No.10, pp. 2781-2789.
- Jan, Teuber. 1993. *Digital Image Processing*. Prentice Hall International (U.K.).
- Jang, J.-S. R. , C.-T. Sun, and E. Mizutani. 1997. *Neuro-Fuzzy and Soft Computing* , Printice-Hall, Inc.
- Jang, Rojer, and Jyh-Shing. 1993. ANFIS: Adaptive-Network-Based Fuzzy Inference System, *IEEE Transactions on Systems*, Vol. 23, No3, pp. 665-685.
- Jie, Zhang, and A. Julian Morris. 1999. Recurrent Neuro-Fuzzy Networks for Nonlinear Process Modeling, *IEEE Transaction on Neural Networks*, Vol. 10, No. 2, pp. 313-326.

- Kahan, Simon, Theo Pavlidis, and Henry S. Baird. 1987. On The Recognition of Printed Characters of Any Font and Size, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. PAMI-9 , No 2. Pp. 274-286.
- Kim, B.S., and S. B. Park. 1986. A Fast k Nearest Neighbor Algorithm Based on The Ordered Partition, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 8, No.6, pp. 761-766.
- Kimura, F. , and M. Shridhar. 1991. Handwritten Numerical Recognition Based on Multiple Algorithm, Pattern Recognition, Vol. 24, No. 10, pp. 969-983.
- Lippman,P. Richard. 1987., An introduction to computing with Neural Nets, IEEE ASSSP Magazine, PP 4-22
- Louis, J., and Jr. Galbiati. 1990. Machine Vision and Digital Image Processing Fundamentals. Prentice-Hall Inc Englewood Cliffs, New Jersey.
- Meganati, Massimo, Francesco S. Saviello, and Roberto Tagliaferri. 1998. Fuzzy Neural Networks for Classification and Detection of Anomalies, IEEE Transaction on Neural Network, Vol. 9, No.5, pp. 848-861.
- Michael, Blumenstein, and Brijesh Verma. 1996. A Neural Network for Real-World Postal Address Recognition, IEEE Transaction on Neural Network, Vol 2, No.3.

- Michie, D. , D.J. Spiegelhalter and C. C. Taylor. 1994. Machine Learning, Neural and Statistical Classification , Ellis Hoewood Limited.
- Morton, Nadler, and Eric P. Smith. 1993. Pattern Recognition Engineering, John Wiley and Sons, Inc.. Printed in the united States of Amirica.
- Murad, A. H. 1994. Arabic Character Recognition Using Microprocessor, M.Sc. Thesis, University of Bagdad.
- Nandhakumar and J.K. Aggarwal. 1985. The artificial Intelligence Approach to Pattern Recognition A perspective and Overviews, Pattern Recognition, Vol. 18 No. 6, pp 383-389.
- Nouh A., A. Sultan, and R. Tolba. 1980. An Approach for Arabic Characters Recognition, Journal of Eng. Science, University of Riyad, Vol. 6, pp. 185-191.
- Nyongesa, H. 1998. Enhancing Neural Systems by Fuzzy Logic and Evolutionary Reinforcement, Neural Computing and Application, Vol. 7, pp.121-130.
- Patterson, Dan W. 1996. Introduction to Artificial Intelligence and Expert Systems , Prentice-Hall Inc., Englewood Cliffs, New Jersey.
- Paw, Yoh-Han. 1989. Adaptive Pattern Recognition and Neural Networks. Addison-Wesley.

- Riedmiller, M. and H. Braune. 1993. A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm, Proceedings of the IEEE International Conference on Neural Networks (ICNN) San Francisco, pp. 586-591.
- Riedmiller, M. 1994. Advanced Supervised Learning in Multi-Layer Perceptron From Backpropagation to Adaptive Learning Algorithm, Journal of Computer Standard and Interfaces.
- Senior, Andrew William. 1994. Off-Line Cursive Handwritten Recognition Using Recurrent Neural Networks, Ph. D. Thesis, Trinity Hall, Cambridge, England.
- Sing, Tze Bow. 1992. Pattern Recognition and Image processing. Marcel Dekk. New York Inc.
- Song, Hee-Heon , and Seong-Whan Lee. 1998. A Self-Organizing Neural Tree for Large-Set Pattern Classification, IEEE Transaction on Neural Network, Vol. 9, No.3, pp. 369-380.
- Tabaskis, Joe. 1995. Speech Recognition Using Neural Networks , PhD. Dissertation, Carnegie Mellon, USA.
- Witold, Pedrycz. 1998. Conditional Fuzzy Clustering in the Design of Radial Basis Function Neural Networks, IEEE Transaction on Neural Networks, Vol. 9, No.4, pp. 601-612.

Yacu, S. G. A. 1985. Design and Implementation of English Reading Machine for the Blind and Arabic Character Recognition, M. Sc. Thesis, University of Baghdad.

Yager, Roland R., and P. Dimitar. 1994. Essential of Fuzzy Modeling and Control. John Wiley and Sons Inc., USA.

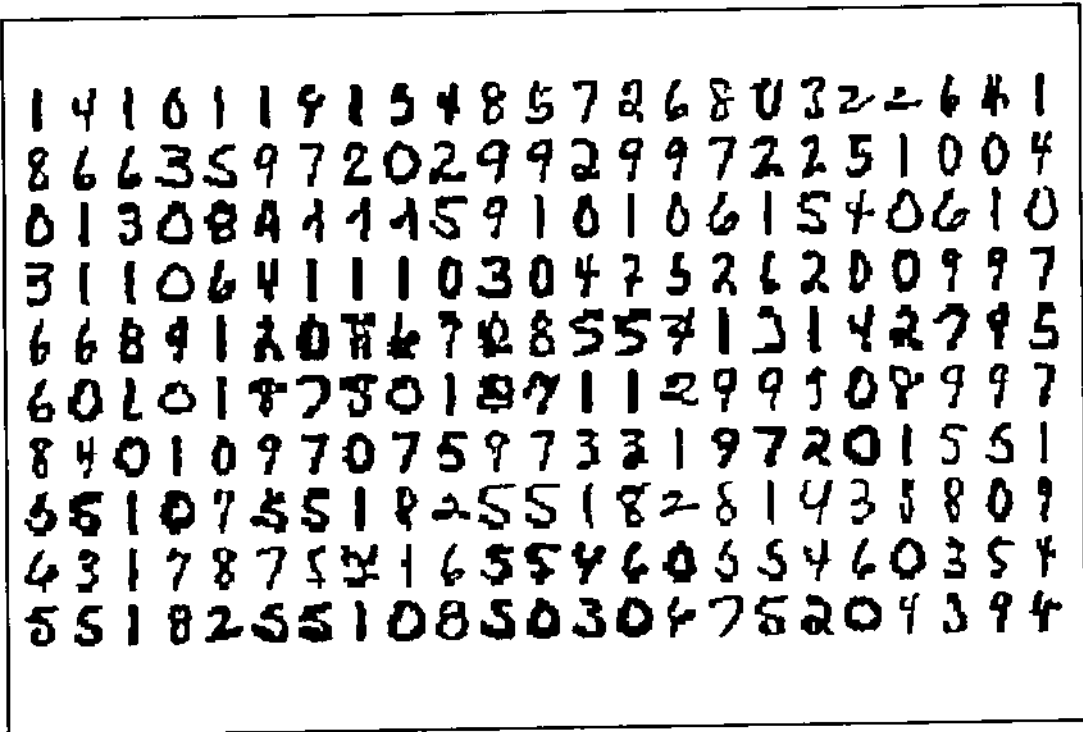
Zadeh, L. A. 1973. Outline of a New Approach to the Analysis of Complex Systems and Decision Process, IEEE Trans. on Systems, Man, and Cybernetics, Vol. 3. pp. 28-44.

Zadeh, L. A. 1988. Fuzzy Logic, Computer, Vol. 21, pp. 83-92.

432631

## Appendix A

## Samples of Handwritten Digits



## Appendix B

### Source Code

Programs are written using MATLAB ver 5.2.0. The source code of some programs will be displayed in the following.

```
%main program

gen_set1; % This file generates two text files train_d.txt
          % and test_d.txt for testing ( 180-200)

[P1,T1]=load_img('train_d1.txt');
[P2,T2]=load_img('test_d1.txt');

% load_img(file_name) function to convert the input
% patterns into two matrices one for input pattern
%( p1,p2) and other for targets (T1, T2)
```



```

P1=P1'; % Take transpose of every matrix to be used in NNW
P2=P2';
T1=T1';
T2=T2';

P1=double(P1); % To insure that take it as a double not
T1=double(T1); % as logical values because its elements
P2=double(P2); % either 0 or 1.
T2=double(T2); % for all

mm=minmax(P1); % Calculate the min & max of training matrix
P1
mm=double(mm);

tic
net=newff(mm,[100,10,10'], {'LOGSIG' 'LOGSIG' 'LOGSIG'},'
'trainrp');
% NNW with 2 hidden layers (226 units and 50 units ) and
% output units
% using Gradient Descent algorithm and sigmoidal function

net.trainparam.goal=0.01;

```

```

net.trainparam.epochs=80000;

net.trainparam.show=25;

]net, tr]=train(net, P1, T1, [], [], [], []);

save mat1.mat % save the parameters of NNW

%Using gen_ratio funtion to calculate the

%generalization ratio of trainingdata (P1) and testing(P2)

% data with there targest T1,T2 respectevly and calculated

% values( Y1,Y2) respectevly.

y1=sim(net,P1); %To find the generalization

%          ration of training samples

TA=gen_ratio(y1,T1(

y2=sim(net,P2); % To find the generalization ration of

GA=gen_ratio(y2,T2) % (testing samples

```

```
%Program Gen_set1
```

```
%The purpose of this program is to generate the file
```

```
%that specify the training and testing sets
```

```
training_set=[1:180]
```

```
testing_set =[181:200]
```

```
prefix)-Pu0', 'Pu1', 'Pu2', 'Pu3', 'Pu4', 'Pu5', 'Pu6', 'Pu7', 'Pu8  
, 'Pu9{'
```

```
fo=fopen('train_d1.txt', 'wt');
```

```
for j= 1:length(prefix)
```

```
    for i=1:length(training_set)
```

```
        set=num2str(training_set(I)
```

```
        name=strcat(prefix{j}, set);
```

```
        name=strcat(name, '.bmp{'
```

```
        code=num2str(j-1{'
```

```
        if (j==length(prefix)) & (i==length(training_set))
```

```
            fprintf(fo, '%s %s', name, code{'
```

```
        else
```

```

        fprintf(fo, '%s %s\n', name, code);
    end; % if
    end; % i
end; %j

fclose(fo);

fo=fopen('test_d1.txt', 'wt');

for j= 1:length(prefix)

    for i=1:length(testing_set)

        set=num2str(testing_set(i));

        name=strcat(prefix{j}, set);

        name=strcat(name, '.bmp');

        code=num2str(j-1);

        if (j==length(prefix)) & (i==length(testing_set))

            fprintf(fo, '%s %s', name, code);

        else

            fprintf(fo, '%s %s\n', name, code);

        end; % if
    end; % i
end; %j

fclose(fo);

function [P,T]=load_img(input_f1)

```

```

%This function read input file and return twomatrices

%P: generate to be the matrix contains images

% T: contains target for P matrix

fi=fopen(input_fl,'rt')

P=[]; % P represents the input array
T=[]; % T represents the Target array
ide=eye(10)

while ~feof(fi)

    file_name=fscanf(fi,'%s',1)

    code=fscanf(fi,'%s',1)

    ncode=str2num(code)

    cp=imread(file_name,'bmp');% cp = current input
    s_cp=size(cp); % s_cp = size of cp
    c_v=[]; % c_v = current vector
    t_v=ide(ncode+1,:); % t_v : target vector

    for i=1:s_cp(1)

        c_v=[ c_v cp(i,:)];%convert picture array to vector

    end; % end of for

    P=[P; c_v];

    T=[T; t_v];

end; % end of while

```

```

function ratio=gen_ratio(input,target_v(

%This funtion findes the generalization ratio of

%recognition for training and testing sets

correct=0;

s=size(input(

for i=1:s(2(

] number,position]=max(input(:,i)((

if ( target_v(position,i) ~= 0(

    correct=correct+1;

end% end if;

end % end for

ratio= (100 * correct) / s(2)

return;

```

%This function make the normalization operation which is  
%one of the preprocessing operations.

```
function [cp]=anorm(input_img)
```

```
a=input_img;
```

```
for i=1:8.
```

```
    ss=sum(a(1)
```

```
    if (ss == 16)
```

```
        for j=1:15.
```

```
            a(j,:)=a(j+1)
```

```
        end %for j
```

```
    end %if
```

```
end % for i
```

```
for i=1:8.
```

```
    ss=sum(a(:,1)
```

```
    if (ss == 16)
```

```
        for j=1:15.
```

```
            a(:,j)=a(:,j+1)
```

## ملخص

## تقنيات لتمييز الأرقام العربية المكتوبة بخط اليد

إعداد

أمجد أحمد هديب

المشرف

الدكتور أحمد الجابر

المشرف المشارك

الدكتور محمد بلال الزعبي

تمييز الأرقام العربية المكتوبة بخط اليد من المواضيع البارزة التي جلبت اهتمام الباحثين، هذا الاهتمام كان بسبب التطبيقات العملية المتعددة لهذا الموضوع مثل التطبيقات الصناعية، والمالية، وغيرها.

تهتم هذه الرسالة بتطوير تقنيات لتمييز الأرقام العربية المكتوبة بخط اليد باستخدام نظام شبكة الخلايا العصبية الاصطناعية، ونظام شبكة الخلايا العصبية المحيرة. حيث يتم إدخال صورة الرقم عن طريق جهاز ( scanner ) ثم يتم التعامل معها عن طريق عمليات تمسح عملية التمييز مثل فصل الخانات التي تمثل الرقم، وضعها بحجم مناسب لعملية التمييز باللونين الأبيض والأسود، ثم وضع الخانة المراد تمييزها في مكانها المناسب. التقنية الأولى اعتمدت على استخدام نظام شبكة الخلايا العصبية الاصطناعية متعددة الطبقات ذي التغذية الراجعة والذي تم اختباره على ( ٢٠٠٠ ) عينة جمعت عشوائياً من أشخاص مختلفين، وكانت نتيجة التمييز ٩١.٦ % . أما التقنية الثانية فقد اعتمدت على نظام شبكة الخلايا العصبية المحيرة التي تم اختبارها على نفس العينة وكانت نتائجها ٩٥.٨٥ %.

وبناء على هذه النتائج تبين لنا أن كلا النظامين مناسبين لتمييز الأرقام العربية المكتوبة بخط اليد، وأن استخدام نظام شبكة الخلايا العصبية المحيرة كان ذي نتائج أكثر دقة في تمييز الأرقام العربية المكتوبة بخط اليد من نظام شبكة الخلايا العصبية الاصطناعية.