RECOGNITION OF HANDWRITTEN ARABIC CHARACTERS VIA NEURAL NETWORKS

BY
Muhammad Aref Muhammad Al-Shridah

[illegible]

Supervisor
Asst. Prof. Dr. Ahmad Sharieh

[illegible]

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science

Faculty of Graduate Studies
University of Jordan

January 2000

This thesis was successfully defended and approved on 2/1/2000.

Examination committee	Signature
Dr. Ahmad Sharieh, chairman Asst. Prof. of Parallel Processing	[illegible]
Dr. Ahmed Al-Jaber, member Assoc. Prof. of Algorithm Analysis	[illegible]
Dr. Khalil Al-Hindi, member Asst. Prof. of Artificial Intelligence	[illegible]
Dr. Mohammed Bilal Al-Zoubi, member Asst. Prof. of Graphics and Pattern Recognition	[illegible]
Prof. Saleh Irsheid Oqeili, member Prof. Of Analysis and Design of Structured Methods for Increasing Computer Memory Reliability	[illegible]

To my parents

To my wife

To my sons

ACKNOWLDGEMNETS

I thank Dr. Ahmad Sharieh, my supervisor, for providing me with different facilities and resources that make this thesis possible. I thank all members of computer science department at the university of Jordan, especially Dr. Ahmed Al-Jaber, Dr. Khalil Al-Hindi, Dr. Munib Qutiashat, Dr. Mohammed Bilal Al-Zoubi. I thank Prof. Saleh Oqeili in Amman University.

I would like to thank my best friends Amjad Hdaib for his help during my search and Issa Qunbor.

Special thanks to my special friend Fadi Al-wedian for his helping and the Jordan Armed Forces (Headquarter) for their support. And the last but not least to my family, specially my wife Tagreed Ayideh for her support, understanding and tolerance.

CONTENTS

LISTS OF TABLES

LIST OF FIGURES

Abstract

Arabic character recognition has attracted relatively less attention compared to the great programs in recognition techniques of other languages such as Latin, Chines, and Japanese. The segmentation of Arabic words into components is a difficult task, especially for handwritten text. There are no clear rules that can be applied, Arabic characters can take many sizes and shapes, and they can be overlaid, increasing the complexity of the segmentation. The importance of this work stems from the fact that such a research is different from other writings, and the great variations of handwriting styles that exist among individuals.

The main domain of this thesis is the off-line recognition of handwritten Arabic characters. In this research, a fuzzy ISODATA for finding skeletons of Handwriting Arabic words are used. This algorithm has been used in earlier study but suffers from two major problems. The first is choosing an initial number of clusters, and the second is the algorithm's high processing time. In this research, adding thinning step before the algorithm is applied solves the exestuation time problem. This has reduced the processing time to about 25% of its original value. Moreover, the thinning step improves the performance of the algorithm because the produced skeleton represents the clusters of the pattern's medial axis. The second problem is solved by suggesting the choice of an initial number of clusters that depends on the size of the words, then the skeleton can be refined by deleting the extra insignificant vertices.

Neural networks have been widely and successfully used to perform different tasks in pattern recognition. In this work a multiple classifier type of neural networks is developed. The neural networks were used in classification the handwritten Arabic characters. The NNs were trained and testing in data sets before and after pre-processing.

The experimental results of applying the NN on handwritten Arabic text to recognize a 28 letters were very good. The average accuracy rate reached up to 89% in segmentation phase. The average accuracy rates range from 89 to 97% for training sets and 75 to 83% for testing sets when applied in classification phase. The results are better than others results from research using NN to recognize 28 Arabic characters, and less than some other using other methods rather than NNs with RPROP Algorithm. In conclusion, it is advisable to use the proposed techniques and the built NN to recognize handwritten Arabic characters and similar natural languages, in most of pattern recognition phases.

Chapter 1

Introduction

Character recognition is a trivial task for human but to make a computer program that does character recognition is extremely difficult.

This research presents skeleton representation, segmentation and classification stages in a system, using Neural Networks, to recognize letters in handwritten Arabic text.

1.1 Introduction

The invention of machines that imitate human functions has captured the attention of many scientists and researchers in different disciplines. Continuous efforts have been exerted to extend the role of computers from data processing to more complex tasks of perceiving the environment. One of these tasks is character and handwritten recognition, which is required to imitate the human reading capability. Every time, man reads a book, magazine, letter, check, or newspaper, he is in the process of recognizing alphanumeric characters and words. Such a seemingly trivial task becomes extremely difficult when one tries to use a computer to do it. How can humans learn, understand, memorize and recognize these many different environments? How can we make machines or computers do it? Such secrets remain unknown, but they are beginning to gradually unfold (we hope!).

An immense effort has been spent on character recognition since 1930 (Yamany, 1995), because the needs in business and scientific applications. At present, the most important use for character recognition systems is directed towards office automation, which dominates information processing. Intensive research has made optical character recognition (OCR) an efficient means of entering data directly into the computer and capturing information from books, sheets, and other handprint or machine printed materials. Three main application areas can distinguish the current status of character recognition: text entry in office automation, data entry, and process automation.

In the first application area, automatic entry of text or data is an alternative to cumbersome keyboard entry. While in the data entry area such as banking applications, there are constrained paper formats, a limited character set, and high throughput requirements. Automatic address reading has been applied in post office process automation. For postal mail sorting. A typical application is the automatic sorting of envelopes with totally unconstrained handwritten numeric postal ZIP codes.

Recognition of Arabic characters represents an important goal, not only for Arabic speaking countries, but also for Curds, Persians, and Aurdo-speaking Indians (El-Melegy, 1996). However, in spite of the progress of machine character recognition techniques (both printed and handwritten) of Latin, Chines, and Japanese characters, research in Arabic character recognition has been slowly gaining momentum since the early 1980's (Amin, 1982). The main reason for such a delay is the different characteristics of Arabic writings from other writings, and results in techniques developed for other writings. This also results in that the recognition of Arabic characters. Moreover, most work in this field has been directed to machine printed characters, whereas little work has been devoted to handwritten characters. This is attributed to the fact that problems encounter in handwriting recognition are generally greater.

Problem Definition

The problem of concern in this thesis is to recognize the handwritten Arabic character using Neural Networks techniques. An optical scanner is used to transform spatially continuos 2-D documents of handwritten words or characters to a digital image. The image is saved as a file of Bitmap format. A computer program processes this file, recognizing the written characters, and transforms them into symbolic representation that can be processed later by the computer.

The difficulty of this task stems from numerous problems.

1- problems due to data acquisition:

During the scanning of input documents, noise is imposed in the images. Character may expose to deletion, erosion, translation, skewness, and rotation.

2- problems due to individuals:

The difficulty of recognizing natural handwriting comes from the inherent ambiguity of some handwritten characters and the human inaccuracy during writing. This is aggravated by the fact that handwriting is a mean of communication between people who usually share the same language. Therefore, writers assume readers will be to guess missing or distorted characters from the context. This can be viewed in extra strokes at the start and end of the main strokes, incorrect positioning of points and dots, and different shapes of the same charter for the same writer.

3- problems due to the language itself:

The characteristic of the Arabic writing adds to difficulty of recognition. Arabic writing is of cursive nature, which necessitates the separation of the connected characters for recognition. The language heavily employs complementary strokes (secondary) such as dots and hamza("ء",".") . Arabic letters generally have 2-4 shapes according to their position within a word.

Thesis Objectives

The research in the field of off-line recognition of Arabic characters has not achieved significant results, much work is still required. Hence, the objectives of this thesis can be stated in the following points:

1- Reviewing the previous work done in the field of Arabic characters recognition.

2- Investigating the different difficulties encountered when trying to recognize handwritten Arabic characters.

3- Introducing approaches using neural networks for segmentation and classification the handwritten Arabic characters.

Manuscript Structure

In this thesis , Chapter 1 introduces the concept of pattern recognition and the pattern recognition methods in general. Chapter 2 describes the Arabic languages and its letters and the main characteristics of the Arabic languages and the main problem encounter the researchers in recognition the handwritten Arabic characters, different approach of the most previously works done in the recognition and segmentation of Arabic languages, the last section of Chapter 4 illustrates the Neural Networks and the major type of Backpropagation, and the algorithm of Neural Networks used in segmentation and recognition. Chapter 3 contains the preprocessing of the characters, data acquisition, scanning, and smoothing using famous algorithms in this field. The algorithm for secondary components recognizing is illustrated and the major feature of the skeleton are extracted to be entered to the neural network for segmentation section 2 and section 3 illustrate the constructing of Neural Networks for segmentation. Chapter 4 contains the results and discussion of the segmentation and classification stage. Finally chapter 5 contains summary and conclusion of this research.

1. 2 Pattern recognition

Pattern recognition is the research area that studies the operation and design of systems that recognize patterns in data. It encloses subdisciplines like discriminate analysis, feature extraction, error estimation, cluster analysis (together sometimes-called statistical pattern recognition), grammatical inference and parsing (sometimes called syntactical pattern recognition). Important application areas are image analysis, character recognition, speech analysis, man and machine diagnostics, person identification and industrial inspection.(Yong,1997)

Character Recognition

During the past 50 years, substantial research efforts have been devoted to character recognition (CR) which is used to translate human-readable characters to machine-readable codes. The problem of CR can be generally stated as the problem of transforming two-dimensional spatial form of writing to symbolic representation. The demand for machine recognition of characters increases as the volume of information in daily life increases, and consequently the more the need for man-machine communications. For the reason, automatic CR systems have been developed both for printed and handwritten characters. CR is more widely known as optical character recognition (OCR) because it deals with the recognition of optically processed characters. The ease with which humans recognize and read characters (however according to (Suen et al,1997), even humans have an error rate of about 4% when reading handprinting in the absence of context) often leads to the incorrect assumption that this capability is easy to automate.

1.3 Trend of Optical Character recognition

The optical character recognition system is classified as on-line and off-line system.

1- On-line character recognition systems capture the temporal (dynamic) information of the writing such as the number of pen-down, the order of pen-strokes, the direction of writing for each pen-stroke and speed. It is interactive, so that the user can adapt his/her writing to the system in order to improve recognition. However, it requires special equipment uncomfortable and unnatural to human. It cannot be applied to printed or handwritten documents.

2- Off-line character recognition (OCR) systems capture data through optical scanners or cameras into a bitmap pattern. They allow previously written or printed text to be processed and recognized. However ,these systems require costly and imperfect pre-processing techniques and they lack the temporal information of on-line system (Atic ,1994,Eric,1997).

1.4 Pattern Recognition Concepts

There are three concepts that pattern recognition depend on,Template Matching or Membership Roster, Common Feature Concepts and Clustering Concept.

1.4.1 Template matching

A matching class can be identified through one of the pattern that it belongs to. In template matching, a group of classes are stored where each one represents one of the identified classes. An entered pattern is compared with these classes according to a pre-defined scales. For example, if the entered pattern matches class I better than any other classes, then this entered pattern is classified as class I.

Disadvantage of applying this concept is difficulty of choosing the right scales for matching. And it gets very difficult with the changes on the same patterns that belongs to the same class and also with the harm that could affect the entered pattern. Actually ,it is very hard to find a true pattern recognition system that depends wholly on this concept (Trier and Taxt, 1996).

1.4.2 Common Feature Concept

The main assumption that this concept is built upon is that all the patterns that belong to a specific class share among them a group of features. These feature reflect the resemblance between them, eventually distinguish them from each other (Al-Taani, 1994). That necessarily implies find a definition and features through processing the patterns.

Every common feature for every class is stored. When an unidentified pattern is entered, the features of this pattern is extracted and compared with the features of the stored classes. Then, the pattern is classified within the class that it shared with the largest number of features.

The difficulty of this concept lies in finding the common features that can be observed from the designer by studying a group of pre-defined classes (Al-Taani, 1994).

1.4.3 Clustering Concept

When a pattern belongs to a specific pattern is represented in a vector with real numbers, the class can be recognized in its ability to form a cluster in a pattern space. Establishing a recognition system based on this concept depends heavily on the geometric arrangement for the different classes clustering. If expressing those classes were possible through clear separate clustering, then using a simple classifier technique such as

Minimum-distance Classifier would be sufficient. If those clustered were interlaced with no clear boundaries between them, it becomes necessary to use more complicated techniques such as the trained classifier (Al-Taani, 1994).

1.5 Pattern Recognition Methods

The basic concepts in pattern recognition can be applied by using five major methods: heuristic, statistical , linguistic or syntactic , neural networks and the last but not least the combined methods (Al-Taani,1994).

1.5.1 Heuristic Methods

This method depends on the efficiency and the experience of the designer and it uses Template Matching and the Common Features Concept. The designed system comprises mostly from a group of developed procedures to make a definite and specified recognition and being such doesn't give general solutions. Mostly, it provides a solution that goes with the problem which has been developed for. The heuristic methods undoubtedly are among the most important methods in pattern recognition. Although they depend on designed experience (Bishop, 1995).

1.5.2 Statistical Methods

In this group of methods, a set of features from the entered pattern is extracted and then classification is performed throughout partitioning the features space. And after the patterns is expressed as a dot in the feature space, the pattern features is represented (Pao, 1989).

The statistical methods could be divided into parametric methods, non-parametric methods and serial methods. Also the statistical methods may depend on the clustering analytical concept or fuzzy sets theory. The application of pattern recognition through the statistical methods are the character recognition, medical diagnosis, speech recognition, automatic inspection ... etc (Al-Taani, 1994).

1.5.3 Linguistic or Syntactic Methods.

As mentioned above in the statistical methods the information concerning the pattern structure either neglected or vaguely represented. In some pattern structure problems, the structure of relations that describe the pattern is playing an important role in recognizing it, and the recognition process might include description for the features that made this pattern belongs to a specific class. One of the ideal examples for this kind of recognition is the scene analysis. Using the Linguistic methods considered a good choice for this kind of problems.

The linguistic methods use the natural linguistic theory. The components of the linguistic pattern recognition resemble to a great deal the components of the language translators. The entered pattern is represented through a set of primitive components that the researcher has to find. That group of primitives makes what is called a string in the linguistic theory. Then it is subjected to, a grammar system that classified the pattern to the language it belongs to and in which represents one of the classes (Al-Taani, 1994).

1.5.4 Neural Network

This can be considered as a function between the input space (I) and the output space (O). Mathematically, the Neural Network is represented as a function from (I) to (O);

$$F: y = f(x), \quad \text{where } x \in I \text{ and } y \in O$$

Since the classification process is a function between the pattern space and the classes group, we can identify the Neural Network as a classifier.

The Neural Network is suitable in pattern recognition than the traditional methods(Bishop,1995). Pattern recognition requires the ability to match a great deal of inputs and then extracting a general output or a class. It requires the ability to deal with incomplete input. The Neural Network provides these capabilities along with the learning capability. The neural network provides the possibility to attain a special structure suits the handwritten recognition problem (Bishop, 1995).

1.5.5 The Combined Methods

It is possible that the recognition system contains several classifiers that might not all belong to the same method and then all results will be merged according to one of the existing methods. The purpose of such thing is to benefit from all the strong points in each method and then getting the best results. One of the traditional merging methods is 'Majority Voting' where the classification that agrees with more than half the classifiers will be resulted, and when the agreement is not achieved the result is considered wrong (Cho and Kim, 1995). There is also the 'Borda Count', which is the totals of the number of classes under one classifier and for the class with the largest number (Cho and Kim, 1995)

An example on the modern methods in building a recognition system with multi classifiers is the Neural Network. In this, a special Neural Network will choose the best classification result (Huang et al, 1995). Cho and Kim (1995) also use the fuzzy ISODATA to merge the result of a classification system made of several Neural Network classifications. This method gives a special important role for each classifier according to the qualification, so that the most qualified classifier will participate in the final decision. The combined methods give a great strength to the recognition system. But we keep it for the problem that can't be solved with simple methods. It is only natural to seek the solution among the simple methods before trying to apply complicated methods.

1.6 Categories in Character Recognition

The major categories found in character recognition ,arranged in ascending order of difficulty ,are as follows .

1. Fixed- Font character recognition: Recognition of a specific of machine –printed font
2. Multi-font character recognition :Recognition of more than one machine printed font.
3. Handprint character recognition: Recognition of handwritten characters, which are unconnected or not written in calligraphy.
4. Cursive character recognition: Recognition of handwritten characters, which may be connected.

Problems encountered in handwritten characters as in 3 and 4 are greater than in printed characters due to the infinite variations of character shapes resulting from the writing habit, style, education level, mood and the health of writer. However, handprint character recognition is simpler than cursive recognition due to absence or near absence of segmentation problem. In order to reduce shape variation, so as to achieve less confusion and to obtain higher recognition rates, constraints may be imposed on handwritten CR systems. As the number constraints is reduced , the system becomes more flexible but more difficult.

Chapter 2

Literature Review

2.1 Characteristics of Handwritten Arabic

Several methods have already been developed for the recognition of Latin and many other character sets (Matsui et al, 1996,Mori et al, 1992). However, intrinsic differences between the writing styles of Arabic and other languages do not allow a direct application of these algorithms to Handwritten Arabic. The key to high recognition performance of handwritten characters of any languages is the ability to detect and utilize the distinctive characteristics of the characters of the languages. Arabic handwritings have their own characteristics, which pose difficulties in designing a general system for the recognition of unconstrained handwritten texts. These characteristics should be first studied.

This chapter consists of two parts. The first part is devoted to language description, the Arabic character sets, and some popular fonts. The second part presents the primitive strokes from which all Arabic characters can be built. The following section introduces the problems associated with handwritten recognition systems of Arabic texts, from primarily a pattern recognition point of view.

Table 2.1: Comparison of various languages, where R(right) and L(left).

Characteristics	Arabic	Persian	English	Hebrew	Hindi
Number of letters	28	32	26	22	40
Cursive	Yes	Yes	No	No	Yes
Justification	R-to-L	R-to-L	L-to-R	R-to-L	L-to-R
Possible	1-4	1-4	2	1	1
Diacritics	Yes	Yes	No	No	Yes

2.1.1 Languages Description

Arabic language is similar to English in that they use letters, numerals, punctuation marks, as well as spaces and special symbols for mathematical expressions. However , it differs from English in its character set and writing direction. In addition , the structure of Arabic characters consists of curves and line segments, and some characters contain one or two closed loops in their body. Table 2.1 shows a comparison of the characteristics of various languages. Diacritics in this table are the marks, which are sometimes added to a letter to indicate a special pronunciation.

Character Sets of Arabic

There are 28 characters in the Arabic character set shown in Table 2.2. This does not mean, however, that the character sets are only 28 unique shapes. The reason is that there are number of characters with the same body but they differ in the number of dots and their positions. Furthermore, although there is no upper or lower case characters in Arabic languages. There exist different shapes for some characters, depending on their position in a word. Some characters have up to four different shapes.

Arabic language uses different fonts. Texts written from right to left and numerals are written from left to right. Arabic language uses cursive writing, which implies that the boundaries of characters in a word can easily overlap. Cursive words are separated by spaces, and some of the characters can only appear at the beginning or at the end of the word. However, some characters are not connectable from the left side with the succeeding characters. Consequently, a word may also be divided into one or more subwords. A subword is either a single isolated character or a combination of two or more connected characters. Figure 2.1 shows words with one, two, three, and four subwords, respectively. The first word in (a) consists of six connected characters while the one in (d) has four isolated characters.

فلسطين	أمين	طهران	آزرو
(a)	(b)	(c)	(d)

Figure 2.1: Arabic words with : (a) one subword (Palastain), (b) two subwords (Amin) ,(c) three subwords (Tehran), and (d) four subwords (Azero).

Table 2.2: Arabic alphabet with different shapes of characters upon their position in a word.

Name	Isolated	End	Middle	Beginning
Alef	ا	ـا	-	-
Ba	ب	ـب	ـبـ	بـ
Ta	ت	ـت	ـتـ	تـ
Tha	ث	ـث	ـثـ	ثـ
Jeem	ج	ـج	ـجـ	جـ
Heh	ح	ـح	ـحـ	حـ
Kha	خ	ـخ	ـخـ	خـ
Dal	د	ـد	-	-
Zal	ذ	ـذ	-	-
Ra	ر	ـر	-	-
Za	ز	ـز	-	-
Seen	س	ـس	ـسـ	سـ
Sheen	ش	ـش	ـشـ	شـ
Sad	ص	ـص	ـصـ	صـ
Zad	ض	ـض	ـضـ	ضـ
Tta	ط	ـط	ـطـ	طـ
Dha	ظ	ـظ	ـظـ	ظـ
Ain	ع	ـع	ـعـ	عـ
Gain	غ	ـغ	ـغـ	غـ
Fa	ف	ـف	ـفـ	فـ
Ghaf	ق	ـق	ـقـ	قـ
Kaf	ك	ـك	ـكـ	كـ
Lam	ل	ـل	ـلـ	لـ
Meem	م	ـم	ـمـ	مـ
Noon	ن	ـن	ـنـ	نـ
Waw	و	ـو	-	-
Ha	ه	ـه	ـهـ	هـ
Ya	ي	ـي	ـيـ	يـ

Printed and Handwritten Fonts

There are many different fonts used in Arabic handwriting texts .Some of these fonts are also used in printed texts as well .Only a few of these are popular and are used by people in their normal everyday handwriting . Some fonts differ in the shapes of some of the characters, but other fonts differ in the combination of connected characters. Sometimes those characters are not connectable from the left. In addition, sizes of characters are different for different fonts, and the characters in some fonts include a hook-like curvature at the end or at the beginning of a word. Table 2-3 shows ranking of some popular fonts .The ranking in this table is extracted from (Damrah , 1985), and is accepted by calligraphy experts. Among these fonts, Naskh is the most popular font in machine printed documents. In the next few paragraphs, let us introduce Kufi and Naskh fonts.

Kufi

It is known as the first official font of writing in Islamic texts. Figure 2.2 shows an example of Kufi fonts. Thus, this font was invented before the origination of Islam. The characters use more straight vertical lines than any other font. This font is widely used in Islamic architecture and buildings. It has different versions grouped as simple, medium and decorative.

Table 2.3 :Ranking of Arabic fonts.

	Popularity	Chronological order	Reading simplicity	Writing simplicity
Naskh	8	1	7	6
Thuluth	4	2	5	4
Kufi	2	3	1	3
Diwani	6	4	6	5
Broken	5	5	4	1
Roqa	3	6	4	1

من وصايا علي بن ابي طالب كرم الله وجه

يا بني احفظ عني اربعا واربعا لايضرك ما عملت معهن

ان أغنى الغنى العقل واكثر الفقر الحمق واوحش الوحشة

العجب واكرم الحسب حسن الخلق .

Figure 2.2 Arabic text written in Kufi font.

Naskh

This font is the most frequently used font in printed documents and the second most frequently used font in handwritten text. Figure 2.3 shows an example of Naskh font. Invented in the seventh century by ibn Mogleh by inspiration from the Kufi font (Damrah, 1985). It is one of the simplest fonts in handwritings. There are two versions of these fonts in Arabic. The Arabic version of these font is called Yaghooti and the Persian one is called Neyrizi (Damrah, 1985). This font is extracted from the old Kufi font, but because it changed some of the rules of Kufi font, it was called Naskh, which is translated as "abolition" in English (Damrah, 1985).

عالم وجاهل

أخو العلم حي خالد بعد موته

واوصاله تحت التراب رميم

وذو الجهل ميت وهو ماشي على الثرى

يظن من الاحياء وهو عديم

Figure 2.3 :Arabic text written in Naskh font.

2.1.2 Problems of Handwritten Recognition

In this section, we present the characteristics of Arabic text from a pattern recognition points of view. To design a method for recognition of the patterns ,we should first understand the distinctive characteristics of the pattern space. This will help us determine the intrinsic problems associated with designing a high performance recognition system. We will look at the characteristics and difficulties of a recognition system for handwritten Arabic texts.

Segmentation Problem

One of the most prominent difficulties in Arabic character recognition systems is the segmentation process. Segmentation is the process of separating a line of text into words and subwords, and then dividing the subwords into characters. Due to the cursive nature of Arabic writings, segmentation is a very difficult task, even in printed texts.

Handwriting Types

As described in (Habib,1997), handwriting in Latin language and English can be characterized into five categories:

1- Boxed discrete characters,

2- Spaced discrete characters,

3- Run-on discretely written characters,

4- Pure cursive script writing, and

5- Mixed cursive script writing.

These categories are listed in the order of increasing difficulty of recognition. Figure 2.4 shows different types of handwritten texts in Latin language. In Arabic , there are boxed discrete characters and mixed cursive script writing. As the segmentation can be very ambiguous, cursive script writing requires more complicated segmentation methods. One possible solution to these problems is intersection of segmentation with recognition.

1:

B	O	X	E	D		D	I	S	C	R	E	T	E		C	H	A

2: S p a c e d   d i s c r e t e   c h a r a c t e r s

3: Run-on discretely written characters

4: Pure cursive script writing

5: Mixed cursive and discrete

Figure 2.4: Different types of handwriting.

Overlap in Handwritten texts

Horizontal spaces between the subwords or between discrete characters are of the great help in segmenting them into their constructing parts. This is usually accomplished by using a vertical projection of text line . Those points with minimum value for the projection are candidates for vertically cutting and segmenting. However, there are cases in which vertical cut causes a character to be divided into two different segments.

This usually occurs when two or more characters are vertically overlapping. In Handwritten Arabic texts, there are different types of overlap between characters. They can be three classes: overlap without touching, overlap with touching, and overlap caused by unusual touching.

Overlap without touching: in this, subwords are vertically overlapping but they are not connected. Parts of one subword may be vertically aligned with another character or a group of characters in the neighboring subword. This is marked as type "a" in Figure 2.5. In this Figure, a letter in a circle connected to a vertical line shows the location and the type of overlap.

Overlap with touching: a character is usually connected to its succeeding character by touching an endpoint at the right side of the character to a left endpoint of its neighboring character. In some fonts like Naskh, a character may connect to its succeeding character from the top (marked as type "b" in Figure 2.5). In this case, the two characters align vertically.

Overlap caused by unusual touching: This often occurs in broken font. In this case, some characters not connectable from the left or right side are connected (marked as c in Figure 2.5). All of these types of connections are very common in normal handwritings. None of them is vertical projection. A base-line approach for segmentation shows that projection and base-line approach segmentation is applicable. Figure 2.6 shows an example of using vertical projection. The Figure shows that projection method does not provide adequate information for segmenting the words into their building character. For a survey on segmentation methods in handwritten texts refer to (Casey and Lecolinet,1995).

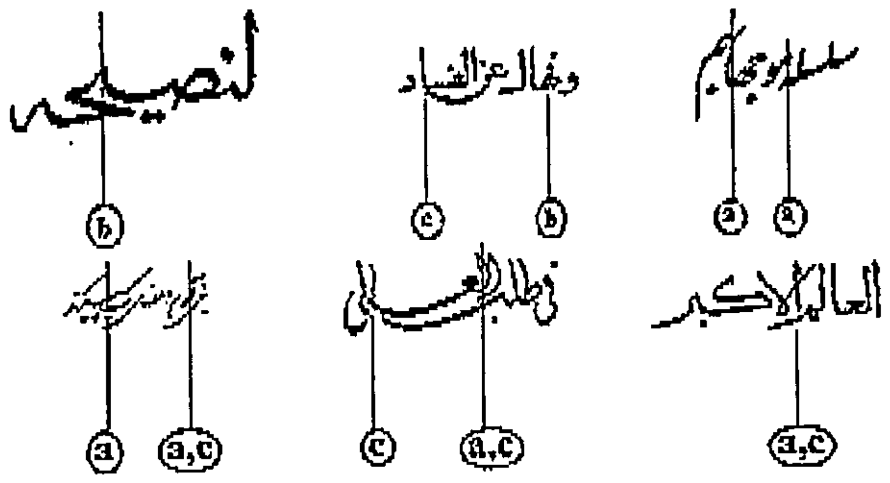


Figure 2.5 Different types of overlap between characters: "a" denotes vertical without touching, "b" vertical with touching and "c" unusual touching.

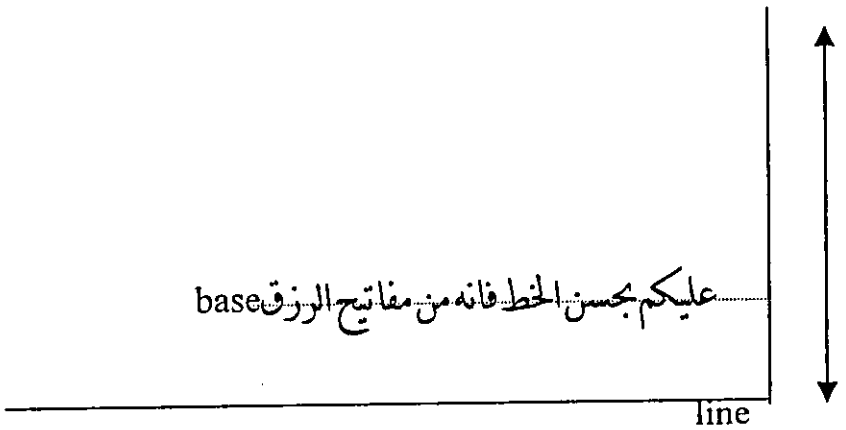


Figure 2.6 vertical projection of a line of Arabic text. This method is not applicable in Arabic segmentation.

Character Primitives

People learn to combine strokes to build each character at an early age. We define primitive strokes as the straight lines and simple curves and corners that make up all the entire character set. Each character can be built by a combination of one or more of these primitives stoke. The primitive strokes of characters of Naskh font are shown in Figure 2.7.

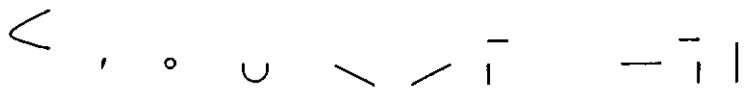


Figure 2.7: Primitive strokes of Arabic Naskh font.

Number of Classes

Because Arabic characters have more than one shape, the actual number of patterns to recognize is not the same as the number of characters. In addition, dots and diacritics are considered as complimentary in the characters, hence those characters, which differ only in dots, and diacritics have almost the same patterns. Dots can be segmented and recognized by a different system, and the output of this system at the post-processing stage. Thus the actual number of patterns to be recognized is always different from the number of characters in a font.

Handwriting Variability

Handwriting is a free form process. There are an infinite number of ways of writing a word. No one can write his or her own name exactly the same way twice in the entire lifetime. Thus, every person has a range of handwriting variations determined by different factors including physical ability, illness, medication, drug or alcohol use, stress, the writing surface, the writing instrument, attempted disguise, and personal pretences.

Handwriting characters come in two categories: general or class characteristics, and individual characteristics. Depending on the cultural setting (time and place) when writing is learned, entire groups of individual are taught to write in the same way. When these individuals are first learning to write, there are difference s in their ability to perform the task, and the results are not all the same, but the true individual writing style difference appear only over time. As we grow physically and mentally, our handwriting becomes more of an individual product through conscious changes made to fit a mental picture of how we want our writing to appear . This may even be an unconscious process to some extent.

Confusion of Similar Characters

One of the most important limits for achieving a high recognition rate for handwritten Arabic characters is confusion between the very similar characters. As it is mentioned before, there are groups of similar characters, which only differ in position, and number of dots like JEEM,HEH and KHA in Table 2.2; the character body is the same. However, there are other ways by which two or more handwritten characters become similar. We divide the problem of similarity of characters into five categories : similar shapes, similar when rotated, similar when scaled, and similar because of the writing styles.

Similar Shapes: This is a group of characters with the same body shape. Regardless of what type of feature extraction technique we use for this character, we always have a large probability of confusion. Figure 2.8 shows some groups of similar isolated characters of Naskh font.

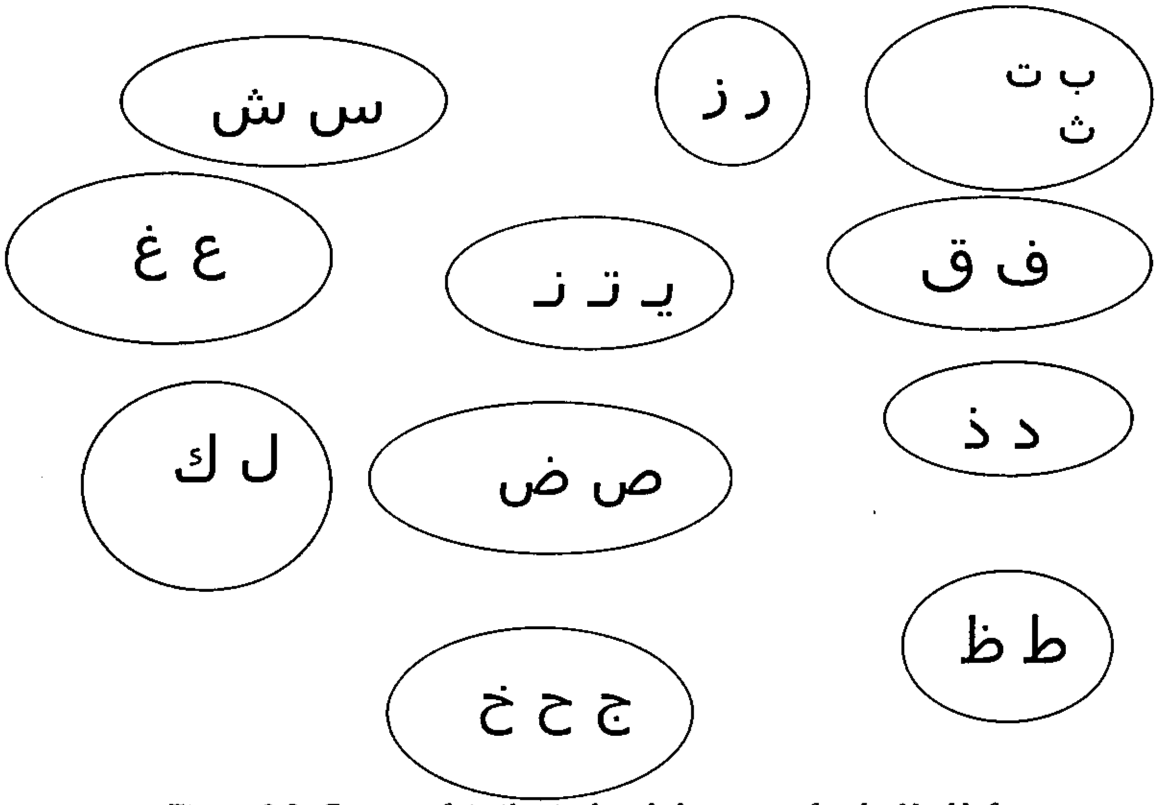


Figure 2.8: Groups of similar isolated characters for the Naskh font.

Similar when rotated: A body of a character becomes similar when one another is rotated. For example, if the character "MEEM (beginning)": "مـ" is rotated 90 anti-clockwise, it becomes very similar to the characters "HAA (end)": "ـه" ,see Table 2.2. For these characters, any extracted feature from rotation invariant causes them to be confused.

Similar when scaled: For these characters, the scaled version of one becomes similar to another. For example, the enlarged version of the character "BEH (beginning)": is very similar to the normal "LEH (beginning)" (Table 2.2). Scale invariant features produce almost the same feature vectors for this group of characters.

Similar because of writing styles: Variability in handwriting and mixture of fonts can cause some characters to become similar.

Similar feature vectors: Depending on the method of feature extraction, some characters may have the same feature vectors.

Mixture of Fonts

Most people mix different fonts in their normal handwritings. There are no rules to define how and when fonts are mixed, and it depends very much on every person's style of writing. Some people may even modify the original shape of a character in a font set. Mixture of fonts increases the number of patterns to be recognized. It also increases the probability of confusion by increasing the number of similar and ambiguous patterns.

Problems of Dots and Diacritics

Diacritics are the marks added to a letter to indicate a special pronunciation. Another problem with handwritten Arabic documents is that of dots and their locations. Many Arabic characters have a number of dots. There are different numbers of dots, which are located in different positions within the character (see Table 2.2). Dots are considered as complimentary character (Al-sadon and Amin, 1995). Any version or deletion of these complimentary character results in a misrepresentation of the whole character. This is especially important in any preprocessing such as thinning or segmentation process. A thinning preprocessor should take great care of dots so as not to change the identity of the character. The major difficulties with recognition of dots in handwritten documents are misplacement and their shapes. In handwritten texts, dots can easily be misplaced . In some cases, it becomes difficult to tell if a dot belongs to a certain character or its neighboring characters. Human readers use other clause such as context to recognize the actual location of a dot or a group of dots. Changes in shape lead to different shapes for dots. For example , some people use only an incomplete circle to represent three dots. Some use a straight line as two dots. Figure 2.9 shows a word using letters "SHEEN": "ش" with three dots, but of different shapes.

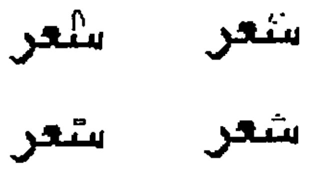


Figure 2.9: Arabic word using the character "SHEEN" with different shapes for its three dots.

Lack of Handwritten Data

One of the main problems of Handwritten Arabic recognition system is that there aren't standard data sets to evaluate new developed algorithms. All research done is based on the character sets selected and collected by algorithm developers. This means that it is not possible to compare these algorithms, and so we need to have a standard data set. Designing such a collection of patterns which covers all the possible combination of characters is, however, very difficult.

The main problem is that for even a single character, there are many different patterns.

To design a collection of handwritten patterns, one should first answer some questions (Safabakhsh and shaygan, 1995):

1- Should the data set include segmented isolated or words?

2- What is the minimum number of words that cover all the possible practical patterns?

3- What are the criteria for readability of patterns and what are the conditions for which a pattern should be rejected?

2.2 Trends in Arabic Characters Recognition

2.2.1 Research Directions

Many OCR systems have been developed over the last two decades , but more work is still required to attain results close to human recognition abilities. This is true for the recognition of unconstrained handwritten documents in which every individual has his/her own writing style. There exists a large number of techniques for features extraction and classification of both handwritten and printed texts (Suen , 1982); however, no simple scheme is likely to solve all the problems associated with largely variable input data in handwritings. Two groups of methods have been often used: structural and statistical approaches. Structural features often result in a better performance than statistical features, but they may be difficult to define and they may be sensitive to data sets(Habib,1997). Thus, having a high performance in one data set does not necessarily mean that the method will give the same performance for other different data sets (Suen and Lam ,1993).

It is often difficult to compare different approaches in handwritten character recognition, as they are generally based on different databases. However, neural networks have shown the best performance among all the different methods used for character recognition (Guyon, 1991). The most remarkable feature of artificial neural networks lies in their ability to learn by examples. Due to the ability of neural networks to overcome some differences of conventional pattern recognition techniques, the applications of different types of neural networks in the area of handwritten character recognition have been increased. For a comparison of statistical and neural classification techniques for recognition of handwritten numerals, see (Cao et al, 1992).

Variability of handwriting introduces a kind of fuzziness in handwritten recognition systems. Neural networks and fuzzy logic are complementary tools to deal with the same problem (Habib , 1997). Although Hidden Markov Model (HMM) has been widely used, especially for on-line cursive recognition as well as speech recognition (Habib, 1997). Some researchers applied the HMM technique for off-line systems, mostly in word recognition application. The HMM was used for representation of printed characters in noisy document images and for handwritten recognition (He et al,1994). Neural networks can not only be used as excellent feature extractor and classifier, but also as trainable and good classifier combines .

2.2.2 Segmentation of Arabic Characters

In all cursive character recognition systems, segmentation problem considered the most challenging problem facing researchers in this area. Segmentation means separation of objects or regions of a digitized image. When applied to the task of character recognition, segmentation is the isolation of various writing units, such as characters or strokes, of a word. If segmentation is done prior to recognition, it is called external segmentation (segmentation-then recognition approach). Because several characters can be written with one stroke in a cursive word, some recognition can be used for their isolation. Segmentation requiring recognition is called internal segmentation (recognition then segmentation approach)(Fujisaki et al,1991).

Perhaps the earliest means of segmentation was an explicit signal from the user in on-line systems, or the requirement of writing in predefined boxes (on-line and off-line systems). Since then, researchers using various successfully. For example, the presence of dots distorts the histogram, and may lead to segmentation errors.

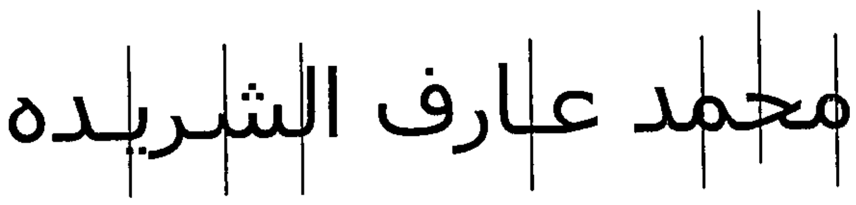


Figure 2.10:Segmentation of three Arabic words using vertical density histogram.

Segmentation by Contour Following

In this segmentation technique (Guindi, 1987) the word closed contour is obtained. The distance between the two extreme points on the contour that intersect a vertical line at every column is calculated. A transition between Arabic characters occurs if the calculated distance falls below a certain threshold. The start and end of a character are marked on the contour by an algorithm. The use of contour following in this approach can solve the overlapping problem.

Segmentation Based on Contour analysis and Baseline Location

This method is introduced in (Allam, 1995). After tracing the edge contour, analysis provides information about finding the location of the base line with the help of histograms. After finding the base line of any given Arabic portion, the portion is examined column by column. Whenever, there is a transaction from a column having all the black pixels inside the base line to a column having black pixels outside the base line, a segmentation point is marked. An error in locating the base line would give erroneous segmentation points. This method is suitable for printed characters. It is difficult to be applied to the segmentation of handwritten Arabic characters.

Segmentation by Extraction of Strokes

An algorithm proposed by Almuallim and Yamguchi, (1987), the world is thinned first, then, the word is divided into a number of strokes. A stoke is defined, in this approach, as a continuos curve represented by a string of coordinates. The order of these coordinates represents an approximation of the pen movement during writing. A stroke can be extracted by finding out its start point and then following the curve until reaching an end point. However, the thinning process would cause some problems to which the author has not referred.

Segmentation by Neural Networks

In (Abdulmageed, 1994), contour analysis is performed using neural networks to determine a preliminary set of segmentation points. For every point on the contour, the directional codes of the contour in a defined neighborhood around this point are employed as input to a multilayer feedforward neural network. The net decides whether the point is a segmentation point or not. An optimization problem, then, is formulated to select a subset of the obtained points as actual segmentation points. This technique is of great theoretical value, but would not be accurate when it is applied to text of different fonts. In addition, relying on a feedforward neural network trained with the Backpropagation algorithm to suggest segmentation points makes the system incapable of easily learning new possibilities of segmentation points.

Segmentation Based on an Event-Follower Strategy

This approach proposed in (Abdelhamid, 1995) using a set of event. An event is a zone consecutive bit. A number of rules to follow the events are used to split the word into characters. This method is heavily font dependent.

Post-Segmentation

In this method (Ramsis et al, 1988), characters are segmented after they are recognized (internal segmentation). The idea is to extract sequentially, accumulative invariant moments as features while moving from right to left along an Arabic word. The moments are calculated and used to classify this portion. If mismatch occurs, a new column is added and the process is repeated until character is recognized. This method is time-consuming, sensitive to noise and heavily font-dependent.

Comments on the Previous Segmentation Techniques:

1-Most of the previous approaches depend on the use of histograms. Moreover, if an approach, mainly, depends on using a threshold, then failing to determine a suitable threshold would yield erroneous results.

2-Most of the above mentioned techniques yield character primitives rather than characters. Then, a recognition algorithm is usually required to recombine the primitives into characters. Splitting an Arabic word into primitives is generally suitable to the segmentation problem than separating a word into characters.

3-The approaches followed in (Abdlelazim, 1985, Abdlelazim, and Hashish, 1989, Abdlelazim, and hashish , 1990) fail to overcome the problem of the overlapping of some characters. Concerning this particular point, contour following based approaches would provide better results.

4-All the above techniques are applied to the segmentation of Arabic printed text (excluding (Almuallim and Yamguchi, 1987,Al-Yousefi and Upda, 1992)). Moreover, they are expected to give poor results if applied to Arabic cursive handwriting.

In conclusion, a segmentation technique for Arabic handwriting should accommodate, to a reasonable extent, variant and different styles of the writing.

2.2.3 Previous Efforts in Arabic Character Recognition

Search in the area of Arabic character recognition is relatively recent; it did not start until the early 1980's. This section reviews some previous efforts in this field, for both machine-printed characters and handwritten characters.

Machine-printed Arabic character Recognition

Some researchers worked in the recognition of isolated Arabic characters (Nouh and Sharaf Eldin, 1988,Sharkawy et al, 1988,). (Nouh et al, 1980) suggested a standard Arabic character set to facilitate computer processing of Arabic characters. The disadvantage of this approach is the assumption that the incoming characters are generated according to specified standard rules. Abd-Elazim, 1985 addressed typewritten text recognition, and implemented a complete system. At first, an input Arabic word is segmented into its primitives using the approach based on word vertical histogram, as previously described. Then the primitives are assigned to the possible pattern class using statistical Bayes' classifier, parallel correlation, and probabilistic decision tree approach. A recognition-mapping algorithm, then, maps the recognized sequence of primitives to the appropriate sequence of characters using structural rules. In (El-Gowely, 1989), a segmentation algorithm, also, based on the word vertical pixel density histogram is used to obtain character primitives. The main features that were used in classification were character width, weights above and below the base line and height. Character bit map and horizontal histogram were used.
A neural approach for Arabic character recognition is proposed in (Sharkawy et al, 1988). During segmentation of Arabic text into characters, characters are also divided into primary and secondary parts. The feature

extraction stage uses a set of moment invariant descriptors, which are invariant under shift, scaling, and rotation. Classification is done using a multilayer neural network with back-propagation learning. The moments are very sensitive to noise, any spurious dot, stroke, ligature, etc, are considered as part of the shape by the moment generation algorithm. The system is expected to give poor results if tested against data of different fonts and sizes.

A research based on a scientific framework rather than heuristic is proposed in (Abdulmaged , 1994). In this approach, a neural network is used, differently, to suggest a set of segmentation points. The input to the network consists of the direction codes generated by contour tracing. A binding phase treats the problem as an optimization problem to select from the suggested points a set of segmentation points that minimize a cost function. The optimization problem is solved using two approaches, Hopfield network and the Viterbi algorithm (Abdulmaged, 1994). Relying on a multilayer feedforward neural network trained by the backpropagation algorithm to obtain segmentation points makes the system inflexible to learn more fonts or character shapes. To learn more examples, training is restarted all over again.

Allam (1995) presented a system for Arabic text recognition based on a Hidden Markov Model. The approach relies on contour analysis to isolate components and detect base line. The features consist of labeling the bit distribution with respect to the base line of every column. A model describing the probability of occurrence of the labels at each column position is generated for each character. A character is recognized as the one corresponding to the model with highest accumulated probability.

Handwritten Arabic Character Recognition

Since the problem of handwritten Arabic character recognition is more challenge than printed characters, less work has been done in this field compared to printed character field. Some of the early work was done by Amin et al, at the university of Nacy in France (Amin et al,1980). They used an on-line system called the interactive recognition of Arabic characters(IRAC). This system was designed to recognize isolated Arabic characters. But since Arabic writing has a cursive nature and cannot be written isolated, this study was not practical. They introduced the IRAC II system (Amin and Masini, 1982). The system is aimed to recognize certain words stored in a dictionary. In this system, the written words are segmented into their constituting characters based on the observation that the characters of a word are connected together by horizontal strokes drawn from right to left. The classification of isolated characters is done depending on position of the character within a word, number of strokes, shape of secondary stroke, and position and size of dots. Although the system is suitable only for limited vocabulary. A low recognition rate was reached. The system did not take into consideration levels of characters and the base line which are necessary to differentiate between some characters. Amin and Masini developed the IRAC III system, where the word was segmented into strokes instead of characters (Amin and Masini,1982). They defined a stroke as position of a word between two successive pen lifts. The reported recognition rate of the system was higher than that of IRAC II system.

In (Abo-Samra, 1996), an on-line system for Arabic cursive script recognition is presented. Points of maxima and minima in both x and y directions are detected as feature points. The regions between the feature points are represented by four direction codes. Syntactic rules are then applied to find a set of codes that can define a character. Some contextual information and statistical rules are utilized to differentiate between similar characters having the same structure.

The off-line recognition is more difficult than on-line recognition. Less number of researches dealt with off-line Handwritten Arabic character recognition. H. Almullim and Yamguchi (Almuallim and Yamguchi, 1987) implemented a structural method for the recognition of off-line Arabic cursive handwriting. Words are first thinned, then segmented into strokes. A special definition of feature points is given to help in the extraction of strokes and a special algorithm is introduced for stroke extraction. Those strokes are then classified using their geometrical and topological properties. Finally, the relative position of the classified strokes is examined, and the strokes are combined in several steps using certain rules into a string of characters that represent the recognized word.

Al-Yousefi and Udpa (Al-Yousefi and Udpa, 1992) followed a statistical approach for the recognition of Arabic characters. At first, characters are segmented using vertical pixel density histograms. Then a lower level of segmentation is performed to segment the character into primary and secondary parts (dots and hamza). The vertical projections of the primary parts are then calculated and normalized with each character using simple measures of shapes calculated from the normalized moments. Classification is accomplished using quadratic discriminate functions. Once the primary character is recognized, it is associated with the corresponding secondary to obtain eventually the final character identity. However, the approach makes use of only simple shape measures derived from the normalized moments for the purpose of character classification. These measures as features are not sufficient to accommodate the expected different variations and styles of handwritten characters.

General Comments on Arabic Character Recognition Approaches

The following comments can be observed about the above approaches:

(1) Major source of errors in all Arabic character recognition systems is due to the segmentation process. This justifies the efforts made by researchers to arrive at more elaborate segmentation techniques suitable for Arabic writing.

(2) Many report using the structural approach for Arabic character recognition has been published (Saleh, 1994,Abuhaiba et al, 1994). Those are based on the fact that the essential information about a shape (character) is stored in its skeleton. Many algorithms for skeletonization has been proposed (Davies and Plummer, 1981,Stefanelli and Rosenfeld, 1971). An advantage of skeletonization is that it reduces the memory space required for storing the essential structural information present in the characters. It also simplifies the data structure required to process the character and reduces the required processing time. However, the skeletonization (thinning) process introduces some problems, which may be the main problem in off-line recognition of Arabic characters. Some Arabic characters contain small holes in the their body, e .g و, ف , and ق.

These holes, which may be solid either due to some writing styles or due to noise in the scanning stage, are recognized as noisy ends after skeleton. In addition, it would make discrimination between some characters more difficult, e.g. و and ر.

(3) Neural network approaches have been adopted for recognition of Arabic characters (Abelhamid, 1995, Majid , And Bayoumi , 1994). Neural networks have been proposed as potential solutions to various recognition tasks, especially feedforward multilayer networks trained by the backpropagation algorithm(Hart,1986). However, they suffer from

drawback of the incapability of incremental learning. Training the network with more examples, may restart all over again.

Therefore , neural networks are more convenient if the frequent need to learn more samples other than the already trained ones is not expected. For example, they may be suitable for machine-printed character recognition. For the task of handwritten character recognition, the ability of a system to easily learn more character shapes is a necessity.

2.2.4 Proposed System Outlines

This research deals with three stages of character recognition systems:

1-Skeleton representation.

2-Segmentation.

3-Classification.

Study objective

Our study tends to complete study of an algorithm that was used before to create a skeleton for Arabic characters. This algorithm is called the fuzzy ISODATA. This methodology succeeds to prove its ability to emphasize the combined relations for Arabic characters. Due to the ability of this method to represent Arabic characters, and after the selection of long execution time and initial selection of cluster, the method is considered suitable for Arabic character representation in either learning or operation phases.

Also this research tends to study new methodology for Handwritten Arabic character segmentations which relies on using neural networks for segmentation for the following reasons:

1- The absence of clear rules for segmentation.

2- The Varity of sizes for Arabic characters.

3- The Arabic characters Overlaps.

4- Rapid changes in handwritten characters.

Up to our knowledge ,neural networks were never used before to segment Arabic characters. They were used to segment Latin characters and numerals are not matured yet. The neural networks have many properties that can help in segmentation.

1- It can handle situations where there are no rules.

2- It can handle wide range of changes related to patterns that belongs to the same class.

This makes neural networks suitable to be used especially when it is needed to deal with handwritten Arabic character segmentation, regardless of the type and the size of writing.

This research is to complete previous efforts done to improve Arabic character recognition, taking into consideration that Arabic characters did not get enough attention like other languages. Thus neural networks were built and used to recognize the isolated character after learning the neural to recognize the isolated character.

System Development Phases

We built a Handwritten Arabic Character Recognition System (HACRS) based on two groups of algorithm: experimental and neural networks algorithm.

1-Experimental Algorithms : Those rely on developing experimental values, through a series of tests and experiments till we get the required values. The algorithms are smoothing algorithms, punctuation algorithms, features extraction algorithms, and loop detecting algorithms. A skeleton algorithm relies on a number of thresholds established by the trial and error methodology.

2-Neural networks: learning algorithm that can get desired results by learning. All neural networks used are multilayer Feedforward Backpropagation.

The HACRS consists of five stages: in each stage, The HACRS performs set of tasks. Figure 2.11 shows a block diagram of the HACRS process.

1- Preprocessing

The preprocessing stage contains:

a) Entry method, where all images at Arabic character text and isolated Arabic characters were entered using a scanner. The obtained images of characters are stored in binary window bitmap form.

b) Smoothing, where we get rid of all noise and smooth boundary at text by using one of two algorithms for smoothing. One of them was tested and used by (Khellah and Mahmoud,1994). The other was tested by (Atic ,1996).

c) Separation of connected components and punctuation by using the following boundary algorithms. After the separation of sub-words is done, extracting special characteristics, due to the shape and size of each secondary component, leads to recognize secondary components. The connected components(sub-words) are stored in separate locations.

2- Skeleton Detecting.

This stages consists of two major functions:

a) Applying parallel thinning algorithm to the images resulted from preprocessing stage.

b) Applying the Fuzzy ISODATA to find skeleton for every word of sub-word.

At the end of these stages, each found skeleton is represented by adjacency matrix, and skeleton centers are stored in a special folder.

3-Finding Segmentation Set and Feature Extractions:

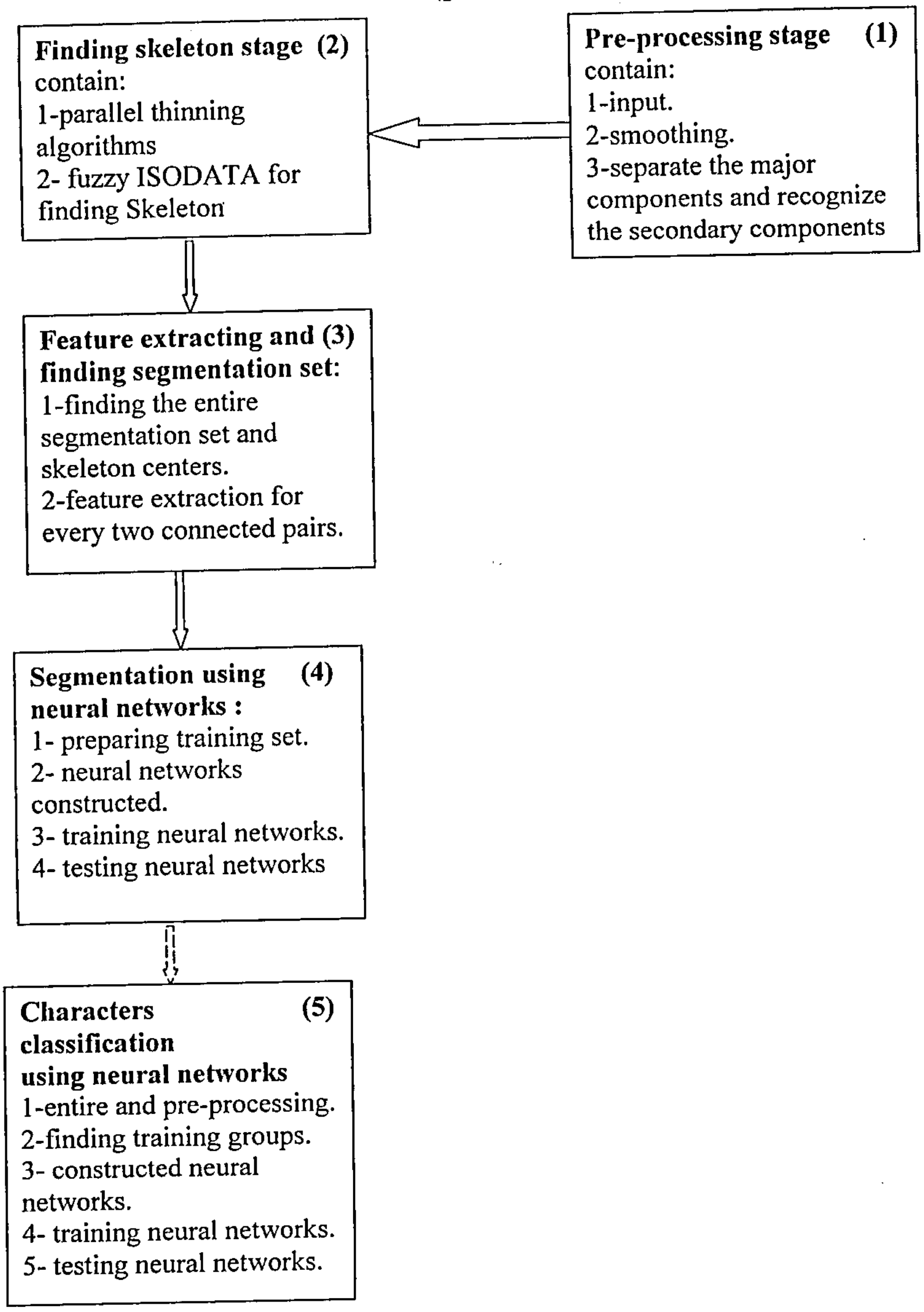
An overall segmentation group consists of pairs of connected vertices (centers). A skeleton is determined after excluding pairs that form loops or end of a sub-word. Then, special features for every two connected vertices are created as well as other generic features like loops, base line, and skeleton centers with special punctuation.

4-Segmentation Using Neural Networks.

Each two connected pair of vertices skeleton features are presented as input vector to the neural network: The neural networks determine the possibility for segmentation. The neural network system consists of eight neural, according to the types of connected pairs. The neural networks were done through epochs of learning and testing. Learning process is stopped when the value of error is lower than previously set value or when the number of learning epochs reaches the maximum allowed number.

5-Classification Using Neural Networks.

Here, the sample of data (isolated Arabic character) which was entered by scanners and stored as binary images was divided into training data and test data set. Learning via neural network was done on training sample of isolated Arabic characters (28 characters). Also we trained neural networks on another training sample which contain Arabic (102 characters) alphabet in all its forms (end form, middle form, beginning form and isolated form). The learning is stopped when the value of error is lower than a threshold, which previously determined or when the number of epoch reaches maximum predetermined constant number.



Pre-processing stage (1)
contain:
1-input.
2-smoothing.
3-separate the major components and recognize the secondary components

Finding skeleton stage (2)
contain:
1-parallel thinning algorithms
2- fuzzy ISODATA for finding Skeleton

Feature extracting and (3) finding segmentation set:
1-finding the entire segmentation set and skeleton centers.
2-feature extraction for every two connected pairs.

Segmentation using (4) neural networks :
1- preparing training set.
2- neural networks constructed.
3- training neural networks.
4- testing neural networks

Characters (5) classification using neural networks
1-entire and pre-processing.
2-finding training groups.
3- constructed neural networks.
4- training neural networks.
5- testing neural networks.

Figure 2.11:Block diagram of HACRS structure.

2.3 Artificial Neural networks

2.3.1 What is a neural network (NN)?

First of all, when we are talking about a neural network, we should more properly say "artificial neural network" (ANN). Biological neural networks are much more complicated than the mathematical models we use for ANNs. But it is customary to be lazy and drop the "A" or the "artificial."

There is no universally unique definition of an NN. Most people in the field would agree that an NN is a network of many simple processors ("units"), each possibly having a small amount of local memory. Communication channels ("connections") which usually carry numeric (as opposed to symbolic) data, encoded by any of various means connect the units. The units operate only on their local data and on the inputs they receive via the connections. The restriction to local operations is often relaxed during training.

Some NNs are models of biological neural networks and some are not. Historically, much of the inspiration for the field of NNs came from the desire to produce artificial systems capable of sophisticated, computations and intelligent similar to those that the human brain routinely performs, and thereby possibly to enhance our understanding of the human brain.

Most NNs have some sort of "training" rule whereby the weights of connections are adjusted on the basis of data. In other words, NNs "learn" from examples (as children learn to recognize dogs from examples of dogs) and exhibit some capability for generalization beyond the training data.

NNs normally have great potential for parallelism, since the computations of the components are largely independent of each other. Some people regard massive parallelism and high connectivity to be defining characteristics of NNs. Such requirements rule out various simple models,

such as simple linear regression (a minimal Feedforward net with only two units plus bias), which are usefully regarded as special cases of NNs.
The following are different definitions of NNs according to different researchers:

.1A neural network is a system composed of many simple processing elements operating in parallel whose function is determined by network structure, connection strengths, and the processing performed at computing elements or nodes (Darpa, 1988).

2.A neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

(a) The network through a learning process acquires knowledge.

(b) Inter neuron connection strengths known as synaptic weights are used to store the knowledge (Haykin,1994).

3.A neural network is a circuit composed of a very large number of simple processing elements that are neurally based. Each element operates only on local information. Furthermore, each element operates asynchronously; thus there is no overall system clock (Nigrin, A.1993).

4.Artificial neural systems, or neural networks, are physical cellular systems, which can acquire, store, and utilize experiential knowledge (Zurada,1992).

2.3.2 Basic Concepts

A simple neuron consist of the following (Patterson,1996), as shown in Figure 2.12:

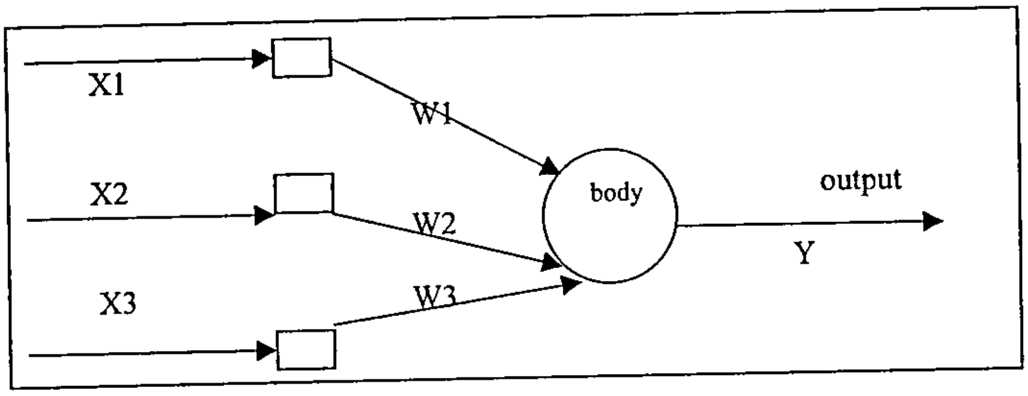


Figure 2.12: A Simple Artificial Neural Network.

1.Input links: which carry the input values to the neuron body.

2.Weights on the input link: which act to either increase (excitatory input) or decrease (inhibitory input).

3. Body: which receives the input values from the different input links and performs some computations and produces an output.

4.Output links: which carry the output value computed by the neuron body. The output value can be passed to another neuron or can be one of the final outputs of the networks.

The neuron behaves as an activation or mapping function f(.) producing an output y= f(net), where net is the cumulative input stimuli to the neuron. The f is typically a nonlinear function of net. Equation (2.1) shows how net is computed.

$$Net = x_1w_1 + x_2w_2 + x_3w_3 + \ldots + x_nw_n = \sum_{i=1}^{n} xiwi \qquad (2.1)$$

A threshold θ is sometimes included in the definition of net, and sometimes is replaced by placing a fixed bias input of $+1 * w_0$ on one of the input links and setting the value of $w_0 = -\theta$.

Activation function: which takes the net as input, and responds if the net exceeds the threshold value. It expressed as in equation (2.2)

$$F(net) = \begin{cases} 1, net & > \theta \\ 0, net & \leq \theta \end{cases} \tag{2.2}$$

Figure 2.12 shows simple ANNs. Figure 2.14 and Table 2.4 simulate the AND logic function. Note that x_0 and w_0 are used to represent the threshold value θ, such that x_0 is always 1 and $w_0 = -\theta$.

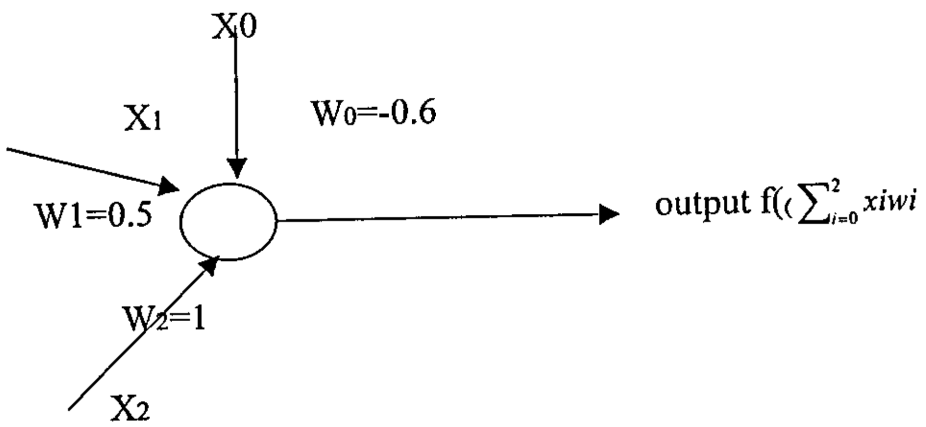


Figure 2.13 a simple neuron represents AND functions.

The value of output is $f(net) = \begin{cases} 1 \ , \ net & > \ 0 \\ 0 \ , \ net & \leq \ 0 \end{cases}$ (2.3)

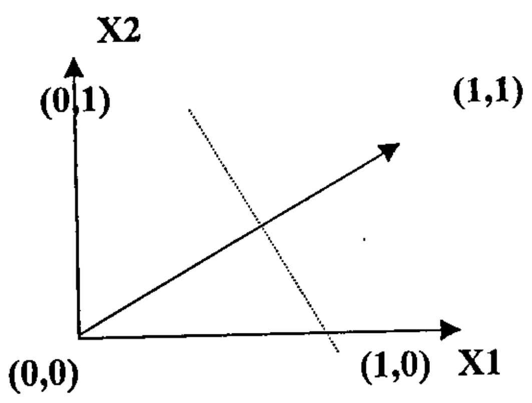


Figure 2.14:The AND function is linearly separable.

Table 2.4 AND function.

X1	X2	X1 AND X2
0	0	0
0	1	0
1	0	0
1	1	1

Figure 2.13 shows that the AND function is linearly separable.. This XOR function, which is illustrated in Figure 2.15 and Table 2.5, is not linearly separable.

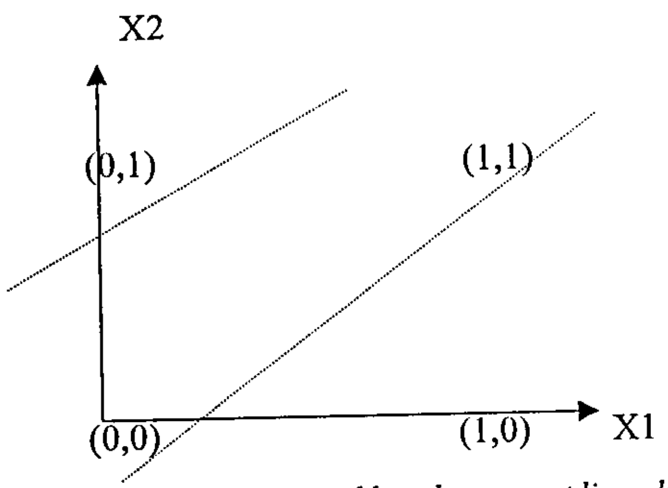


Figure 2.15 XOR problem that are not linearly separable.

Table 2.5 XOR function.

X1	X2	X1 XOR X2
0	0	0
0	1	1
1	0	1
1	1	0

In Figure 2.15, there is clear no line (plane) that can separate the two types of points. The Boolean exclusive or (XOR) function, with the truth table shown in Table 2.5, the desired computation is "yes" when one or the other of input is on and "no" when they are both off or both on. Here, we cannot represent this function with a simple perception, we need more layer.

2.3.3 Multilayer Feedforward Neural Networks and Backpropagation

In this section, a popular ANN architecture, the Multilayer Feedforward networks (MLFF) with Backpropagation (BP) learning is described. Figure 2.16 shows a general MLFF network. It is a Feedforward, fully connected hierarchical network consisting of input layer, one or more hidden layers, and an output layer. The internal layers are called "hidden" because they only receive internal inputs (inputs from other processing units) and produce internal outputs (outputs to other processing units). Consequently, they are "hidden" from the outside world.

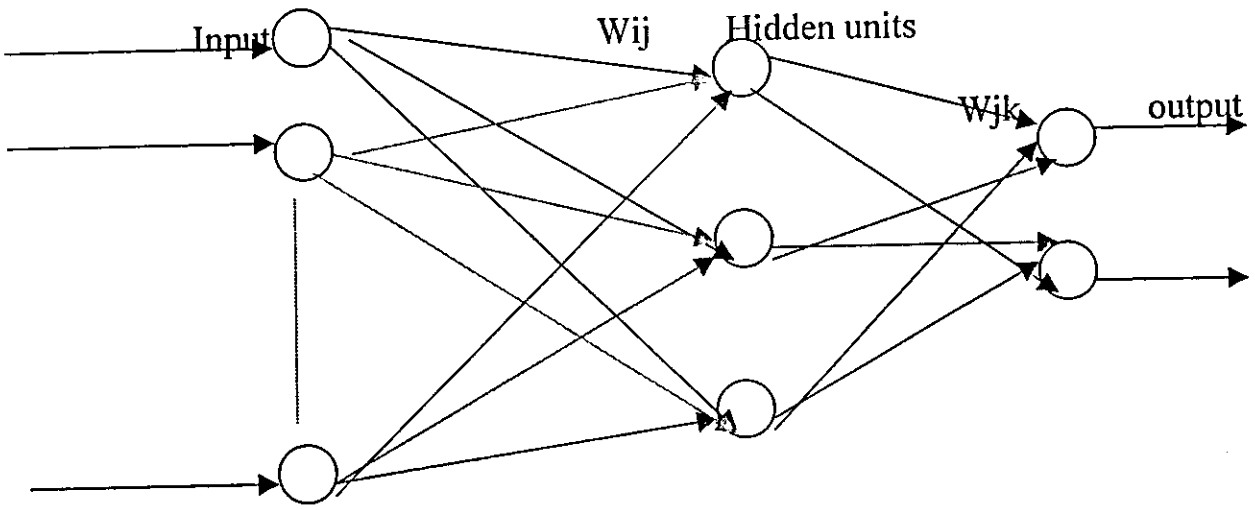


Figure 2.16 A general Multi-layer Feedforward Network.

Because of its importance, BP learning algorithm is summarized in Figure 2.17 for MLFF with a number of layers Q, and q=1,2,3...Q. Figure 2.17 outlines the basic steps needed to train an MLFF network (Patterson, 1996). Before specific network can be used effectively in a given application, many questions must be answered and a set of parameters used in BP learning must be specified. These include specifying the BP activation function, error measure function, the value of learning rate , η required number of layers, etc.

Backpropagation Algorithm

1- Initialize all weights W to small random values within the range [- λ , λ].

2-Randomly select a pair of training patterns $[x^p, t^p]$ (input, target) and compute in a Feedforward direction the output values for each unit j of each layer q, thus

$$O_j^q = f\left(\sum_i o_i^{q-1} W_{ji}^q\right)$$

Note that the inputs to layer one are indexed with the superscript o, and hence.

$-O_j^o$ Xj.

3-Use the values O_j^q compute by the final layer units and the corresponding target values t_j^p to compute the delta quantities

$\partial_j^q = (O_j^q - t_j^p) f'(H_j^q)$ For all j using pattern p.

4- Compute the deltas for each of the preceding layers by backpropagating the error using

$$\partial^{q-1} = f'(H_j^{q-1}) \sum_i \partial_i^q W_{ji}^q$$

For all j in each of the layers q =Q, Q-1…

5- Update all weights Wji using

$$W_{ji}^{q}\,{}_{3} + W_{ji}^{new} = W_{ji}^{old}$$

For each layer q, where $-W_{ji}^{q}\,{}_{3}\ .\ \partial_i^{q} O_j^{q-1}$

6- Return to step2 and repeat for each pattern p until the total error has reached an acceptable level.

Figure 2.17: A summary for Backpropagation Algorithm (Patterson,1996).

2.3.4 Error Surface and Convergence Properties

BP training is based on the gradient descent computations derived before. This process iteratively searches for a set of weights W* that minimizes the error function E over all training pairs. MLFF networks with nonlinear activation functions have an MSE error surface, which is in general complex and believed to have many local and global minima illustrated for the two-dimensional weight space case in Figure 2.18.

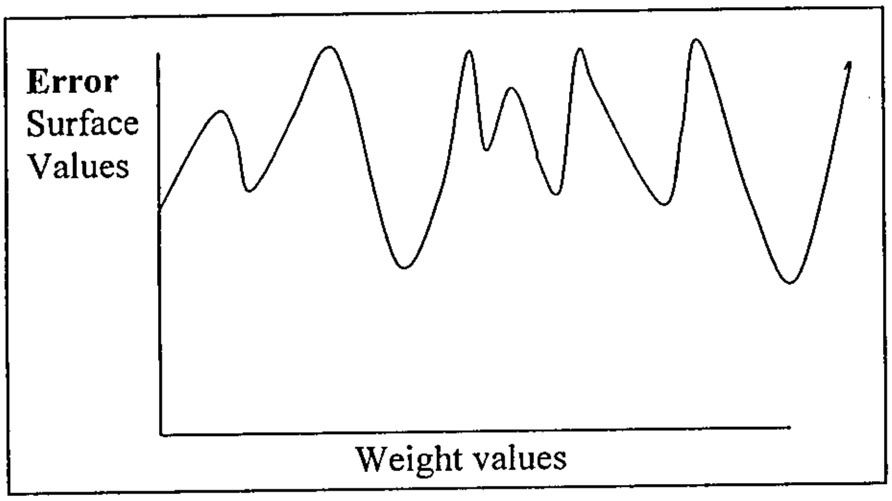


Figure 2.18:Typical Error Surface for MLFF Networks with Nonlinear Activation Functions (Patterson, 1996)

The multiplicity of minima arises in part from the symmetry of the weights and nodes in the network. During training, the gradient decent computations determine how weights should be modified at each new location to move down the error-weight surface. The path followed by BP algorithm is not always the ideal path. It depends on the surface shape and the learning coefficients η. Sometimes the followed path may slide down the error surface into a set of weights that is a local minimum. The network may never reach the optional set of weights (global minimum). This requires reruning the algorithm with new random initial weights and other network parameters hoping to reach the global minima in the new run. As a consequence, BP is never assured to find a global minima.

2.3.5 Resilient Backpropagation

To solve the problems of BP, many algorithms were devised, and in fact are variants of BP, like Delta-Bar-Delta algorithm, Quickprop and others. Later a very efficient and fast algorithm was invented, it is called "resilient Backpropagation" or RPROP, devised by Martin Riedmiller (Patterson, 1996).

Rprop is a local adaptive learning scheme (local adaptation strategies are based on weight-specific information, only, as the temporal behavior of the partial derivative of this weight. The local approach is more closely related to the neural network concept of distributed processing in which computations can be made in parallel) performing supervised batch learning (or learning by epoch). Though Rprop is also based on gradient decent, its basic principle is to eliminate the harmful influence of the size of the partial derivative on the weight step, only the sign of the derivative is considered to indicate the direction of the weight update.

$$\Delta ij(t) = \begin{cases} -\Delta_{ji}^{(t)}, \ if \ \frac{\partial E^{(t)}}{\partial wij} > 0 \\ +\Delta_{ji}^{(t)}, \ if \ \frac{\partial E^{(t)}}{\partial wij} < 0 \\ 0 \ , \ otherwise \end{cases} \qquad (2.4)$$

The algorithm is still that of gradient decent, same as BP, but the weights update procedure is different. It depends on the derivative sign rather than its value. The size of the weight change is exclusively determined by a weight-specific, so-called "update-value" $\Delta ij^{(t)}$, as show in equation (2.4).

The $\frac{\partial E^{(t)}}{\partial wij}$ denotes the summed gradient information over all patterns of the pattern set (Batch learning, or learning by epoch). This means that the weights update and adaptation are performed after the gradient information of the whole pattern set is computed. By replacing the Δij(t) by a constant updates value $\Delta ij^{(t)}$, equation (2.5) yields the so-called 'Manhattan' update rule.

The second step is to determine the new update values $\Delta ij^{(t)}$ as in equation (4.5). This is based on the sign dependent adaptation process.

$$\Delta ij^{(t)} = \left\{ \begin{array}{l} \eta^{+} * \Delta_{ij}^{(t-1)}, if \ \frac{\partial E^{(t-1)}}{\partial wij} * \frac{\partial E^{(t)}}{\partial wij} > 0 \\ \eta^{-} * \Delta_{ij}^{(t-1)}, if \ \frac{\partial E^{(t-1)}}{\partial wij} * \frac{\partial E^{(t)}}{\partial wij} < 0 \\ \Delta_{ij}^{(t-1)}, otherwise \end{array} \right\}$$

(2.5)

The adaptation rule works as follows: every time the partial derivative of the corresponding weights Wij changes sign, which indicates that the last update was too big and the algorithm has jumped over a local minimum. The update-value $\Delta ij^{(t)}$ is decreased by the factor η^{-}. If the derivative retain its sign, the update-value is slightly increased in order to accelerate convergence in shallow regions. Additionally, in case of a change in sign, there should be no adaptation in the succeeding learning step. In practice, this can be achieved by setting $\frac{\partial E^{(t-1)}}{\partial wij}$ =0 in the adaptation rule.

In order to reduce the number of freely adjustable parameters, often leading to a tedious search in parameter space, the increased and decreased factors are set to fixed values. The choice of the decrease factor η^{-} was lead by the following considerations: if a jump over a minimum occurred ,the previous update value was too large for it cannot be derived from gradient information. The correct value had to be estimated. On average it would be a good guess to have update-value (maximum likelihood estimator), so it was chosen as η^{-} =0.5The increase factor η^{+} on the one hand, has to be large enough to be large enough to allow fast growth of the update-value in shallow regions of the error function. On the other hand, the learning process can be considerably disturbed if a too large increase factor leads to persistent changes of the direction of the weight step. In several experiments, it was found that the choice of η^{+}=1.2 gives very good results, independent of the examined problem. Slight variations of this value did neither improve nor deteriorate convergence. So, in order to get parameter choice more simple, it was decided to constantly fix the increase parameter to η^{+}=1.2

All this allows Rprop to try to adapt its learning process to the topology of the error function.

Rprop Algorithm

The Following pseudo code describes how Rprop updates the weights. The rest of the algorithm is the same as previously defined BP.

The code shown here is only for one weight layer, but it is applied to any number weight layer. The minimum (maximum) operator delivers the minimum (maximum) of tow numbers. The sign operator returns +1 if argument is positive, -1 if argument is negative and 0 otherwise . Figure 2.19 shows the Rprop algorithm.

We used the RPROP in the classification stage, while other algorithm is failed to learn the handwritten Arabic characters in our experiments.

```
$\forall i,j : \Delta ij^{(t)} = \Delta 0$
$\forall i,j : \frac{\partial E^{(t-1)}}{\partial wij} = 0$
{Values for $\Delta 0$ can be set by random function}
Repeat
Compare Gradient $\frac{\partial E^{(t)}}{\partial wij}$
For all weights and biases}
If$\left\{\frac{\partial E^{(t-1)}}{\partial wij} * \frac{\partial E^{(t)}}{\partial wij} > 0\right\}$then
$\Delta ij^{(t)}$=minimum ($\Delta ij^{(t-1)*}$ $^{+}$,$\Delta$max)
$\Delta wij^{(t)}$=-sign ($^{*}\frac{\partial E^{(t)}}{\partial wij}$ $\Delta ij^{(t)}$
$\Delta wij^{(t+1)}$= $wij^{(t)}$+ $\Delta wij^{(t)}$
$\left\{\frac{\partial E^{(t-1)}}{\partial wij} = \frac{\partial E^{(t)}}{\partial wij}\right\}$}
Else if
$\left\{\frac{\partial E^{(t-1)}}{\partial wij} * \frac{\partial E^{(t)}}{\partial wij} < 0\right\}$then {
$\Delta ij^{(t)}$=maximum ($\Delta ij^{(t-1)*}$ $^{-}$,$\Delta$min)
$\frac{\partial E^{(t-1)}}{\partial wij}$=0}
Else if
$\left\{\frac{\partial E^{(t-1)}}{\partial wij} * \frac{\partial E^{(t)}}{\partial wij} = 0\right\}$then
$\Delta wij^{(t)}$=-sign ($^{*}\frac{\partial E^{(t)}}{\partial wij}$ $\Delta ij^{(t)}$
$\Delta wij^{(t+1)}$= $wij^{(t)}$+ $\Delta wij^{(t)}$
{$\left\{\frac{\partial E^{(t-1)}}{\partial wij} = \frac{\partial E^{(t)}}{\partial wij}\right\}$
End for
Until (converge)
```

Figure 2.19: Weight update in Resilient Backpropagation at instance t (RPROP) (Riedmiler, 1994)

Chapter 3
Handwritten Arabic Characters Recognition Systems (HACRS)

3.1 Pre-Processing, Finding Skeleton and Features Extracting Phases

Initial processing for the picture means applying special techniques of "Image Restoration" or "image enhancement". Image Restoration can be considered as an "estimation process" where the given picture is processed to estimate the form it should be if the input process was ideal. (i.e. estimate the ideal image). On the other hand image enhancement requires using a wide set of techniques to clear the picture or converted it to a form suitable to process used by machine or human (Michael, 1986).

3.1.1 Digitizing and Pre-Processing

Pre-Processing Objectives for Arabic Text Pictures:

The Bi-image, gap filling, and boundaries should be dealt with first.

1- Binarization

Digital image is a set of pixels produced by a scanner in which there is 600 bpi. Every gray level is represented by number in (0 –255). A special threshold is used to convert the pixels to white pixels (gray level 0) or black pixels (gray level 1). The gray level for the pixels is called 'pixel value'.

2- Gap filling and noise elimination.

Many reasons cause gaps and noise to be in the text that has to be cleared, resulted from the input machine or the input media itself (for example; paper) or the writing pen.

3- Making the boundaries of any connected area 8-connected, this condition is essential for the boundary-following algorithm to succeed.

Statistical Smoothing Algorithm

This algorithm removes noise, which appears in the image as small black holes. It also fills the small gaps on the image's boundary, using the statistical decision criterion.

For any pixels in the Bi-image the value of this pixel p_0 modified according to its original value and the values of the neighboring pixels using the following rules (Mahmoud et al, 1991):

1.if $p_0=0$ then

$$P0' = \begin{cases} 0 \text{ if } \sum_{i=1}^{8} pi < T \\ 1 \text{ otherwise} \end{cases}$$

2- if $p_0=1$ then

$$P0' = \begin{cases} 1 \text{ if } pi + pi - 1 = 2 \text{ for at least one } i = 1...8 \\ 0 \text{ otherwise} \end{cases}$$

Where p_0 the current pixel value, $P0'$ is the current pixel value after modification, and T is a fixed threshold.

The 8 pixels are labeled in the order shown in Figure 3.1.

P_4	P_3	P_2
P_5	P_0	P_1
P_6	P_7	P_8

Figure 3.1: initial pixel p0 and the order of the eight neighboring pixels.

3.1.2 The Separation of The Components

Definitions and objectives.

Path: a path from the pixel at (i_0,j_0) to the pixel at (i_n,j_n) is a sequence of pixel indices (i_0,j_0), (i_1,j_1),..., (i_n,j_n) such that a pixel at (i_k,j_k) is a neighbor of the pixel at (i_{k+1},j_{k+1}) for all k with $0 \le k \le n-1$.

Foreground: The set of all 1 pixels in an image is called the foreground and is denoted by S.

Connectivity: A pixel p$\in$S is said to be connected to q $\in$S if there is a path from p to q consisting entirely of pixels of S.

Connected components: A set of pixels in which each pixel is connected to all other pixels is called a connected component.

Background: The set of all 0 pixels in an image is called the background (the complemented of S) and is denoted by S'.

Boundary: The boundary of S is the subset of pixels of S that have 4-neighbors in S'. The boundary is usually denoted by S''.

Interior: The interior is the subset of pixels of S that are not in its boundary. The interior of S is (S - S'').

Figure 3.2 represents the four neighbors and the eight neighbors for a pixel p0. The pixel connected to its 4 neighbors is called 4-connected and the one connected to its 8-neighbors is called 8-connected.

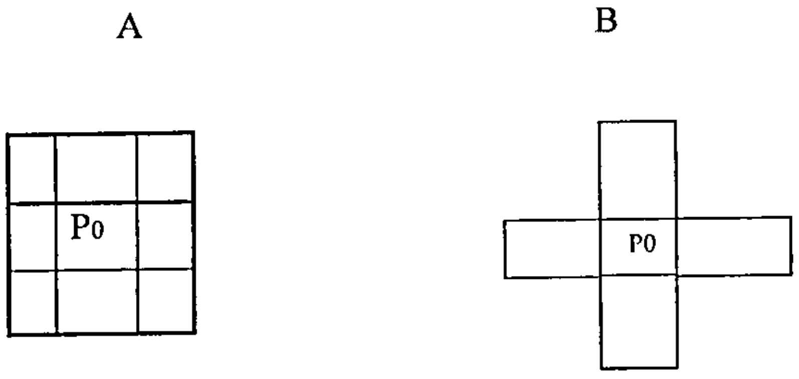


Figure 3.2: A) 8-connected (8-neighbored) and B) 4-connected (4-neighbored)

The aim of the separation of component stage is to separate the connected areas in the image. Connected areas are either characters or sub-words and in this case they are called primary component or may form a punctuation marks (dot '.', 2 dots '..',3 dots '.:', hamza 'ء') which are called secondary components.

Also, this stage aims to distinguish the secondary component based upon the features that distinguish it from primary components. At the end of this stage, every primary component is stored in its image file.

Boundary Following Algorithm

For any connected area S, in a Bi-image, the boundary-following algorithm is as follows (Jain et al,1995).

1- Find a starting pixel s $\in$ S for region using a systematic scans, say from right to left and from top to bottom of the image.

2- Let the current pixel in boundary tracking be denoted by c. Set c =s and let the 4-neighbor to the east of s be b $\in S'$.

3- Let the eight 8-neighbors of c starting with b in clockwise order denoted by n_1, n2...n8. Find n for the first i that is in S.

4- Set c $=n_i$ and b $=n_{i-1}$.

5- Repeat steps 3 and 4 until c =s.

6- Separate the area with the following boundary from the rest of the image.

7- Repeat step 1-6 till there is no connected area in the image with size more than 1 pixel.

The steps 1-5 form a boundary following algorithm. The boundary following algorithm selects a staring pixels s $\in$ S and tracks the boundary until it comes back to the starting pixel, assuming that the boundary is not on the edge of an image. This algorithm work for all regions whose size is greater than 1 pixel.

Recognition of Secondary

Figure 3.3 illustrates the Algorithm used for recognizing the secondary. The punctuation marks are not classified. It depends on three features and five experimental thresholds. The features are:

1-Size of the connected area: The size of connected area is the number of black pixels that the connected area has. In most cases, there is clear difference between the size of primary component and the size of the secondary one. As shown in Figure 3.3, this feature deals with two thresholds (T1, T2).

2-Length/Width Relation: The length or width of a connected area is defined as the column numbers (width) or rows (length) that is occupied by the smallest closing box that can contain the connected area. This feature is not affected by the font size (Jain et al, 1995). It deals with one threshold (T3).

3-The Ratio between black and white pixels in the closing box that contains the connected area. It deals with two thresholds (T4, T5).

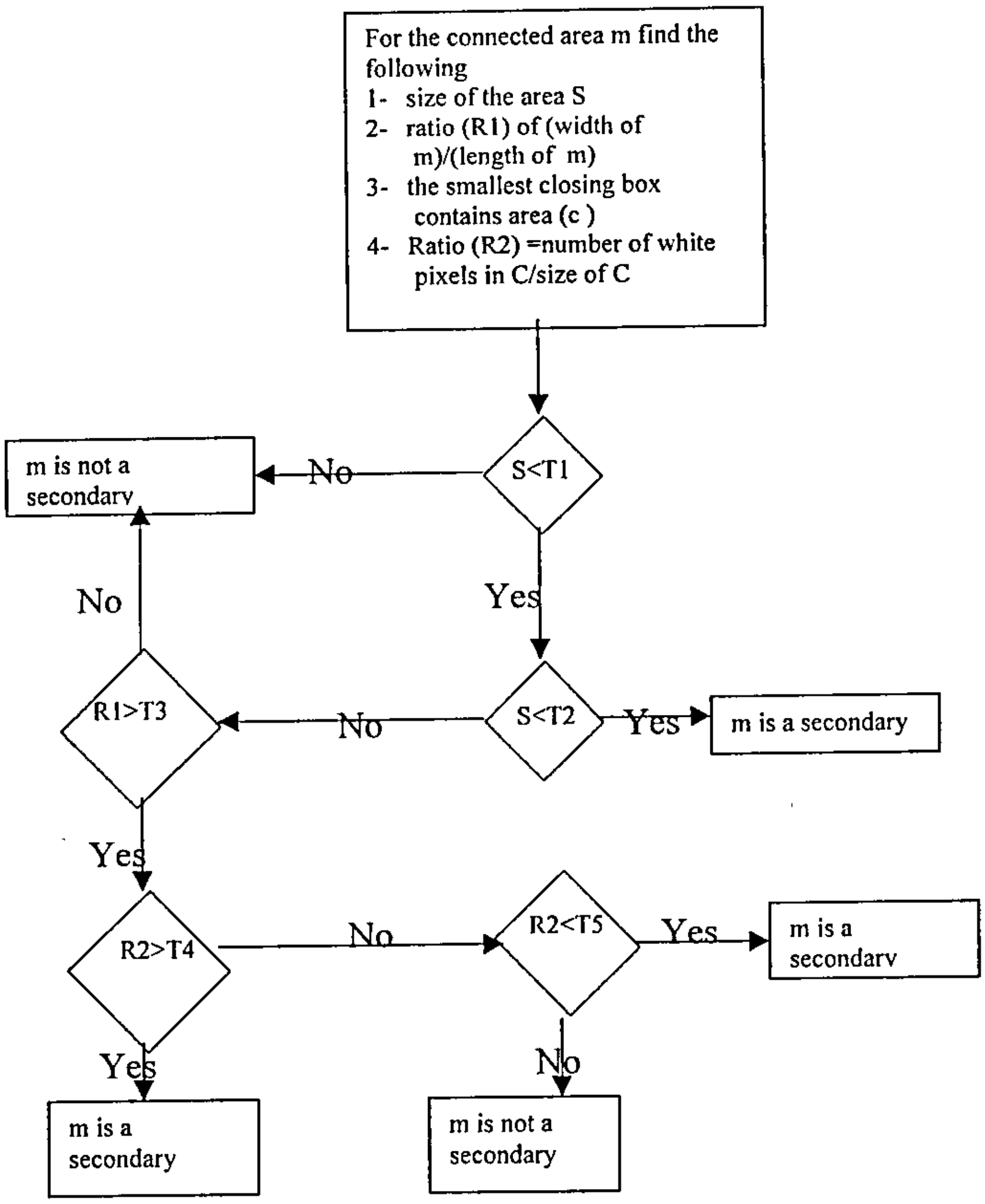

For the connected area m find the following
1- size of the area S
2- ratio (R1) of (width of m)/(length of m)
3- the smallest closing box contains area (c)
4- Ratio (R2) =number of white pixels in C/size of C
S<T1
No
m is not a secondary
Yes
S<T2
Yes
m is a secondary
No
R1>T3
No
Yes
R2>T4
No
R2<T5
Yes
m is a secondary
Yes
m is a secondary
No
m is not a secondary

Figure 3.3.Punctuation Recognition Algorithm, where (T1, T2, T3, T4) are experimental fixed thresholds and T1=270,T2=100,T3=0.5,T4=2.0,T5=0.8 in our experiments.

Figure 3.4 explains those features. The punctuated box c surrounding the connected area represents the smallest boundary that can contain m. R1 =w/l represents the width/length ratio in m. R2 represents the white and black number of pixels inside the box and its size. If the size m =236 pixels, then R1=3.46 and R2=0.63

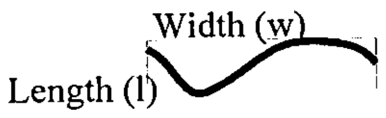


Figure 3.4: The image here shows a connected area m representing a punctuation mark (2 connected dots).

3.1.3 Skeletonization of Arabic Word or Sub-word

The algorithm used here is based on extracting a skeleton for the entered pattern. The pattern consists of cluster of connected pixels. Each cluster is represented by its center and the groups of centers are control points or skeleton vertices. A cluster is a group of connected pixels that can be represented by its center (Mahmoud et al, 1991). This algorithm has been reached by Bezdek and Dunn, and it is well known as fuzzy ISODATA. It is ahead of the traditional ISODATA algorithm in its ability to gather the different data in cluster. It has been used in solving many problems in pattern recognition. Mahmoud et al (1991) used the fuzzy ISODATA algorithm for skeleton of handwritten isolated Arabic characters.

In this study, the fuzzy ISODATA algorithm is used with two modifications, first one adding a thinning process before finding the skeleton and the second is a method for choosing the initial number of clusters.

Parallel Thinning Algorithm

One of the famous thinning algorithms, is the one that has been developed by Zhang and Suen (1984).

The thinning algorithm is consist of consecutive passes, each one is made of two steps. In the first step, pixels are flagged for deletion if the following conditions holds:

a- $2 \leq N(p_i) \leq 6$

b- $S(p_i)=1$

c- $P_1*P_3*P_7=0$

d- $P_1*P_5*P_7=0$,

Where $N(P_0)$ is the number of neighbor for P0 and belongs to S (black pixels) and $N(P_0)=P_1+P_2+....+P_8$.

S(P0) is 0-1 transition of neighbors according to the order ($P_3,P_2,P_1,P_8,P_7,P_6,P_5,P_4,P_3$). Figure 3.5 shows how to calculate conditions a and b.

In the second steps the conditions c and d are modified as

c- $P_3*P_1*P_5=0$

d-$P_3*P_7*P_5=0$

In each step, a flag for deletion is made for the pixels. The deletion is made after examine all the image pixels, therefore the iteration for applying this algorithm is made up from:

1- Apply step1 to flag the deleteable pixels.

2- Delete the flagged pixels.

3- Apply step 2 to flag the deletable pixels.

4- Delete the flagged pixels.

This algorithm ends when the number of flagged pixels becomes zero.

0	0	1
1	P0	0
1	0	1

Figure 3.5: how to calculate condition a and b.

In this case ,S(P0)=3, and N(P0)=4.

Fuzzy ISODATA Algorithm.

The following is a brief description of the fuzzy ISODATA algorithm of Bezdek and Dunn. Bezdek and Dunn have shown that their fuzzy algorithm is superior to ordinary ISODATA from the standpoint of detecting the presence or absence of well-separated clusters in data and identifying such clusters if they exist. A fuzzy c-partition of a set X is a c-tuple of functions u_i (.):X (0,1), $0 \le i \le c-1$,which satisfies the condition $\sum_{i=0}^{c-1} ui(x) = 1$ at each $x \in X$. The u_i is called a fuzzy subset or fuzzy cluster in X. The u_i (x) is the grade membership of x in fuzzy set u_i.

The following steps summarize the fuzzy ISODATA algorithm.

Step1:choose a fuzzy partition u_i (.) of c nonempty membership functions (c is a fixed integer between 2 and the number n, of elements in x).

ui(.) $\neq 0$, $0 \le i \le c-1$.

step 2: compute the c weighted means mi,

$$mi = \frac{\sum_{x \subset X} (ui(x))^2 x}{\sum_{x \subset X} (ui(x))^2}, 1 \le i \le c$$

These are centers of clusters, which are iteratively modified depending on the values of u_i (.).

step 3:construct a new partition $\hat{u}$(.), as follows:

if $0 \le i \le c-1$ *and* $mi = x$, *then*

$$\hat{u}(x) = \left\{ \begin{matrix} 1 & j = i \\ 0 & j \neq i \end{matrix} \right\}, 1 \le i \le c$$

otherwise, (the usual case), then

$$\hat{u}(.) = \frac{(1/\|x - mi\|^2)}{\sum_{j=1}^{c} (1/\|x - mi\|^2)}$$

step4 : compute some convenient measure, , of the defect between ui(.) and $\hat{u}(.)$. if δ is less than a specified threshold (β), then stop. The

$\delta = \| \hat{ui} - ui \| \leq \beta$,where ||.|| is Euclidean distance.

otherwise , put $u_i(.) = \hat{u}(.)$ and go to step2.

Step5: the ultimate goal of clustering process is to segment each character into c disjoint clusters. Each $x \in X$ has different membership values in c clusters. To minimize the risk. Each pixel x is assigned to the cluster in which it has the maximum membership value. That is

$u_h(x) > u_i(x)$, for $1 \leq i \leq c, i \neq h$

Finding Adjacency Matrix.

An clustering algorithm is based upon computing the cluster centers. These centers and the adjacency relations between them form a skeleton of the entered pattern. The adjacent relation is expressed through the adjacency matrix H, which is a matrix of (c*c) entries and can be obtained by scanning the image through bi-window 2*2, for any 2 connected pixels in the windows belonging to 2 different clusters (i,j).These 2 clusters are adjacent and the 2 entries (Hij=1,Hji=1) are added to the matrix.

Initial Membership Assignment

In the first step (initialization) of the ISODATA algorithm, a primary Fuzzy is initialized for the pattern where the pixel is distributed among the clusters with a primary belonging grades using 2 strategies:

1- c-Consecutive slices strategy.

In this strategy, the n elements of the X array are divide into c consecutive segments, G_i, $1 \leq i \leq c$. Each segment G_i , except the last segment initially contains an equal number of elements. The last segment and the number of the black pixels (n) may not be a multiple integer of the cluster number (c). The elements of each segment have full membership. So, if an element $x \in X$ is initially in Gi, then

$$u_j(x)=\begin{Bmatrix} 1, j = i \\ 0, j \neq i \end{Bmatrix}, 1 \leq i, j \leq c$$

2- The c-interlaced slices strategy.

Here, the cluster to which a black pixel x_k, $0 \leq k \leq N-1$ is determined by finding the remainder of dividing the element numbers k by the number of clusters (k mod c). The remainder is always less than c. In this strategy, the pixels are assigned to clusters in semi-random fashion.

The difference in the two strategies lies in the distributed strategy. The consecutive slices strategy distributes the pixels over the clusters in an organized way, while the interlaced slices strategy distributes the pixels over the clusters in an interlaced way.

Initial Selection for the Cluster Number.

Applying the previous fuzzy ISODATA algorithm requires specifying the number of cluster in the range $2 \leq c \leq N$. Choosing a few number of clusters leads to a skeleton that does not correctly represent the entered pattern. A large number of clusters produce an overloaded skeleton with extra clusters that does not give any richness to the information obtained.

Our solution depends on the assumption that the number of the clusters proportions to the size of the pattern. Consequently, the clusters initial number must change according to the change of the size of the entered pattern e.i

C=N/T, where T is fixed experimental threshold.

Skeleton Reduction Algorithm

The following algorithm is applied to eliminate extra insignificant skeleton vertices (as a refining and reduction step).

Let a skeleton vertex, v_i be connected to vertices v_j and v_k. The angle, between the two segments. G_{ij} and G_{ik} is found. If α satisfies the condition $180^{\circ}-\gamma \leq \alpha \leq 180^{\circ}+\gamma$, where γ is a threshold angle), then the v_i is deleted from the adjacency matrix H (since the elimination of this vertex does not affect the general skeleton of the character). The entries of the matrix H are modified such that $H_{ij}=H_{ji}=0$, for all $L \in \{j,k\}$ and $H_{jk}=H_{kj}=1$. This process is repeated until no more vertices can be eliminated. Our experimental results have shown $\gamma = 10^{\circ}$ gives acceptable skeletons.

3.1.4 Features Extraction.

The features that will be extracted at this stage are designated to do the segmentation using the neural network. The way they've been organized must reflect this idea.

The proposed segmentation process is done between the connected skeleton vertices. So global features like loops and punctuation will be considered as a feature of the skeleton vertex.

Skeleton Vertex Type.

The types of vertices skeleton are classified according to the number of vertices connected to it:

1. End vertex connected with only one vertex.
2. Connect vertex connected with two vertices.
3. Branch vertex connected with three vertices.
4. Intersection vertex connected with four vertices.

Figure 3.6 illustrates these types.

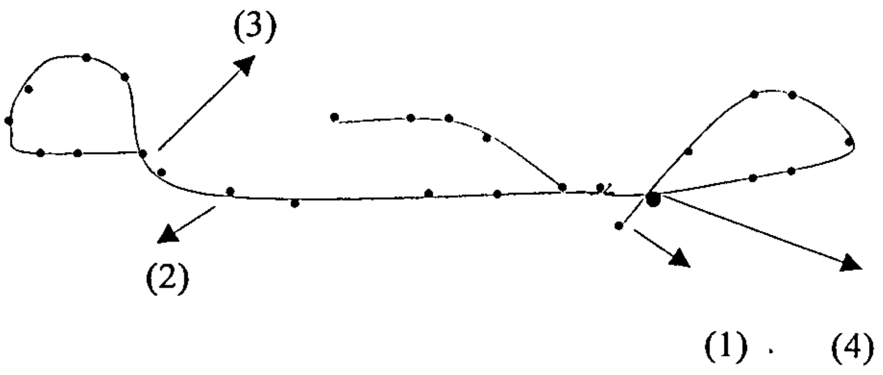


Figure 3.6:Types of vertices in a handwritten word "محمد":(1) end point, (2)connected point, (3) branch point, and (4)intersection point.

Loops

The loops widely appear in Arabic writing. Among the 102 shapes in the Arabic characters, 35 preserve this feature. 12 images might have loops according to the way of the writing. Any vertex (d) in the skeleton forms part of a loop if there is a path between the skeleton vertex and itself. This path comprises of a group of vertices L ,so that, the any vertex is passed only one time (one-way Gate). This means that it is possible to go back to any vertex in the loop through a path in one direction. The group L which contains d is called a loop (Jain et al,1995).

In HACRS, constrained Depth-first search algorithm is developed. This algorithm can find any loop, regardless of the number of vertices and their locations in a handwritten image.

Constrained Depth-First Search Algorithm.

The organized depth-first search algorithm is summarized as following process:

Input: skeleton of the Arabic word.

Output: loops in the skeleton.

For every branch vertex or intersection d:

1- Choose a start vertex s connected to d.

2- Make d= goal state and s= start.

3- Add s to the open state set O.

4- Choose vertex e, the last vertex added to the Open set (Deepest). Add its neighbors to Open set. Add the vertex e from the Open set to the Closed set C.

5- Check the open state set O :

a- if d $\in$ O stop successfully ,save the vertices of the closed state set C as a loop. (i.e. loop is found).

b- If O=   stop with failure.(i.e. no loop).

c- If O contains an end vertex stop with failure.

d- Otherwise go to step2.

The added rules to this algorithm are:

1. The start vertex is a branch vertex or intersection vertex, since any loop should have at least one vertex of these 2 types, so its possible to control or decrease the search attempts in adding this rule.
2. Stop search when the algorithm reaches or finds an end vertex. This is because the loop doesn't have an end vertex. Consequently the search in that direction will stop.

Figure 3.7 illustrates a loop in a skeleton section in Arabic word.

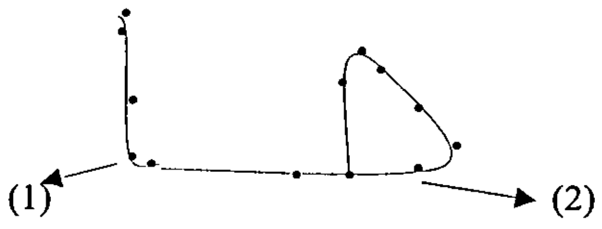


Figure 3.7 loop feature in Arabic writing.

(1) right angle feature with neighbors.

(2) vertices in the inner rectangle.

Punctuation Marks.

For a vertex v in the skeleton, v owns a punctuation mark m if v is the nearest vertex to m.

The concept "nearest" is expressed depending on two distances:

1-Vertical distance: A distance between the vertex v, where v =(x1, y1), and the punctuation mark m with center (x2, y2) =|y1-y2|, where |.| is the absolute value.

2-City-Block Distance: A distance between the vertex v and the punctuation mark m is |y1-y2|+|x1-x2|.

Angle of Inclination

Slopping point is expressed using the geometric concepts sine and cosine instead of the angle values. For any vertex a =(x1, y1) connected with vertex b=(x2, y2). The cases are expressed in equation (3.1) and (3.2). Where θ is the angle between a and b. The origin point coordinated according to vector translation law are x=x2-x1 and y=y2-y1.

$$\cos\theta = \frac{X}{\sqrt{X^2+Y^2}}, \qquad (3.1)$$

$$\sin\theta = \frac{Y}{\sqrt{X^2+Y^2}} \qquad (3.2)$$

Right Angle

The importance of this feature lies in computing the angle between three connected vertices to check if those vertices form a right angle. Figure 3.7 illustrates these features. Where vertex a=(x1,y1) connected to b=(x2,y2) and c=(x3,y3).

If you let ux =x2-x1, uy=y3-x1, vx=x3-x1 and vy=y3-x1, then you get the relation in equation (3.3).

Where β is the angle between vectors v and u (the angle between c and b whose center is a). The ||u|| is a length (norm) of vector u.

$$\cos \beta = \frac{u.v}{\| u \| . \| v \|} \qquad (3.3)$$

Distance

For any two pairs of the vertex skeleton (a,b) where a=(x1,y1), and b=(x2,y2), the

1. Horizontal distance between a,b is |x1-x2|, and
2. Vertical distance between a,b is |y2-y1|.

Baseline

For any skeleton's vertex, the vertical distance is computed between the vertex and the baseline. The feature is given a positive value if the vertex was above the base line and a negative value if it was under. The base line is considered as one feature used in the proposed segmentation system of HACRS among other features.

3.1.5 The Entire Segmentation Set.

The Entire segmentation set is defined as the set with all locations that segmentation could be done at. This is varied in its contents according the representation way and the segmentation method in the system.

The set consists of pairs of connected vertices. Due to having 4 types of vertices, the number of connected pairs type mathematically is 16, three of the 16 don't appear in the Arabic writing, and five types are not suitable for segmentation, reducing these types to 8 pairs. Table 3.1 shows the types., Figure 3.8 illustrates the 8 types.

For any pair (b,a) from the connected vertices, (b,a) is added to the entire segmentation set if :

1. b is left to a,
2. b is not an end vertex, and
3. (b,a) doesn't belong to one loop

Table 3.1: Types of vertices and number of features for each pair.

Type of pair	Vector name	Number of extracted features
C(Connected) with C	V1	24
C with B (Branch)	V2	31
C with I(Intersection)	V3	38
B with C	V4	31
B with B	V5	38
I with C	V6	38
E(End) with C	V7	15
E with B	V8	23

The first condition guarantees no duplication in pairs. The pair will be taken only once and in one direction from right to left. The second condition excludes the pairs ending with an end vertex. Figure 3.8 illustrates the locations of the pairs. The decision to exclude these pairs from the segmentation set is done after preparing the special training set, where the decision always was not to segment. This means there is no need to build a neural network.

A skeleton might have more than one loop like the letter (Ha,ـهـ), but the loop shouldn't belong to more than one character. This means that all the vertices inside the loop form part of one character. The third condition is added to achieve this feature. Table 3.2 shows the excluded pair types and the rarely appeared pairs in the Arabic word's skeleton.

Figure 3.9 shows examples on the excluded pair.

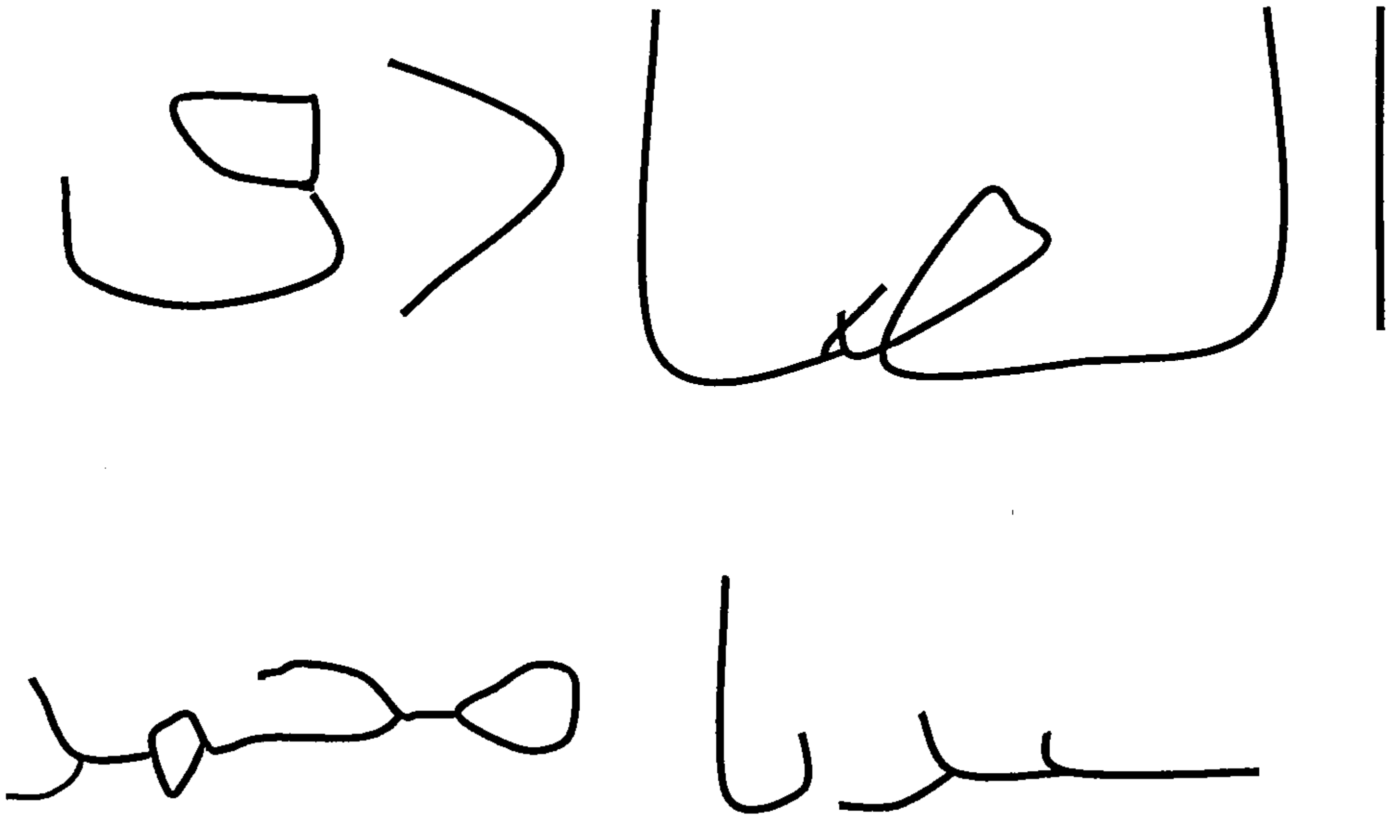


Figure 3.8: (a) Handwritten of three word without punctuation for "الصادق سيدنا محمد"

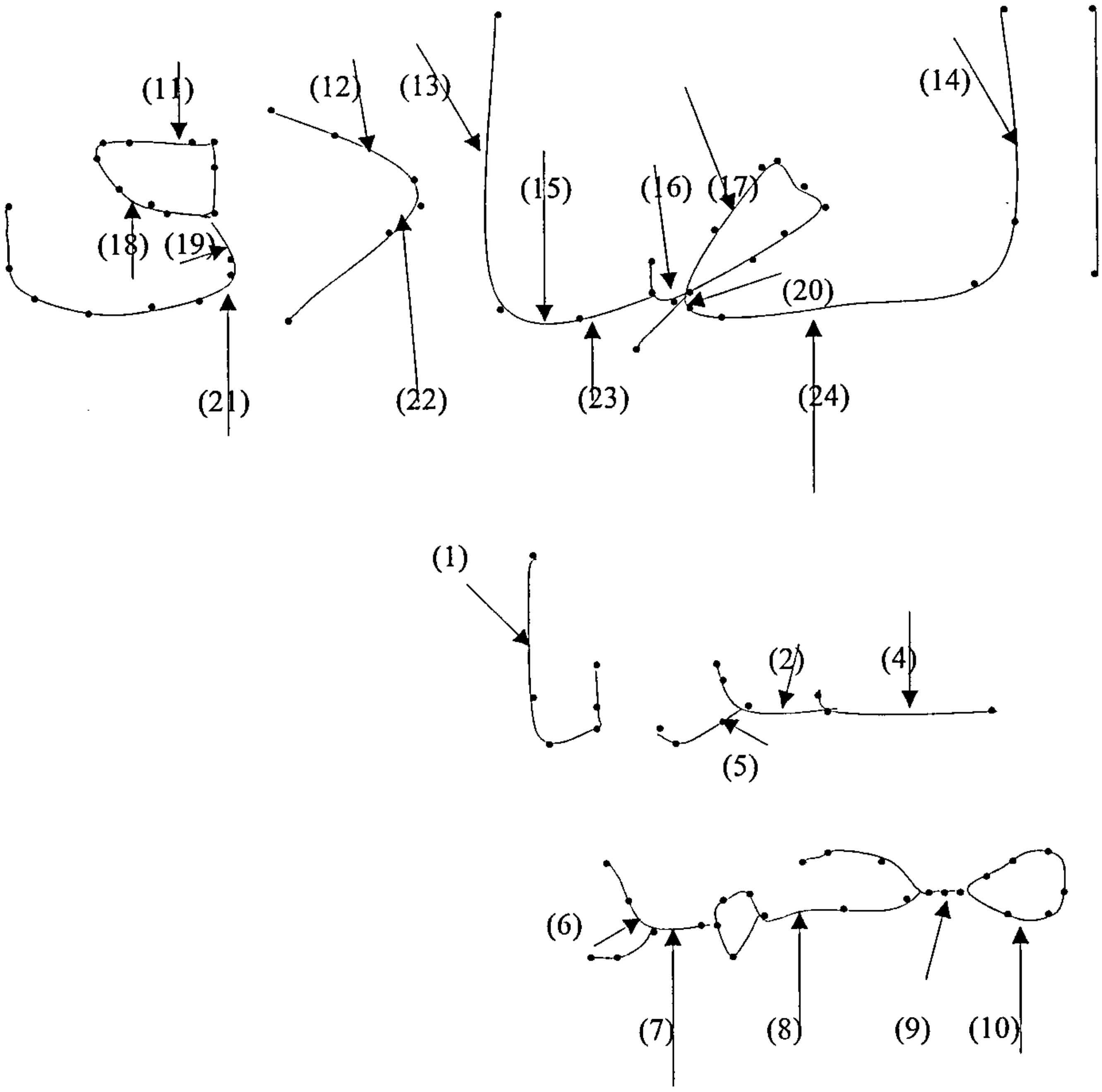


Figure 3.8 (continue): (b) Examples on types of vertices in the words "الصادق سيدنا محمد"

Type	Vertex
V1	11,12,22,15,17,9,10,13,18,19,21,24,3
V2	7,8
V3	20
V4	5,6,23
V5	2
V6	16
V7	14
V8	4

Figure 3.8: c) types of vertices in Figure 3.8 (b).

Figure 3.8: Types of skeleton vertices in the enter segmentation set: (a) The actual image of the words and subword, (b) points to the pairs that make the segmentation set, and (c) shows these types.

Notice the pair V1 the most iterated one in the segmentation set.

Table 3.2:Types of excluded skeleton vertices pairs and the uniquely shown pairs.

Excluded skeleton vertices	Uniquely pair
E with E	B with I
C with E	I with B
B with E	I with I
I with E	
E with I	

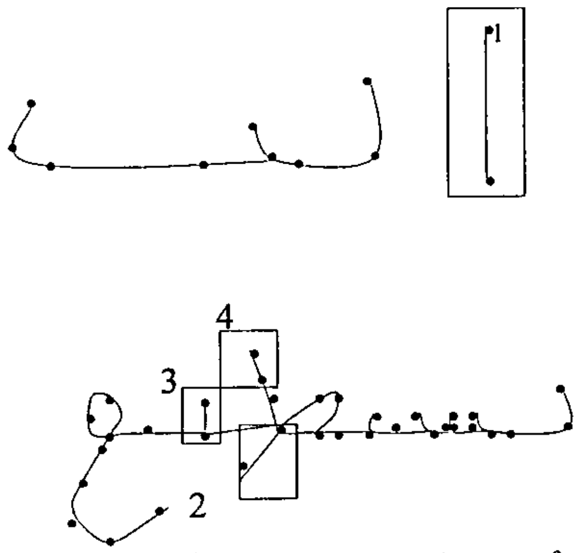


Figure 3.9: Locations in which excluded pairs in the entire segmentation set of "انت تستطيع" means "you can":

1: (end vertex, end vertex), 2: (intersection vertex , end vertex), 3: (branch vertex, end vertex),and 4: (connected vertex, end vertex).

Notice the locations where the pairs form an end vertex. (i.e. "ـــ" a character and "ط" a character).

Table 3.3 illustrates an example for input vector and the features it has (V1).

The preprocessing technique explained in this chapter necessary for the classified in HACRS.

Table 3.3: 24-features that make the vector V1

Suppose the pair (a, b) is a connected vertices and b left to a, c is the Vertex connected to a, and d is the vertex connected to b,then the extracted features are:

1. vertical distance between (a,b),
2. horizontal distance between (a,b),
3. is vertex a part of loop?
4. Does vertex a have punctuation mark?
5. horizontal distance between (a, baseline),
6. the cosine angle between(a,b)
7. the sine angle between (a,b)
8. type of vertex c,
9. is vertex c part of loop?
10. does vertex c have a punctuation mark?
11.horizantal distance between (c, baseline),
12. The cosine angle between (a, c),
13. The sine angle between (a, c),
14. The cosine angle between (b, c) where a is the center,
13. is vertex b part of loop?
16. does vertex b have punctuation mark?
17. horizontal distance between (b, baseline),
18. type of vertex d,
19. is vertex d part of loop?
20. does vertex d have a punctuation mark?
21. horizontal distance between (d, baseline),
22. the cosine angle between (b,d),
23. the angle sine between (b,d),
24. the cosine angle between (a,d) where b is the center.

3.2 Designing and Training of Neural Network

As mentioned before, a system of Neural Network used to segment the Arabic words and syllables. A special neural network is built for every pair of the skeleton vertices that belongs to the division set instead of using one neural network for two objectives. The first one is to enhance the network performance, because using eight neural networks in a mission more specifics to each network instead of using one network for all locations. The preparation for the training set becomes easier. Also, it simplifies the operation and analysis of every network work.

The second objective of building neural network is related to the difference in the number of features that can be extracted for each pair of the training set according to the type differences. The size of the inputs layer must be fixed in the neural network, even if the size of the input layer is equal to the features and their effects between the vectors different. Table 3.4 illustrates the vectors, neural networks (NN1, NN2... NN8) and the sizes of associated vectors.

Table 3.4:Types of neural networks and the vectors with their sizes.

Neural networks	NN1	NN2	NN3	NN4	NN5	NN6	NN7	NN8
Vectors	V1	V2	V3	V4	V5	V6	V7	V8
Vector size	24	31	38	31	38	38	23	15

For each input vector, we build a special neural network. That is for each Vi there is a correspondence NNi , $1 \leq i \leq 8$.

3.2.1 Normalization of Features Values

A Neural Network can establish the function between inputs and outputs, even if their relation is nonlinear. It is possible primarily to input the raw data directly to the Neural Network and find the final result. Practically, using this procedure leads to poor results and put limitation on reaching the generalized stage quickly. Consequently, the training process will be harder (Bishop, 1995).

Most applications require implementing suitable representations before starting the training stage and that's what is called normalization or input preprocessing.

When the input values are connecting real numbers as in HACRS; the normalization process is linear transfer for the inputs to express it in a predefined limited range. (Bishop, 1995).

Also some applications require processing the network outputs by finding the suitable representation for the problem in question and that's what is called post processing. Figure 4.1 illustrates this.

Normalization of the input feature value is done to accomplish two goals:

1- To set bound of features values to a fixed interval (-1,1).

2-To reduce possibilities of having the zero-valued features.

Its better not to represent the input feature to the neural network with the value zero. To facilitate and increase the training process speed. Using the zero- valued features might confuse the work of the neural network during training (Bishop, 1995).

The features in HACRS will not take the value zero, except for trigonometric functions (sine and cosine). The features values normalization is as follows:

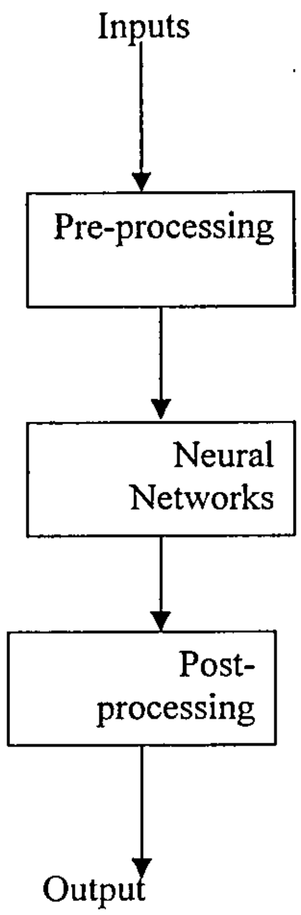


Figure 3.10: input, pre-processing, neural networks, post-processing and output.

1. **Bi-valued features:** The features have the value (-0.9) if the features not exist and the value (0.9) if it exists.

   The bi-valued features are

   a- The skeleton vertex is part of a loop.

   b- The skeleton vertex has a punctuation mark.

The rest of the features will be normalized as follows: (Bishop, 1995).

$$\text{The normalized value} = \frac{\textit{original value}}{\text{the maximum value the feature can have}}$$

2. **The value of the type of the skeleton vertex** (number of neighbors): will be normalized as follows:

$$\text{The normalized value for the skeleton vertex type} = \frac{\textit{number of neighbors}}{4}$$

Notice that the value 4 is the maximum value this feature can have. Table 3.5 gives the values after normalization.

Table 3.5: Normalization values of vertex type,

Features	Values
End point	0.25
Connected point	0.5
Branch point	0.75
Intersection point	1.0

3-The vertical and horizontal distance between a pair of skeleton vertices feature: This normalized as the previous rule in 2.

$$\text{The normalization value for the horizontal distance} =, \frac{\text{horizantal distance}}{\text{skeleton width}}$$

and

$$\text{The normalized vertical distance} = \frac{\text{vertical distance}}{\text{skeleton hieght}}$$

The primary goal of the features normalization is to restrict the feature values in the interval]0,1[. It also makes the feature indicator to more powerful, since it doesn't represent the vertical or horizontal distance, which changes according to the writing size. It represents the ratio between this distance and the skeleton size making this feature less sensitive to the change in the writing size. Figure 3.11 illustrates how to compute this feature.

4- Base line feature value. This is normalized as follows:

If the skeleton vertex higher than the base line, then

$$\text{The normalized feature value} = \frac{\text{vertical distance}}{\textit{base line}}$$

And if the skeleton vertex under the base line, then:

$$\text{The normalized feature value} = -\frac{\text{vertical distance}}{(base\ \ line\ \ - skeleton\ \ heigher)}$$

The base line feature is defined within the internal]-1,1[. It has a negative value when the skeleton vertex is under the base line. This feature indicates two things, the relative distance of the skeleton vertex to the base line, and the location of that vertex under or above the base line. Figure 3.11 illustrates how to compute this feature.

5- The Trigonometric Function: It is defined, previously, on the limited range.

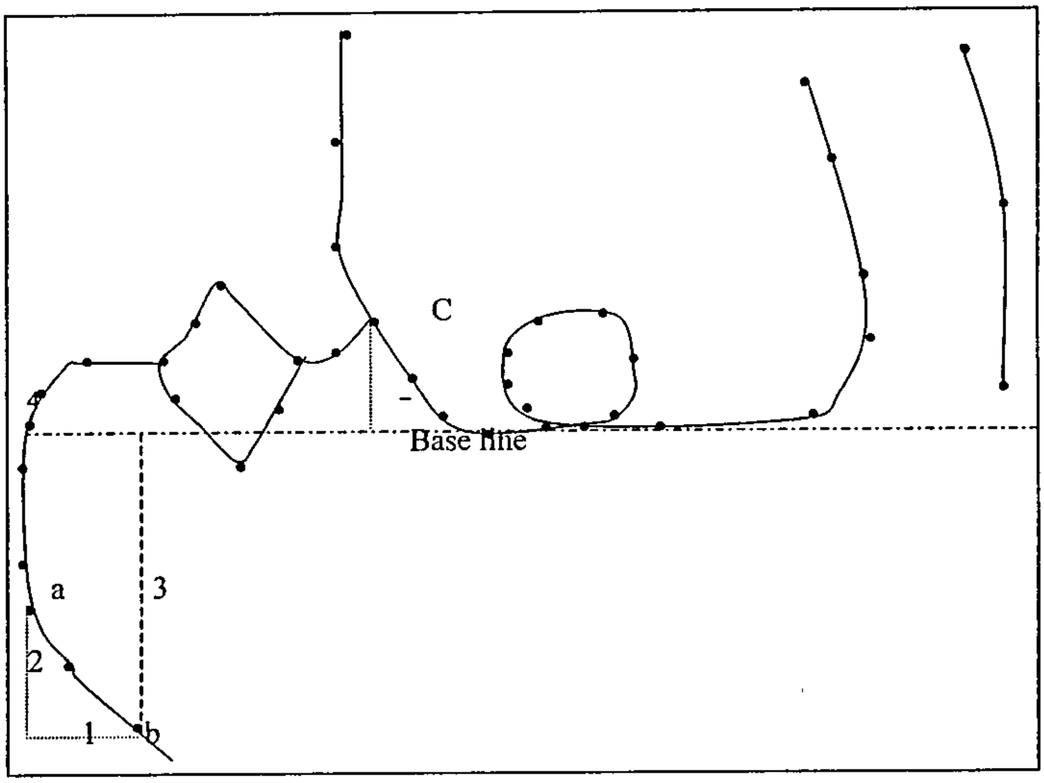


Figure 3.11: Baseline and distance feature values normalization, where 1: horizontal distance between a and b, 2: vertical distance between a and b, 3: vertical distance between b and baseline and 4: vertical distance between c and base line.

In Figure 3.11, the vertex c is above the baseline, so dividing the value into the baseline value (upper part of the image) normalizes the baseline feature value. The vertex b is under the baseline, so its value is divided into (-1)*(image height – baseline). The negative value indicates that the location of the vertex is under the baseline. Dividing it into the image height and width subsequently normalizes the vertical and horizontal distance feature values.

3.2.2 Designing and Organization of Neural Network

The Feed Forward Error Backpropagation Neural Network BPNN is one of the mostly used NNs in pattern recognition. The training is done through under supervised learning algorithm. Figure 4.3 illustrates the general skeleton of the used NN, the NN consists of 3 layers: (1) the input layers which is a traditional passing layer with a number equal to the number of features entered, (2) the hidden layer, which contains a number of neurons prepositional to the size of input layer, (3) and the output layer represents a required decision function and contains one neuron.

As shown in Figure 3.12 the NN is fully connected between every consecutive layer. The $w^h ij$ is the weight used between the neuron j of the input layer and the neuron i at the hidden layer. The neurons in the hidden layer and the output layer are connected with a bias factor. The $B^h i$ represents the used weights between the Bias factor of the hidden layer and the neuron i. The $B^o i$ represents the used weight between output layer and neuron i.

The hidden layer uses Sigmoid function to find its output within the interval (0,1). The activation function for the hidden layer neurons and the output layer perform a standard accumulation.

The neuron in the output layer is the final output for the network and is calculated using Sigmoid function within the vertical (1,0). The neural network is uses one type of decision, which is a bi-valued function produces definite answer "yes" or "no". The answer is "yes" when the function value is greater than the threshold T1; and the answer is "no" if the value is less than T2. A Sigmoid function has the following general formula.

, $f(x) = \frac{\gamma}{1+\exp(-\sigma x)} - \eta$ where- σ , $\eta = b - a$ Slope of f (x), and the made Xmax has a value equal to a specific part of the distance between a and b .(η) In the case where the inputs value trapped within the interval (-1,1) and the output values in the interval (0,1), then
a=0 and b=1.
The entries represent the features. The size of the input layer is between 15 and 38. The size of the hidden layer is between 8 and 19. This size depends on the type of the neural network.
Suppose that Xmax =1,Xmin=-1, ,1- η a=0, thus the function f has two forms as shown in equation (4.4) and (4.5) .Solving the equations for, δ we get.2.197225- δ
This means the final formula for sigmoid function will be equation (4.4)

$$f(X\max) = \frac{9(b+a)}{10} \quad (4.4)$$

$$f(1) = \frac{1}{1+\exp(-\sigma)} \quad (4.5)$$

$$f(x) = \frac{1}{1+\exp(2.197225x)} \quad (4.6)$$

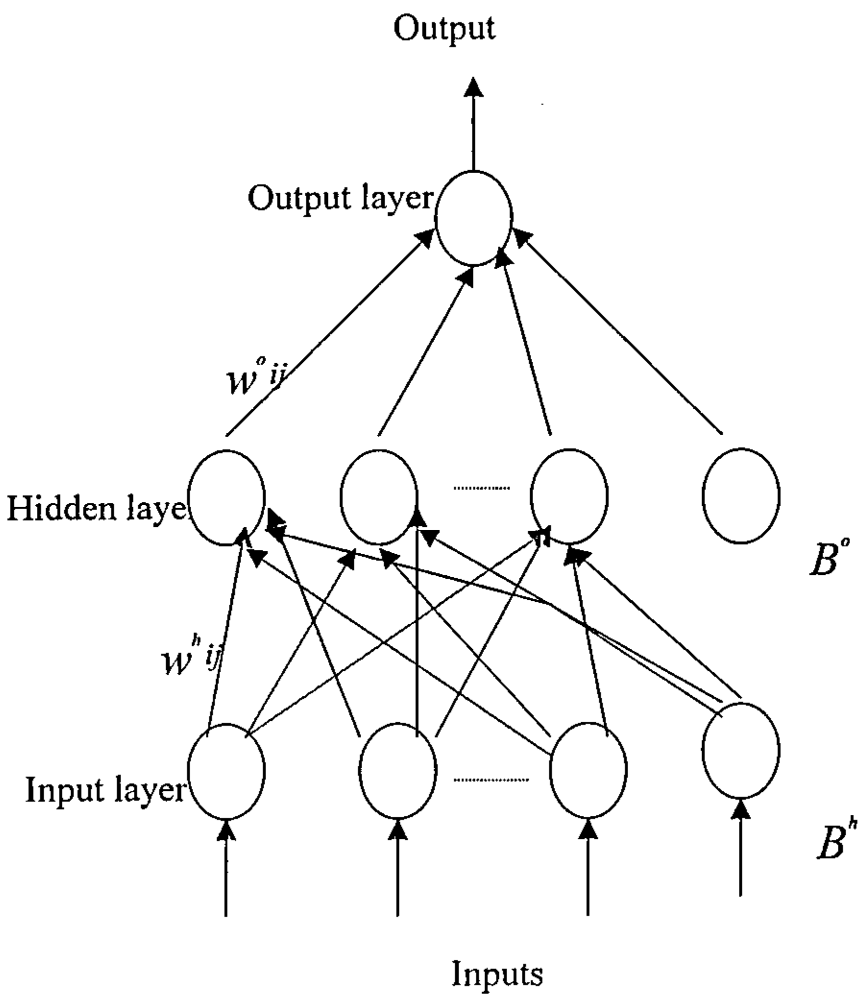


Figure 3.12: The connected NN. The arrows represent the weights between the neuron, while the circles represent the neuron.

During training of the NNs, the value (0.9) indicates that segmentation should be done to the given pair, and the value (0.1) to indicate that there shouldn't be segmentation. This method measuring the vector in (0.1,0.9) better than measuring it in (0,1), since sigmoid function doesn't take values (0,1) (Freeman and Skapura, 1992).

3.2.3 Training of Neural Network

The neural Network has been trained using algorithm of Momentum Error Backpropagation. The training process is measured in epochs. An epoch is a training cycle in which all examples in the training set will be presented to the NN. Training takes many epochs. There are a lot of decisions that the designer should specify concerning the size of the NN and choosing the variables values when training a NN.

Parameters Values

One of the most ambiguous in using the neural network is setting the parameter values; whether it is the initial values that will be modified during training or the fixed values during training and afterwards. These choices are submit to a personal subjective and it also depends directly on the problem nature. The choices have different effects on the training NN process or the work of the NN as a whole.

Choosing Learning Parameter η and the Momentum Factor

As we mentioned in Chapter 2; choosing the learning parameter η has a large effect on the work of the neural network. The learning parameter takes a value within the interval (0.01-0.25) Choosing a small value for the learning parameter means making a simple modification on the weights every epoch while training. The network needs a lot of epochs to reach the solution, which results in more time for the training. While increasing η leads to speed up, the training and not the ideal solution increases because the large jumps make the NN jump over the ideal solution (Freeman and Skapura, 1992).

In the HACRS, many different values for the learning parameter were used according to the training loops and the types of the NN. Table 3.6 shows that the final learning parameter value is relatively small. The reason for this choice is to accomplish constancy in modifying the NN weights during training.

To speed up the training process and to decrease the effect of the irregular exemplars on the weights, a momentum factor m has been used along with the Momentum Error Backpropagation. Table 3.7 shows a summary of the work of the momentum factor includes entering percentage of the previous weight modification for every training exemplar to monitor the modification process in a fixed direction. The momentum factor takes a small value close to (0.5). The value 0.6 was used for all the NNs used in this research (Freeman and Skapura, 1992). The initial training values of weights (-0.5,0.5) granted nonfailure training.

Table 3.6: The learning rate used in training the eight NNs training.

Neural Networks	NN1	NN2	NN3	NN4	NN5	NN6	NN7	NN8
Learning Rate	0.1	0.1	0.05	0.1	0.05	0.05	0.1	0.05

Table 3.7:The variables and their values were kept the same in eight neural networks.

Variable	The use value
Momentum Rate	0.6
Maximum epochs	10000
Sigmoid functions	2.197225
Initial weights	(-0.5,0.5)

Generalization Versus Memorization, and Halting

One of the best features of the Backpropagation Neural Network is its ability for Generalization. The generalization, here, means that if this NN is given several and different input vectors (belong to the same class), then it can find the similarity of these vectors and consequently separate in related classes. The generalization feature is important and its what determine the ability of the NN to deal with vectors for the first time, because it dealt with similar vectors during training. In the other hand, memorization is the ability to retrieve the data that the network had trained on to an acceptable degree. The designer must balance between those two features and that's done through the convenantial time to stop the training process.

There are many ways to stop the training. In this research the absolute error is being used. It is based on the fact that the training of the neural network continue till the error average for all the training vectors reaches a small acceptable value. Using this rule might lead to prolong the training time to no avail; so a limit must be put on the number of epochs. A small value of an error with 0.1 means the training stop if the error average reaches a value less than 0.1 or if epochs reach the upper limit. Also during training, the irregular examples are monitored with average never come down. It is possible to get rid of some of these exemplars that the neural network can't classify.

Size of Hidden Layer

The size of hidden layer is not always so clear as in the input and output layer. The size of the input layer depends directly on the number of the extracted features while the size of the output layer depends on the number of classes and their represented ways. The size of the two layers that deals with the outer world (according to the neural network) is predefined in regard to the nature of the problem.

The size of the hidden layer doesn't depend on the input and output only but also on the nature of the relation between the element that forms the input vector. During the neural network training, the neurons of the hidden layer organize thierselves in away that make some of the neurons specialized in recognizing some features of the input vectors. When the number of neurons is not enough, some features are ignored. If the size than what the hidden layers are larger of the neural network needs, these leads to finding neurons that don't learn. This can be noticed through the weight of these neurons and if they are changed during training.

The major idea in choosing the size of hidden layer is to use the minimum limit of neurons in the hidden layer. The increase of the hidden layer's size without improving the performance of the network decreases the speed of the network and increasing the time that the training process require.

In general, it is better to use a hidden layer with a size not more than 50% of the input layer unless a need to use bigger size is arise (freeman and Skapura, 1992). In HACRS, choosing a size of the hidden layer equal half the size of the input layer was suitable for all the used neural network except NN5 which couldn't learn except when choosing a larger size. Table 3.8 illustrates the hidden layer for the eight neural networks.

Table 3.8: The size of the used neural networks in training (or testing) HACRS, the size of output layer is one neuron in all the eight neural networks.

Neural networks	Size of input layer	Size of hidden layer	Number of weights
NN1	24	12	313
NN2	31	16	529
NN3	38	19	761
NN4	31	16	529
NN5	38	22	881
NN6	38	19	761
NN7	15	8	137
NN8	23	11	276

The weights in the table contain those weight related to the Bias factor and computed as follows:
Number of weights =the size of the input layer *(the size of hidden layer +1) + the size of the hidden layer *(the size of output layer +1).

Preparing the Training Sets

Preparing the training sets is done from samples of Arabic Handwritten gathered from random groups of schools, army and mailing addressees from Ministry of Transport, Posts and Communication. The images were entered using a scanner. A generalization process was done for every image. The major components and the secondary components were recognized. After finding the skeleton and extraction features for each connected component, these feature were exposed to a special software that is been equipped for the purpose of preparing the training sets.
Training set for every network consists of pairs of vectors called Exemplars. An exemplar consists of input vector and goal vector that

represents the decision function. The designer gives the value of the decision vector. A value of (0.9) is given to the segmentation (0.1) to the non-segmentation, and a value of (0.5) in case of unsureness.
Table 3.9 shows the sizes of the special training sets for every neural network. The differences in the sizes of the training sets resulted from two reasons. The first one is the range of changes that affect the input vectors. The wider the change, the bigger the need to prepare large training sets that could cover a large number of input vectors. This helps the neural network in the work stage to deal with such cases. The other factor is that the size of the training set depends on the retriations of the appearance of the vector types that the segmentation set contains.
The vectors don't appear equally. Figure 3.13 shows comparison between the iteration of the appearance of the eight vectors that the whole segmentation group contains. The pair V1 is the most reiterated and forms 60% of the whole vectors. the vectors V3 and V6 are the least iterated and form less than 1% because the intersection vertex form one of the pair that represents vectors V3 and V6 are appearing less in the Arabic skeleton. The decision of no segmentation is high for the sets that related to NN1, NN7, and NN8 (these contain end points). On the other hand, the decision of segmentation for the sets of NN2 and NN3 is very high.

When preparing the training sets, observation is done for all existed vectors. The most reiterated vectors are vectors are excluded. In case of vectors segmentation is difficult to determine a value of (0.5) is given to the goal vector.

Table 3.9: Groups of used neural networks training and their sizes and ratio.

		Decision with segmentation		Decision with no segmentation		Not sure	
Neural Networks	Size of training set	Size	Ratio (%)	Size	Ratio (%)	Size	Ratio (%)
NN1	23974	1558	6.5	21577	90	839	3.5
NN2	4491	2627	58.5	1527	34	337	7.5
NN3	130	89	68	41	32	0	0
NN4	5146	2084	40.5	2856	55.5	206	4
NN5	1035	407	39.3	263	60.2	5	0.5
NN6	176	57	32.4	119	67.6	0	0
NN7	2544	76	3	2381	93.6	87	3.4
NN8	1326	125	9.4	1196	90.2	5	0.4
Total	38822	7023	18.1	30320	78.1	1479	3.8

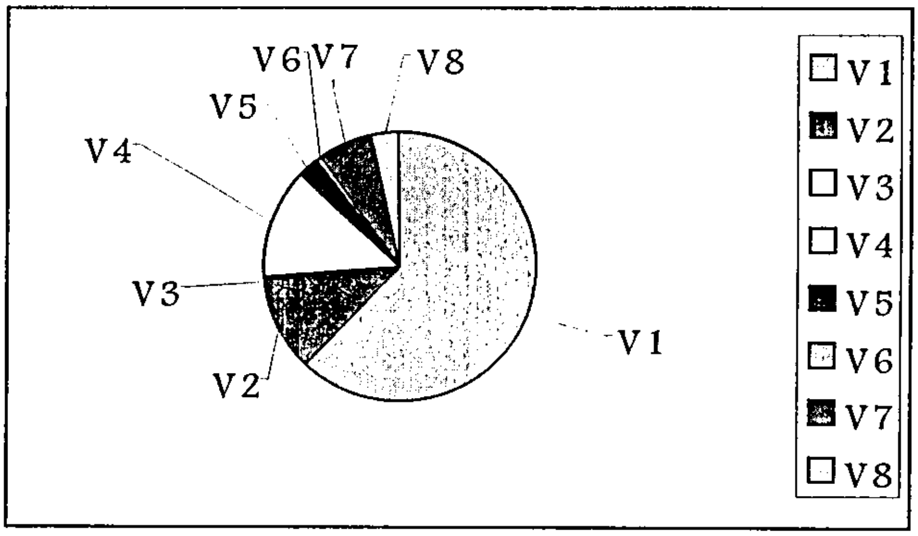


Figure 3.13: The iterated appearance of vector types in the Arabic words Skeleton.

3.2.4 Results of Training NNs

Table 4.7gives results for the final training results of the eight neural networks. There are large differences between the number of epochs of training process between the neural network. The differences are normal and resulted from the different sizes of neural networks and the nature of the relation between the inputs (features) and the segmentation decision. When the relation is linear, the NN can learn fast (Freeman and Skapura, 1992). The training process is also affected by other factors such as learning factor, size of training set momentum factor, and the initial weight values. The training processes were carried out on a personal computer with clock speed 350 MHz. Of course, the speed of the training process will differ where you use different machines. The time in Table 3.10 is measured in seconds.

Table 3.10: The results of training the 8 Neural Networks.

Neural Networks	NN1	NN2	NN3	NN4	NN5	NN6	NN7	NN8
Epochs	18200	6586	324	5879	719	228	11544	5187
Time (s)	25736	1096	194	13446	1083	1129	195	5951

3.3 Classification

This stage is considered the final phase in character recognition. After the segmentation of the words and sub-words into characters, each letters is stored in a special file. This phase becomes the complementary of the other phases and stages. It is supposed that the segmented characters produced from the segmented stage enter the Neural Networks that are designed to classify these characters. This couldn't be done for the following reasons.

1. In the segmentation stage, the punctuation marks wasn't completely recognized; this was a weakness in recognizing (2 dots, 3 dots) and in recognizing (3 dots, hamza).
2. It is difficult to find a text that contains all the shapes of the Arabic characters (102 shapes) (i.e. to get a 200 samples for every possible shape).
3. In the segmentation stage, there was no restriction on the font size. One writes with 15x12-font size another with 40x30 ... etc. On this stage, the characters must have fixed sizes such as (20x20).
4. There is an error in the segmentation stage. If this error is entered to the Neural Network, the error will be propagated to more error and consequently recognizing the characters will be difficult.

For all these reasons, the results of the segmentation stage couldn't be used for learning the Neural, instead a sample of isolated characters has been collected and after the initial preprocessing, they were entered to the Neural Network. After that the training and testing of the network were done.

In the classification phase ,three types of NNs were constructed, one of them has (28) neurons, the other (102), and the third has (15).

3.3.1 Preliminary Processing

In this section, the acquisition of raw images, the preprocessing operations are described. The following operations are carried out:

Data Acquisition

All data were gathered from random persons coming to the Department of Civil Status and Directorate of Passports.

There is a standard sheet prepared for this purpose. Its size is 20x20 pixels. The gathered data were scanned off from input documents (sheets) using HP scanner. The image is stored as binary bitmap file, in which the '0' pixels belong to the background and the '1' pixels belong to the image patterns. For correct sampling in the scanning process, it is advised that at least 300 dpi (dots per inch) to be used. However, 300 dpi slows down the scanning of the sheet and takes a considerable amount of memory. Therefore, resolution of 150 dpi (or even less) can be used.

Smoothing and Small Hole Filling.

Some preprocessing operations are necessary for the scanned images to make the input in a form manageable by further processing modules, and to meet some requirement imposed by certain system modules.

The smoothing method proposed here is used by Atic (1994). Here 4 masks are used to remove isolated pixels and peaks; 4 others masks are used to fill in holes. See Figure 3.14 and Figure 3.15 .The masks are 2x3 and 3x2.

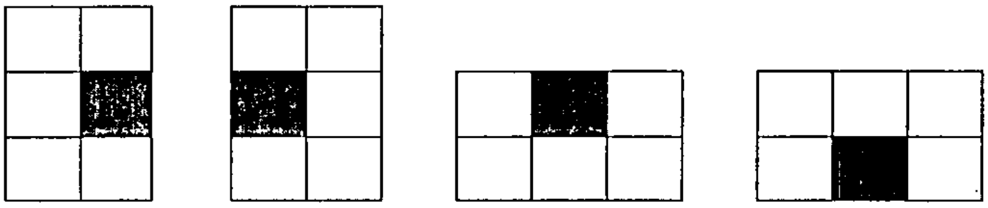


Figure 3.14: Masks to remove peaks.

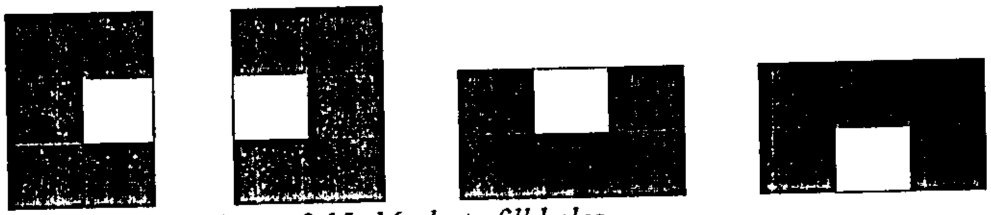


Figure 3.15: Masks to fill holes.

A certain pixel in a mask is referred to as the seed point. The seeds of the masks in Figure 3.14 are marked with black (foreground), and the seeds of those in Figure 3.15 are marked with white (background).

The equality of a seed and a pixel on compound structures (characters) is checked. If they match, the pixels around the current mask are checked. If they match too, the value of compound structure pixel is reversed (i.e. foreground changed to background, or vice versa). This operation is applied to every pixel in the compound structure.

Localization

There is a limitation on the maximum size of a letter (20x20), although most of the characters are written with different size (10x16), (12x15), etc. Characters start from the same point or end at a definite point. In this research, it is assumed that all characters begin at (0,0). Thus a special algorithm has been applied to shift the characters to the upper left side. Figure 3.16 shows different locations of a character.

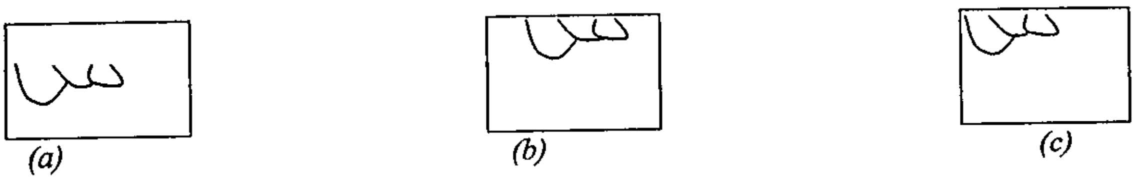


Figure 3.16 The locations of a character in a rectangular) :a) the letter was written in the middle, (b) the letter was up dragged ,and (c) it was dragged to the left.

3.3.2 Preparing The Training Sets.

After gathering the isolated handwritten Arabic letter as previously mentioned, all they were converted to Bi-image. The preprocessing follows. The training sets comprise of two vectors, one is the entry vector and the other is the target vector.

In this stage, a different training sets has been done:

1. Training set consists of training examples over (28) Arabic characters without applying the pre-processing process. And training set after preprocessing.
2. Training set consists of training examples over (102) Arabic characters shapes after applying the smoothing algorithm.
3. Training set contains only (15) characters, after excluding all similar characters as mentioned before.

As a result of the different goals of the Neural Network, the target vector was different for each neural network. One time the vector represents the 28 Arabic characters. Another time, it represents the Arabic characters after excluding the similar characters (15 characters). Table 3.11 shows the sizes of training and testing sets with and without preprocessing for 28 Arabic characters. Tables 3.12 and 3.13 contain the sizes for 102 and 15 Arabic characters, respectively.

Table 3.11: The sizes of training and testing sets for 28 characters before and after preprocessing.

Experiment no.	Training sets	Testing sets
1	50	5
2	50	5
3	100	10
4	100	10
5	100	10
6	100	10
5	100	10
8	200	20

Table 3.12 :Data sets containing 102 Arabic character with preprocessing.

Experiment no.	Training set	Testing set
1	50	5
2	100	10

Table 3.13 :Data sets containing 15 Arabic characters (exclude the characters that have the same body) after applying smoothing algorithm.

Experiment no.	Training set	Testing set
1	90	10

3.3.3 The Designed Neural Network

The Resilient Backpropagation is used in this stage. Three Neural Networks have been built according to the goal that is required from the NN.

The NN in Figure 3.17 for the 28 characters, in Figure 3.18 for the 102 characters, and in Figure 3.19 for the 15 characters.

In Figure 3.17 ,the neural network consists of five layers. The input layer, which is a traditional layer, has the same size of the input vector (20x20), (400 neuron).

There are three hidden layers, because the Neural Network couldn't be learned with one or two hidden layers. Each hidden layer consists of 100 neurons.

Finally, the output layer consists of (28) neurons representing all the Arabic characters (28).

Figure 3.18 looks exactly like Figure 3.17 except that the output layer is different. There are (102) neurons, representing the shapes of the Arabic characters.

Figure 3.19 represents the NN structure, with only one hidden layer consists of 100 neurons, one output layer (15 neurons),to recognize the Arabic characters after excluding the similar ones. Table 3.14 shows some parameters and their values. Table 3.15 illustrates the number of weights in the neural networks.

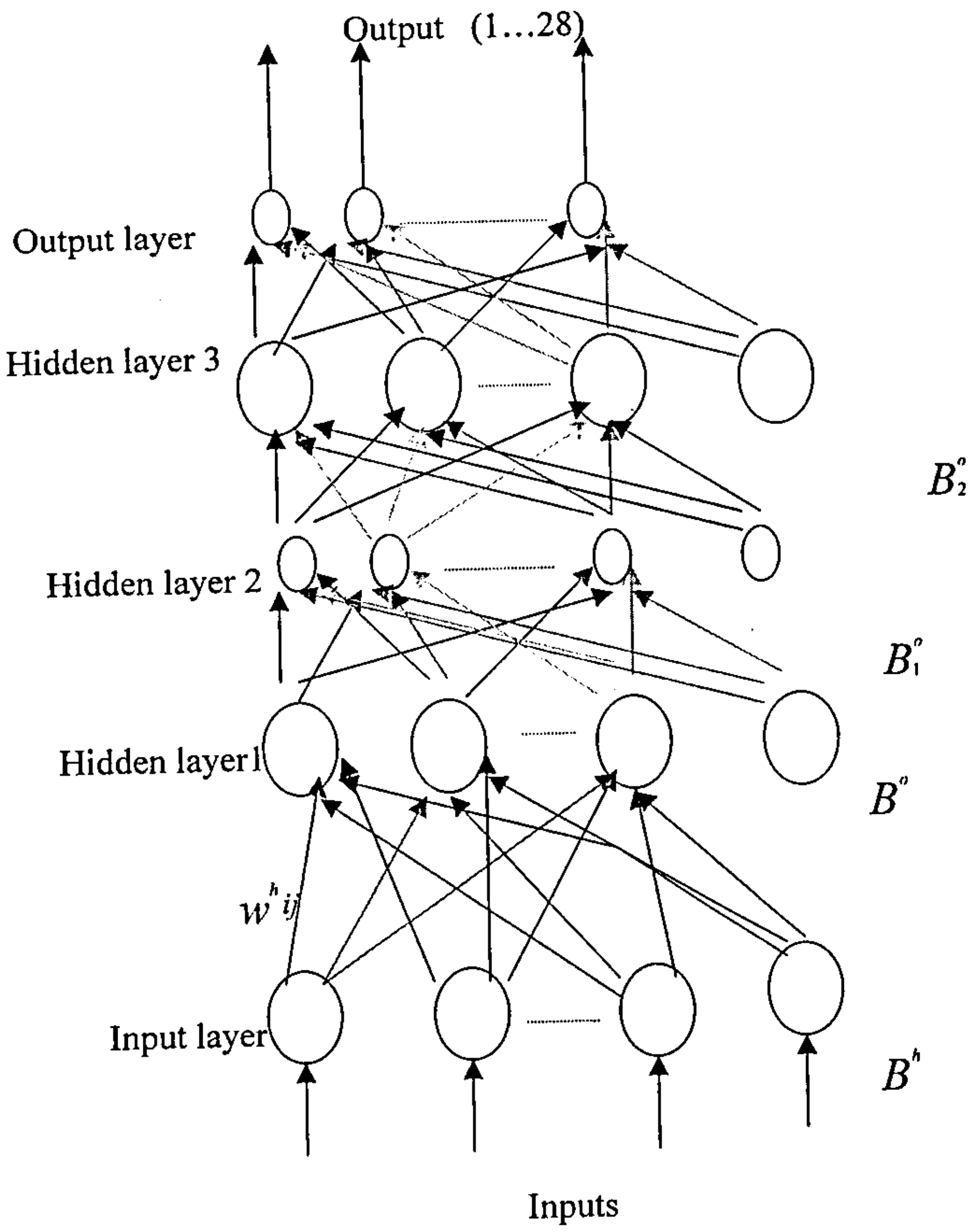


Figure 3.17: The architecture of the neural networks to recognize the separated 28 letters.

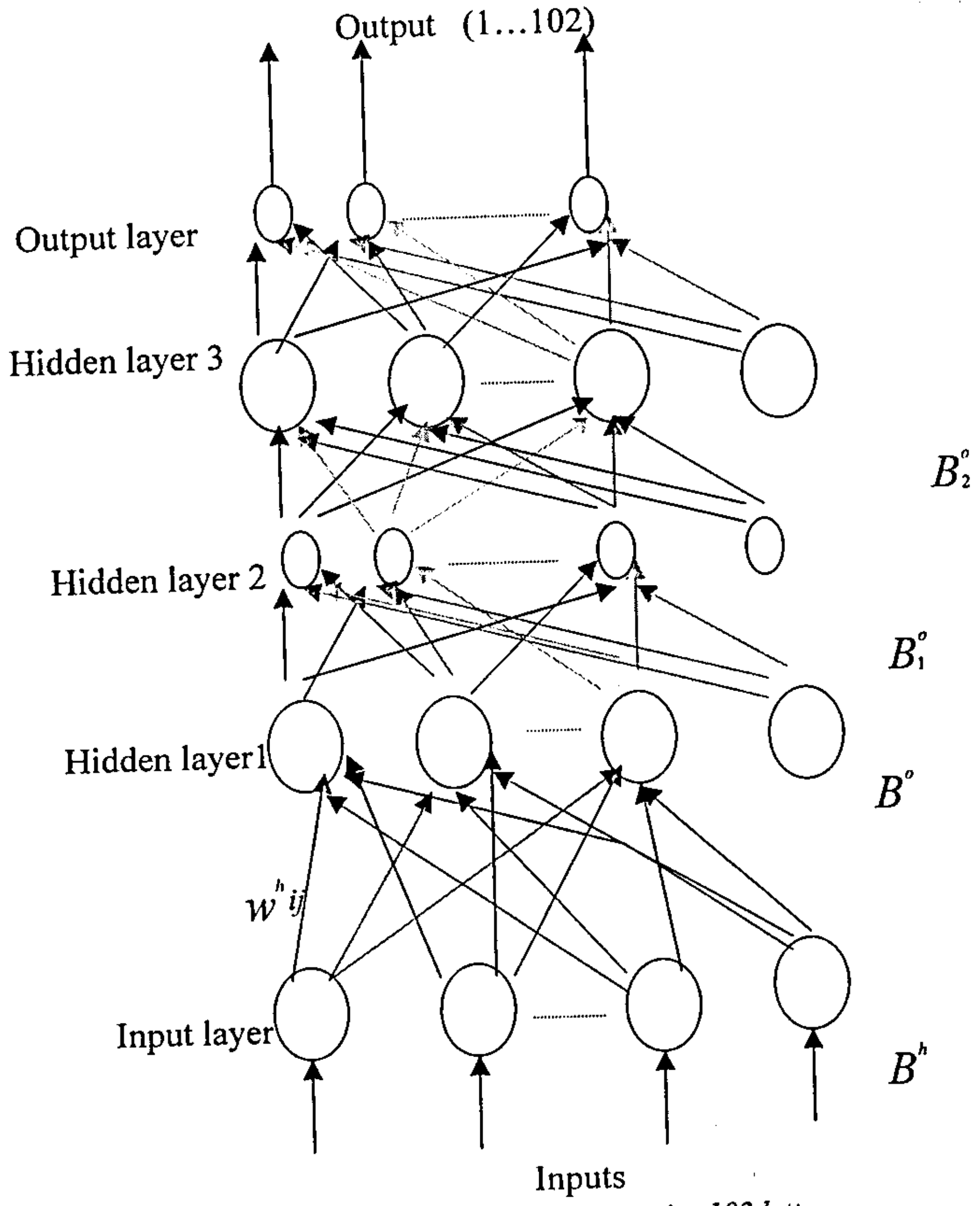


Figure 3.18: The architecture of neural networks to recognize 102 letters..

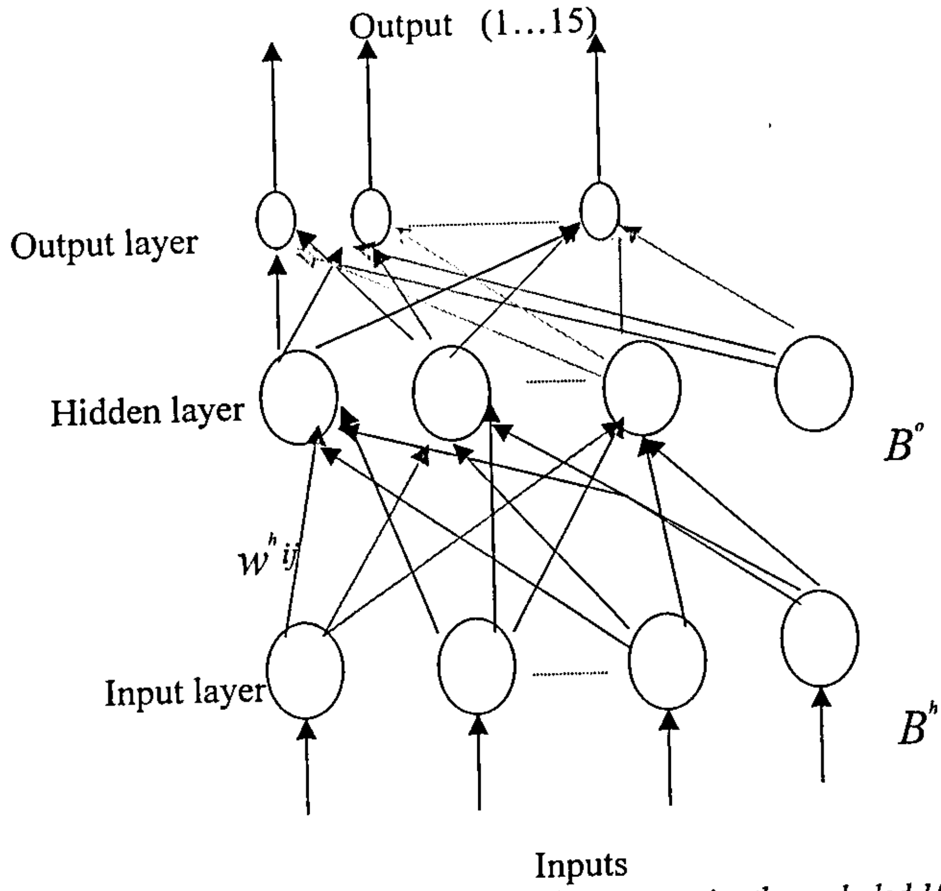


Figure 3.19 :The architecture of the neural networks to recognize the excluded 15 letters in Arabic handwritten characters.

Table 3.14: Variables used in neural networks.

Parameters	Value
Learning rate	0.05
Maximum number of epochs	100,000
Initial weights	[0.5,0.5-]
Performance	1x10E-6
Time	∞

Table 3.15: Total of weights in the three neural networks.

Type of neural/layers	Neural with output 28	Neural with output 102	Neural with Output 15
Input layer	400	400	400
Hidden layer no.1	100	100	100
Hidden layer no.2	100	100	-
Hidden layer no.3	100	100	-
Output layer	28	102	15
Total weights	63134	70608	41615

Chapter 4

Results and Discussion

4.1 Digitizing and Smoothing

Arabic text images are digitized using a scanner with 600 dpi, which is enough to handle the text in general. The image is converted to Bi-image with two gray levels, where the black pixels represent the written text with value (1), while the white pixels represent the background with value (0). Around 2000 images where entered, containing more than (5000) components or words. These images form the training and testing sets of neural networks.

The entered image is processed using statistical smoothing algorithm, developed in (Khellah and Mahmoud, 1994). Their results indicate that this simple algorithm is suitable for the Arabic text images. The smoothing process aimed to eliminate the noise and fill the small gaps appear in the images. It aimed to make any boundary of any connected area in the image 8.connected and it is a condition for the boundary-following algorithm to succeed. This research results indicate that applying the algorithm 4 iterations guarantee that the boundary of any connected area in that image will be 8.connected no matter what the boundary situation was.

The experimental value 5 for the threshold T in the smoothing algorithm was found and it gave the required results. Using a large value for T leads to fill the inner gaps and consequently to change its shape. Also using a small value will lead to a little smoothing. The algorithm is applied two consecutive times or more according to the noise and the nature of the entered image (Khellah and Mahmoud, 1994). Figure 4.1 represents a smoothed image produced by the smoothing algorithm. The noises disappear after the algorithm was applied.

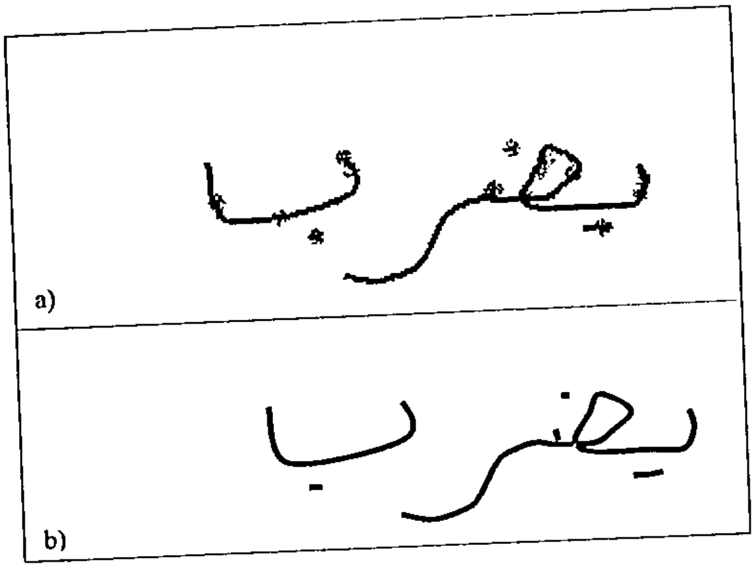


Figure 4.1:Smoothing using the statistical smoothing algorithm:
a) the image as entered, b) the image after applying the algorithm.

4.2 The Separation of Component and Secondary Recognition

The boundary-following algorithm can follow any connected area in the Bi-image under the conditions: the boundary cells (pixels) shouldn't be on the image edge and to be 4.connected. These conditions can be hold easily. The first condition can be established by adding a white pixel line on the image edges from the four sides. The second condition will be guaranteed by the smoothing algorithm (Jain et al, 1995).

This way insures the separation of words and components on the same line with insignificant errors; and without pre-condition on syllable size and the distance between them. Avoiding the errors in an early stage of work increases the efficiency of the system and makes the later steps much easier.

A secondary component is recognized using the algorithm, explained before where the experimental values for the thresholds were (T1=270,T2=100,T3=0.5,T4=2). This simple algorithm proved efficiency in recognizing secondary component from the primary components and

with an error percent less than 3%. Figure 4.2 illustrates the work of the algorithm on some examples.
The threshold T1 is used to recognize the component under examination. A component of size more than T1 is considered a primary component. A component of sizes less than T2 is considered a secondary component. Actually, it is rarely to find a major component of size less than 100 pixels; also a large portion of punctuation has size more than that.
Recognizing a component of size between T1 and T2 depends on two factors: firstly, the proportion between the length and the width of the components and has two thresholds T3 and T4. If the proportion is less than T3, then the primary component is mostly a small alef "ا". If it is more than T4, then the component is a secondary one. Notice that if the secondary component is two dots, then this is a noticeable difference between length and width. In the last case where the ratio between T3 and T4, we used the second factor. This factor is the ratio between the white pixels and the box size surrounding the components. If this ratio is greater than T5, then the major component is a letter (small Ra "ر", or small Dal "د"). If the ratio is less than T5, then the component is a secondary (small Hamza "ء" or three dots like a triangle).

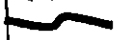
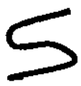
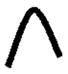
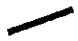
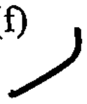
Pictures	(a) 	(b) 	(c) 	(d) 	(f) 
Size	241	134	172	117	215
Ratio 1	2.15	0.78	1.04	3.63	0.94
Ratio 2	0.73	0.53	0.74	0.57	0.89

Figure 4.2: An example on how the algorithm recognizes punctuation marks.

In Figure 4.2, the picture Ratio1 is the ratio of the length to the width and ratio 2 is the ratio between number of white pixels to the number of pixels in the rectangular box. Picture (a) represents two dots, which is a secondary component recognized by Ratio 1 and threshold T3. Picture (b) is hamza recognized by Ratio 2 ,as well as the triangle –three dots- in picture (c). The two dots in (d) is recognized by T4. The small letter “Ra” "ر" in (f) is recognized by ratio 2. A large part of the rectangle that surrounded the letter was white pixels.

4.3 Finding skeleton

The fuzzy ISODATA algorithm has been used to find the Arabic handwritten skeletons. This algorithm depends on clustering concept. In the research done by Mahmoud et al (1991), this concept was used in representing the separated Arabic characters and it proved convenience for the Arabic letters; beside being ahead of many algorithms the researcher compared between. This algorithm is distinguished by its ability to deal with noise, and the desired skeleton is unaffected with any slopes or deviations occurring in the way the letter is entered.

Finding a unique skeleton, no matter what the slope and the size of the letter, is a suitable feature of this algorithm and made it super over many algorithms. On the other hand, the algorithm suffers from two drawbacks:

The first is the great slowness in comparison with other skeleton finding algorithms. Any algorithm depends on grouping in clusters concept would suffer from slowness. The algorithm needs to run many consecutive loops in order to find the final clusters. The greater size, the more time is needed to find the skeleton. Researchers suggested that solving this problem require using the cluster-grouping algorithm at the learning stage and at the working stage it is possible to use other algorithms.
The second drawback is the initial selecting of the cluster numbers. This problem is simple if the primary cluster number is slightly above the require or ideal number. But, it might cause faults in the skeleton forming if the primary cluster number is less than the required number.
To solve the problem concerning the algorithm slowness, we add a step before applying the cluster-grouping algorithm, which is the thinning. This decreases noticeably the size of the word, consequently the cluster finding process will be faster. The famous Zhang and Suen (1984) algorithm has been chosen, which is a parallel thinning algorithm and been used by El-Khaly and Sid-Ahmad (1990) to thin printed Arabic words, where it produces a satisfying results.
The image produced from the thinning process represents the word's medial axis, which is a set of connected pixels with no more than 1 pixel in width. Notice from Figure 4.3 that the thinning algorithm could find the suitable medial axis but it's affected by gaps and deficiencies that could appear in the connected area that represents the word. This sensitivity towards the changes that might affect the pattern shape is a phenomenon all the thinning algorithm is suffering from (Gonzales and Wintz,1987).
In HACRS, the representation doesn't depend on the thinning process, because the final representation is done through the skeleton that the cluster concept algorithm has found. Figure 4.3 presents a comparison of finding a

skeleton with and without thinning and with the same initial cluster and at the value for the ending condition.

Two important remarks for this comparison worth mentioning:

1-Finding the skeleton using the thinning, took time 20% less than the time required for the original algorithm. Actually this percentage will decrease for a bigger size of words. In average adding the thinning step leads to decreasing the processing time to about 25% of the processing time of the original algorithm.

2- The size and the final shape for both skeletons are very close to each other. Actually, adding the thinning step leads in most cases to find a skeleton better in the locations of the clusters centers. This comes from the fact that thinning strips the pattern down to its medial axis. Consequently, the cluster-grouping algorithm isn't required to find this axis. The original algorithm deals with the image of the solid worked, and it has to specify the medial axis for the word and the cluster centers locations.

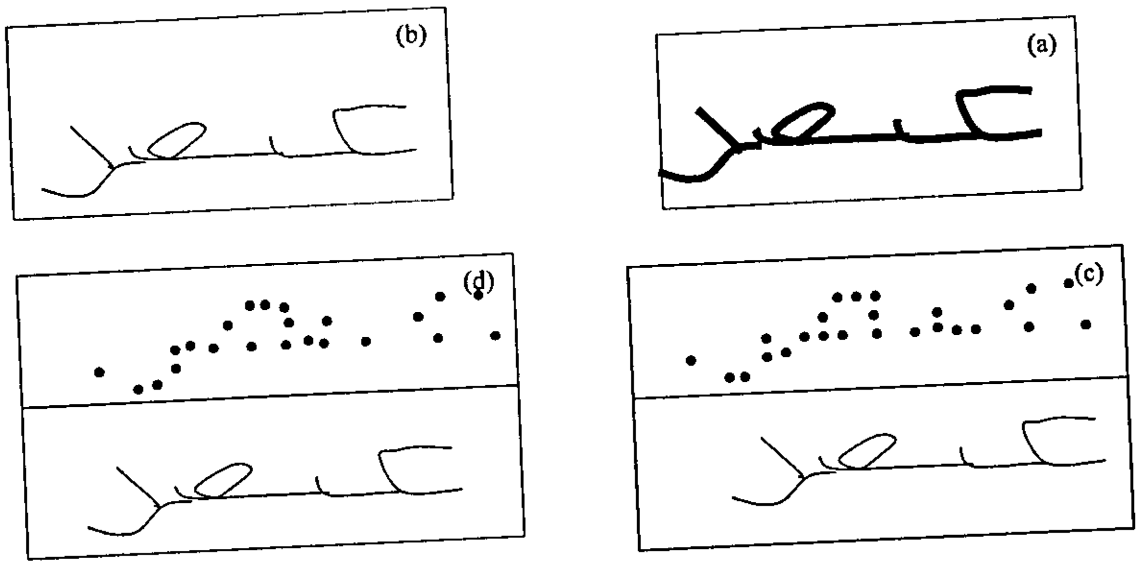


Figure 4.3: The effect of the thinning algorithm on finding skeleton.

Figure 4.3 shows the effect of the thinning algorithm, where , (a) the original picture, (b) the picture after thinning and (c) the top part is the cluster centers of the picture in (a) and the bottom is the skeleton of the picture. The top part in (d) is the cluster centers of the picture in (b), while in the bottom, the skeleton of the picture.

4.3.1 Initial Membership Assignment

The final skeleton extracted through the used algorithm is totally independent from the primary selection for the membership. This independently is rather important and it gives the algorithm the advantage of not being affected by slopping (Mahmoud et al, 1991).

The image of any word is entered with different slopes even if the slope is fixed no one can guarantee that images of the same word is identical because of the faults or flows affect. That means it's rather difficult that primary selection process is the same for the images of the same word even if the algorithm is one. And from this point, one can conclude that the skeleton independence feature using the primary selection earn the algorithm the unaffected slopping feature.

When comparing the time needed for the skeleton finding using the previous algorithm the c-consecutive slice strategy, is 50% faster, this is expected. In the c-consecutive slices strategy there is some kind of initial grouping for pixels that belong to the same cluster depending on the locations of these pixels which save the required time to find clusters. While in the c-interlaced slice strategy the pixels are distributed among the primary clusters in unorganized way. Therefore its recommended to use the consecutive slice strategy to save processing time while the interlaced slices strategy used to illustrate the algorithm capability to find clusters.

4.3.2 The initial selection of the cluster number

The method used to select the cluster's initial number depends on the size of the word or the letter. Generally the larger the size, the more clusters it

ction has been formed in the relation the initial
size of component (N)/Threshold (T). the
contains where 1=6 has been reached to give the required results.
number

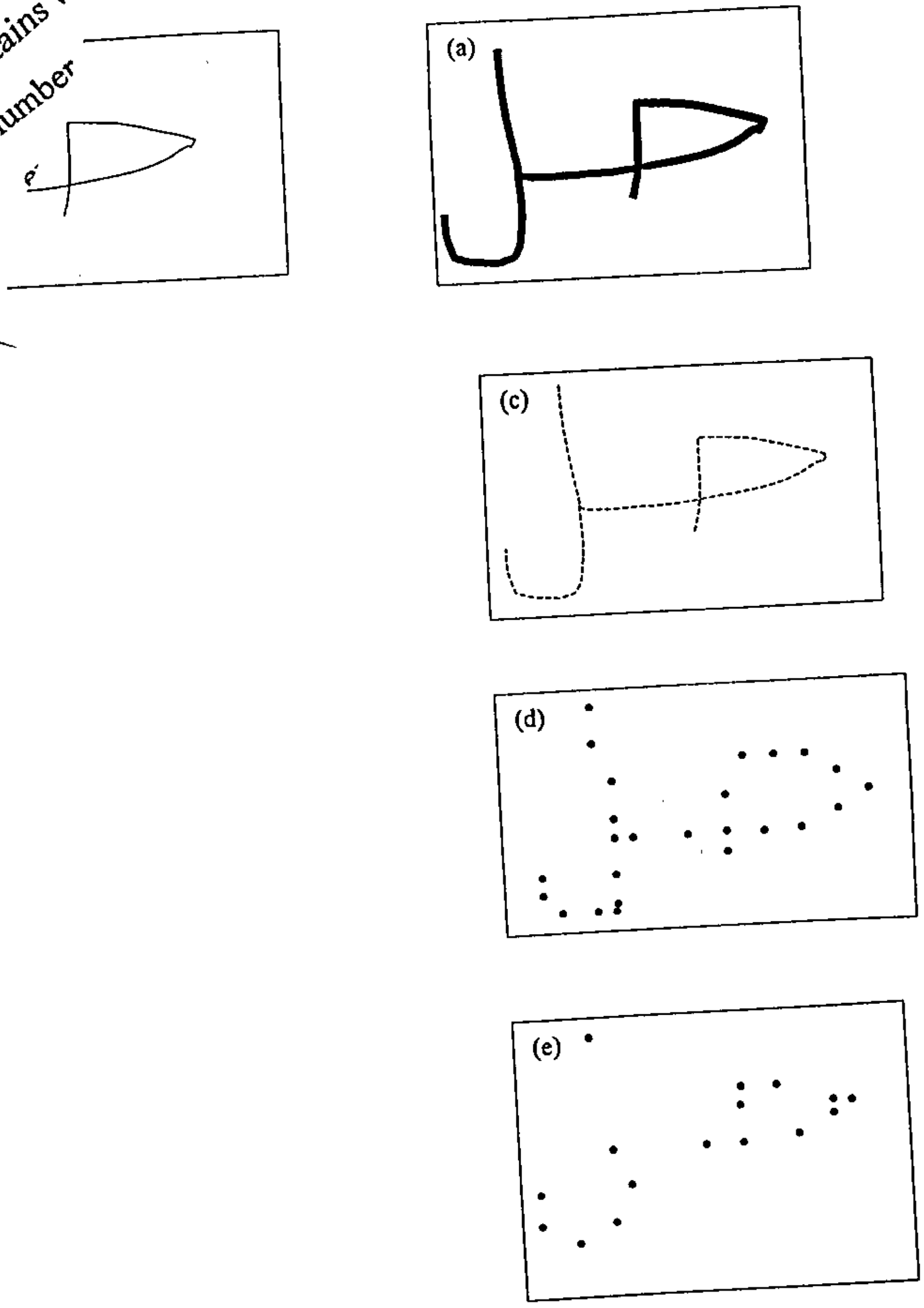


Figure 4.4: Finding a skeleton and reduction:
a) original handwritten word, b) after thinning, c) original cluster, d)centers of the original clusters, and e)final centers of the clusters.

4,
lin
xtra

mber. The
enters that should be
dd any useful information to
't affect the general
naving such clusters

is that the Arabic writing in general contains a number of lines with a fixed slopes, and from the skeleton side it forms one cluster, therefore deleting these extra clusters doesn't affect the skeleton forming.
The fuzzy ISODATA algorithm is capable to reduce the data and to represent the pattern in an easy way. The final skeleton is represented by cluster centers and the adjacent matrix, which makes the later, steps easier.

4.4 Feature Extraction

All features that contained in the entry vectors has been extracted from the words skeleton except for the punctuation mark which gives the features the characteristic of the low sensitivity towards the variations in the writing size or slope. For example, the slope grade feature is computed between the skeleton vertices instead of computing between the neighboring image pixels as done when using other representing methods. The distance feature is taken between 2 vertices of the skeleton as a proportion to the skeleton whole size, which in return decreases the effect of the line size, and so forth for the base line feature.
Finding the loops in the skeleton is done using constrained depth-first search algorithm. This algorithm is highly qualified where the error percentage is recorded less than 1% and it could find any loop no matter where the loop is or its size.
The punctuation feature is used as a proprietary feature for the skeleton vertices. A skeleton vertex is considered having a punctuation if it is closer to the other vertices than to the punctuation mark. The experimental results proved that it is better to use the vertical distance measure and city-block distance measure between the skeleton vertex and the punctuation mark centers. Because of the nature of the relation between the punctuation marks locations and letters in the Arabic writing, it is noticed that the punctuation is vertically easier to the letter that it belongs to in most cases.

The features used is different in its importance according to the neural networks, where the loops feature is the most important one and that can be noticed from the weights values attached to this feature, where the slope feature is considered the least participating in decision making.

4.5 Testing the Segmentation Neural Networks

Developing the neural network is done through learning process and consecutive testing , in which modifying some variables is carried out to ensure that the neural network reached the required results. In this Chapter, the final results of the neural network testing will be presented along with the some explanatory examples.

The testing sets for the neural network where prepared from 2000 words or sub-words , contains around 35000 vector distributed among the eight neural networks. Table 4.1 shows the success rate for every network, as well as the error rate and the rejection rate.

The separation decision is rejected if its in the interval(0.45,0.55) for all the used neural networks. Partition is made if the decision value is more than 0.55 and is derived if it's less than 0.45. As shown in the table, the rejection rate is low for all neural networks. The rejection was considered a sort of error because the neural networks couldn't answer. The rejection rate indicates the cases where the network decision wasn't right. The success rate reached 100% in NN3 and NN6. This return to the limited varieties in the entry vectors, which make the neural work, is easier.

Table 4.1 Testing neural networks results.

Networks	Testing sets size	Success rate (%)	Error rate(%)	Rejection rate (%)
NN1	25000	91	8	1
NN2	2800	81.5	17	1.5
NN3	120	100	0	0
NN4	4220	85.2	14	0.8
NN5	520	82	18	0
NN6	170	100	0	0
NN7	1450	90	8	2
NN8	810	92	7	0.5
Total	35090	89	10	1

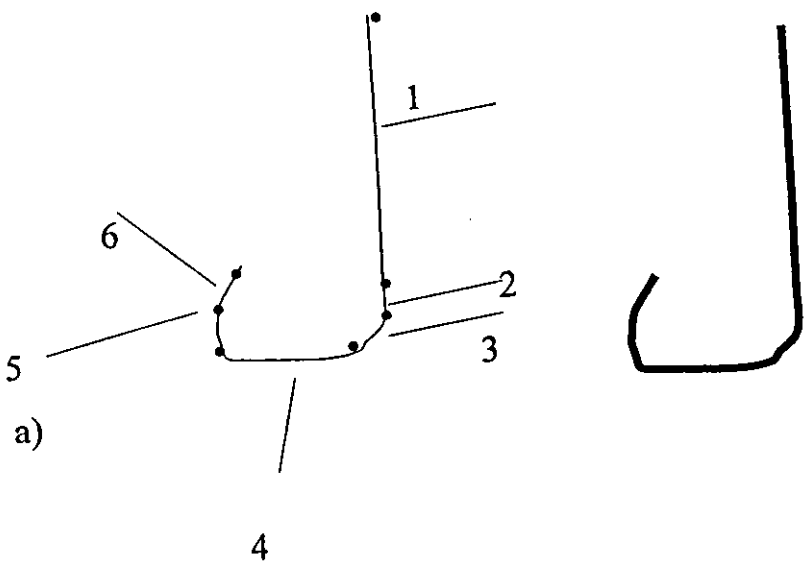
The overall success rate for the segmentation process could be computed by multiplying the success rate for every network by the vector reiterations, which gives a weighted success average of about 89%

4.6 Segmentation Neural Network Evaluation

The NN1 is considered the success key for the neural network system and to the partition process. There are two reasons for that. The first one is the large appearance of the vector V1 in the Arabic written skeleton. The V1 almost has elements as much as the other vectors. Therefore, the success of NN is the success of the segmentation process in half the cases. The second reason is that this network is dealing with the vector that represents the case with the most variations in feature values. Therefore the network NN1 need a large training set in comparison with the rest of networks to reach the success rate that it achieved.

Following are different examples for the segmentation of the NN. The success of the Neural Networks in dealing with these different cases indicates the strength and flexibility that the neural networks has in general.

Figure 4.6 represents the Arabic letters "ل" where, all pairs in the Figure except pair (6) of type V1, which can be handled by NN1. Notice that segmentation is not admissible at any pair in the skeleton.



Pair #	Decision
1	0.08914
2	0.08961
3	0.09228
4	0.00416
5	0.00004
6	0.34153

b)

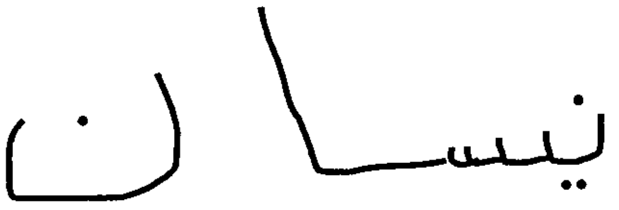
Figure 4.6: Recognition by the NN type . a) segmentation of the isolated letter "ل", b) segmentation decision.

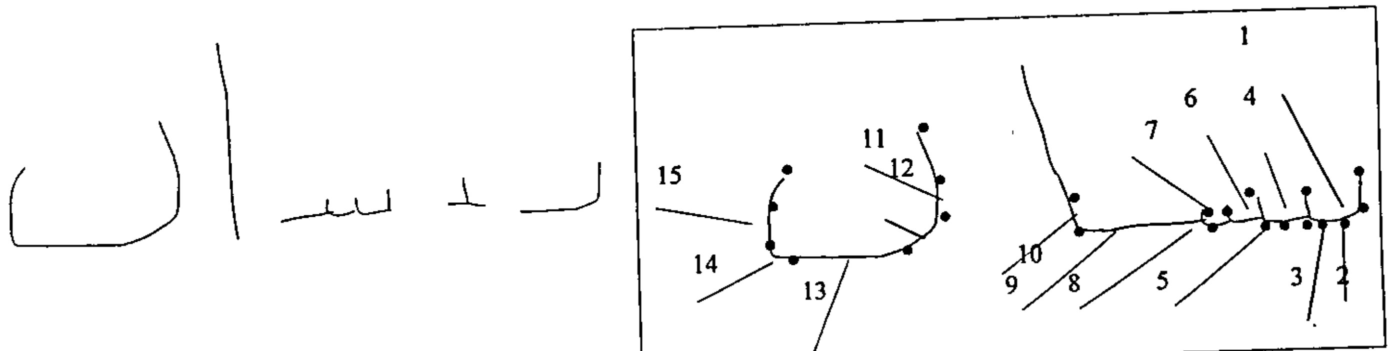
Notice the values and the pair number 6 of type V7 where NN7 dealt with and the decision was also right.

Figure 4.7 shows another Arabic word "نيسان" and contains five adjacent cusps. The letter "س" is the most difficult in representation as well as in segmentation. The difference in the cusps of "seen" "س" and the cusps that represent letters like "ـنـ", "ـيـ" is in the punctuation mark and at the angle between these cusps. The network NN2 has successfully produced a suitable segmentation decision at pair (3) and (5) . The NN1 gave a segmentation decision at pair (9), while in the second syllable "ن" the network gave non-segmentation decision at any pair . Notice the segmentation is relatively big and it approached the rejection area at pair (14); because of the punctuation mark for the letter noon "ن" .

Pairs	Network	Decision
1	NN1	0.16877
2	NN1	0.11883
3	NN2	0.99985
4	NN4	0.14183
5	NN2	0.99887
6	NN4	0.09310
7	NN1	0.10200
8	NN1	0.00014
9	NN1	0.99691
10	NN1	0.02371
11	NN1	0.09567
12	NN1	0.08945
13	NN1	0.25081
14	NN1	0.40159
15	NN1	0.02371

b)





a)

Figure 4.7: a) segmentation the words "نيسان", b)the decision rates of the 15 pairs resulted for segmenting the word "نيسان".

The succeed segmentation done by NN2 on pair number 3 and pair number 5 and NN1 on the pair number 9. The highest decision rates are come from NN2 for pair 3, NN2 for pair 5 and NN1 for pair 9.

4.7 The Result of Neural Networks in Classification Phase

The smoothing algorithm was applied on Arabic characters after the process of gathering, preprocessing and building NN steps. This algorithm was tested on Ottoman languages before (Atic A.,1994).

The RPROP was used in our classification. The MSE was set to 0.01 and the maximum number of epochs was 100,000. All the experiments reached the goal before reaching the maximum epochs.

To compare the results obtained from this research with results from different researchers in Arabic language or other languages which have the same letters as Arabic letters (Ottoman, Persian).See Table 4.2.

Table 4.2: Previous result to recognize Arabic (and Arabic like) characters.

Research	Languages	Tools	Data sets	Results (%)
(walker,1996)	Isolated handwritten Arabic characters without preprocessing	Multi-layer feedforward backpropagation	28 classes (characters)	73
			17 classes	81
≈	≈	≈	28 classes	67
≈	≈	Learning vector quantiser		
			17 classes	74
≈	≈	≈	28 classes	50
≈	≈	Kohen feature map-network		
			17 classes	54
≈	≈	≈		90.7
(Atic,1994)	Ottoman	Hidden Markov Model	-	
			10 classes	97.2
(Hossain.,1997)	Numeral	MLFF NN.	32 classes	80.1
≈	Persian Handwritten	Elastic matching		
(El-Melgy ,1996)	Handwritten Arabic character	Template matching	28 classes	91.3
	Numeral	MLFF NN.	10 classes	95.7
≈			17 classes	92
(Amin,1986)	Handwritten character without punctuation	MLFF NN.		
(El-DosKey ,1992)	Handwritten Arabic characters	Chain-code & binary tree	28 classes	90
(Al-Yosefi and Upda ,1992)	Printed Arabic characters without punctuation marks	Statistical features	17 classes	98.79
(El-Khaly and Sid-Ahmad ,1990)	Machine printed Arabic characters	Hidden Markov model	28 classes	95
(Mahmoud et al ,1991)	Machine printed Arabic characters	Fourier description and character contour	28 classes	98

In this research the recognition were carried out on the image with and without preprocessing. Table 4.3 presents the experimental results without preprocessing.

Table 4.3: The accuracy rate of Training and Testing set without pre-processing for 8 different experiments..

Experiment number	Number of sample	Accuracy rate of Training sets	Accuracy rate of Testing sets
Exp.1	50	85.72	71.90
Exp.2	50	84.2	69.21
Exp.3	100	92.63	77.07
Exp.4	100	93.25	77.21
Exp.5	100	93.13	76.11
Exp.6	100	92.84	75.31
Exp.7	100	90.9	75.31
Exp.8	200	93.90	78.05
Average		91.03	75.96

The results show no encouragement, when the algorithm was applied in a raw data. The accuracy rate increase as the number of sample is increased. The maximum value of accuracy in training set is 93.90, which corresponds to maximum number of sample. The maximum value of accuracy in testing test is 78.05, also correspond to the maximum number of sample 200. The weighted average of accuracy in training set is 91.03, while the weighted average accuracy in the testing sets is 75.96.

Figure 4.8 shows representation of training and testing accuracy with out pre-processing.

Table 4.4 shows the relation between the number of samples in each experiment and the maximum number of epochs.

Table 4.4: The maximum number of epochs according the experiments.

Experiment number	Number of sample	Maximum number of epochs
1	50	37,200
2	50	46,076
3	100	62,701
4	100	60,200
5	100	59,822
6	100	68,956
7	100	66,028
8	200	89,682

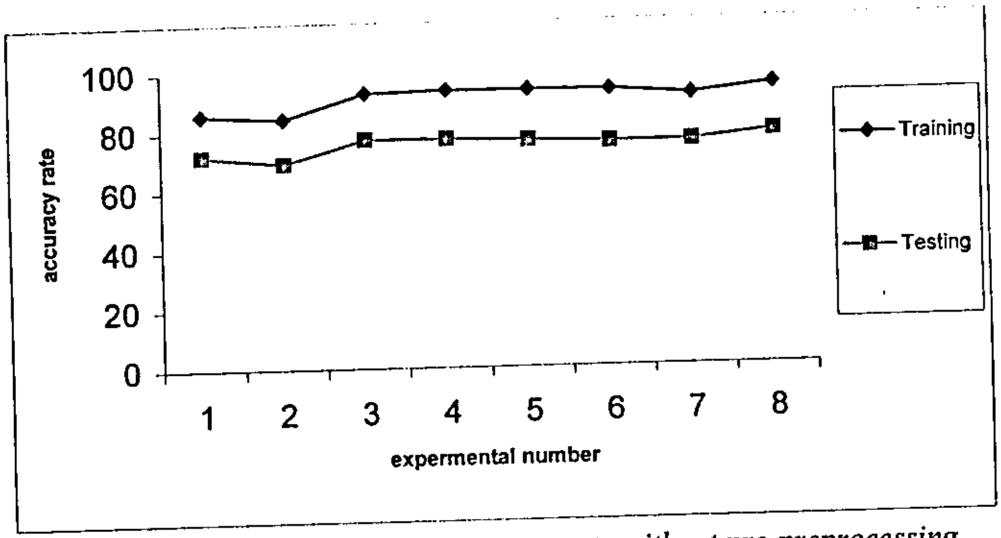


Figure 4.8: Training and Testing accuracy rate without pre-preprocessing.

Table 4.5: The accuracy rate of Training and Testing sets with pre-processing.

Experiment number	Number of sample	Accuracy rate of Training sets	Accuracy rate of Testing sets
Exp.1	50	89.2	75.05
Exp.2	50	88.8	74.85
Exp.3	100	95.3	85.1
Exp.4	100	97.10	84.51
Exp.5	100	97.18	85.37
Exp.6	100	96.91	83.87
Exp.7	100	95.30	84.00
Exp.8	200	97.90	86.82
Average		95.82	83.93

The results on the samples after pre-processing are done. Table 4.5 shows the result of applying the tests on the same samples in Table 4.4, but after pre-processing. All the accuracy rates of Training and Testing sets are increased in all the experimental. Figure 4.9 displays the training and testing results.

The table shows that, the weighed average of accuracy in training is increased from 91.03 to 95.82 after applying the pre-processing process and the average of accuracy rate in testing increased from 75.96 to 83.93. These are not encouragement results, considering the case similarity in shapes of letters as shown in Table 4.2. The properties of Arabic characters decrease the rate of accuracy.

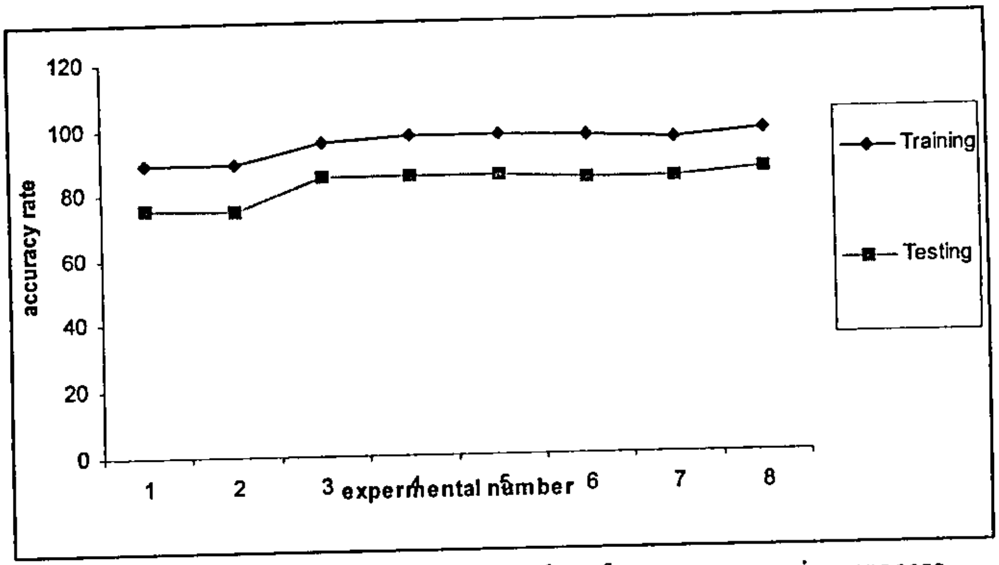


Figure 4.9: The Training and Testing results after pre-processing process.

Figure 4.10 compare the accuracy rate of Training set before and after pre-processing.

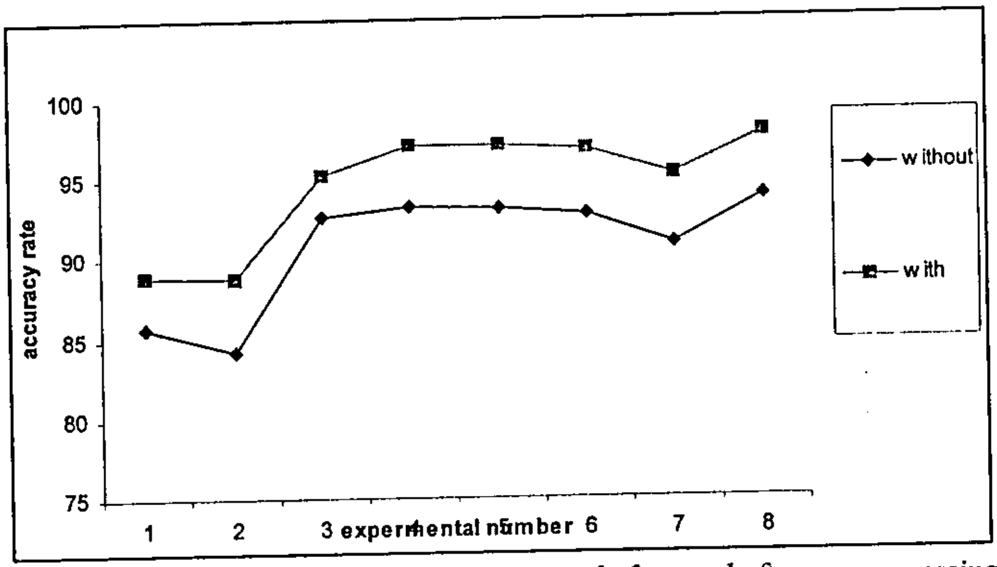


Figure 4.10: The accuracy rate of training set before and after pre-processing.

Figure 4.11 illustrates the rate accuracy rate of testing sets before and after pre-processing.

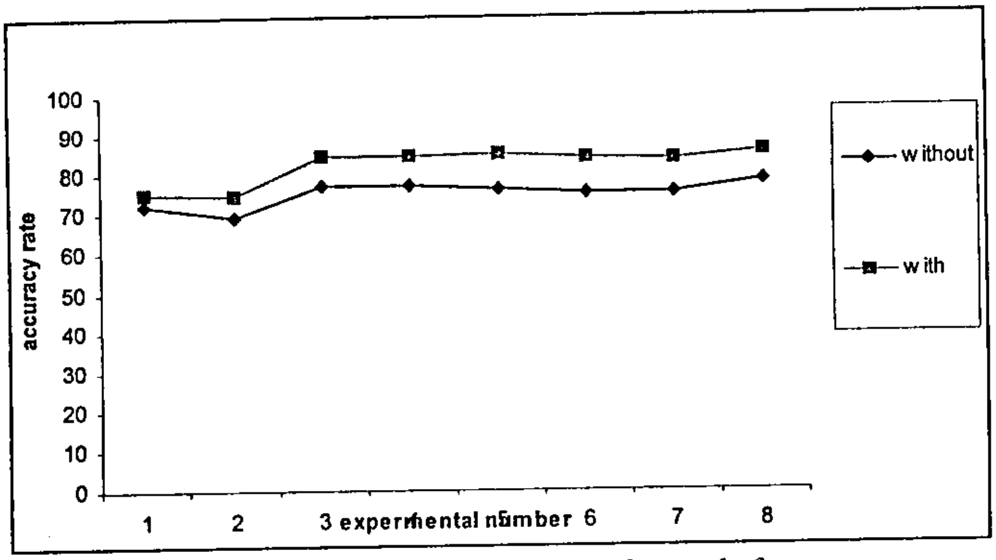


Figure 4.11: The accuracy rate of testing sets before and after pre-processing.

The results indicate the hardest to recognize the letter sheen "ش" then the letter Ta "ت" .The easiest to recognize is the character Alef "أ" . We made an experimental samples after excluding all characters having the same shapes. The character sets were reduced from 28 to 15 characters. Table 4.6 presents the 15 characters and Table 4.7 shows the results of applying Neural Networks on the 15 characters.

Table 4.6 : The Arabic characters after excluding the characters having the same shapes.

Name	Name in Arabic
Alef	ا
Ba	ب
Jeem	ج
Dal	د
Ra	ر
Seen	س
Sad	ص
Tta	ط
Ain	ع
Fa	ف
Lam	ل
Meem	م
Waw	و
Ha	ه
Ya	ي

Table 4.7 : The accuracy rate of 15 character with pre-processing process.

Experiment number	Number of samples	Accuracy of Training sets	Accuracy of Testing sets
1	100	98.18	89.58

When comparing these results with the results from previous steadies done by other researchers, you will find that these results are not the best, but also not the worst.

Our results indicate better accuracy rate in recognizing the 28 isolated handwritten Arabic character than those done by Walker 1996 (73%), when Neural Networks and Kohen map-network (50%) were applied.

The results of (El-Melgy, 1996) , using Template matching and (El-Desokey,1992) using Chain-code & binary tree are better than our results using RPROP algorithm.

Chapter 5

Summary and Conclusion

A system to manage the handwritten Arabic text has been implemented. The research initially aimed to recognize the Arabic characters in Naskh fonts. However, the built system can deal with fonts (i.e Kufi, Diwani…). This research focuses on skeleton representation, segmentation , and classification using neural networks. A Fuzzy ISODATA algorithm was applied to find the hand written Arabic word skeleton. This algorithm is distinguished by a low sensitivity towards flaws and also by its ability to find a unique skeleton in spite of the slope grade the word has entered by. The resulting skeleton comprises of a set of skeleton vertices and an adjacent matrix, which permit using simple data skeleton for representations with simple storage requirements.

Adding a thinning step on the entered word before applying the fuzzy ISODATA algorithm to improves the algorithm performance in two issues: the first one beat the slowness when applying the algorithm where the processing time was 25% of the original time when adding the thinning step. The second one was expressing the words medial axis.

It is possible to overcome the initial section of the cluster number through a direct relation between the primary cluster number and the size of the word. The proposed segmentation system needs a set of features that has been extracted from the syllable skeleton. A constrained Depth-First search algorithm to find the loops in the skeleton was developed and is one of the important features. This algorithm has recorded a success rate up to 99% and a suitable format to adjust the feature values and make it less sensitive to the change in writing size.

This research introduced a new method to segment the words and the hand written syllables based on using a classified neural network system. The used network has small size and limited needs. The neural networks recorded a success rate of about 89% for the tested sample.

The success of the neural network was due to many reasons. The most important one is using entry vectors with a set of suitable features instead of entering the pixels of image. Using a special NN for every type of pair in the segmentation set, instead of using one. N.N. to deal with all the pairs made the works of the network more specified. This made it easier to learn, faster and easier.

We use NN in classification of the isolated Arabic characters. Applying smoothing algorithm and pre-processing increasing the accuracy from 75.91% to about 83.93%.

A lot of gains can be achieved from using the neural network in segmentation. These make it ahead of the traditional methods. An efficient neural network could be developed faster and easier comparing with other methods. The results from classification stage are not encouraging comparing with the other results using another methodologies such as template matching and chain-code & binary tree. It is suggested to use neural network for segmentation and use another technique for classification. If you find the skeleton and feature extraction of isolated handwritten Arabic characters, you might obtained better results. It is advised to find a method to transfer the character size (obtained from segmentation stages of HACRS) to a fixed size. In overall, using neural networks for handwritten Arabic characters is advisable to be used in some stages of the process and not in all recognition stages.

References

Abdlelazim, H.1985. Text recognition, theory and implementation, Ph.D. thesis Cairo University.

Abdlelazim, H. and M. hashish.1989. Automatic reading of bilingual typewritten text, proc. Computero'89 VLSI & computer peripherals, Vol.2, pp.140-144.

Abdlelazim, H. and M. hashish .1990. Arabic typeset : OCR approach, single processing V: Theories and Applications, Elsevier Science publishers, pp1019-1022.

Abdlehamid, T.1995.Neural network pattern recognizer : recognition of printed Arabic text, Cairo University.

Abdulmaged, A.1994. A Novel approach for a trainable Arabic/Latin text reader ,Cairo University.

Abo-Samara ,G. 1996.on-line Arabic cursive script recognition, Ph.D. thesis, and faculty of Engineering, Cairo University.

Abuhaiba ,I., S. Mahmoud and R. Green .1994. Recognition of handwritten Arabic characters," IEEE Trans. PAMI, Vol.16,No. 4,324-335.

Al-Badr B. and Haralick .1998. Symbol recognition without prior segmentation, University of Washington.

Alimi, A. and O. Ghorbel. 1994. Comparative study of two algorithms for the recognition of on-line Arabic Handwritten characters, proc. ICECS94,Criro,Egypt,pp.973-977.

Allam , M. 1995. A hidden Markov model based approach to Arabic optical character recognition, Cairo University.

Almuallim, H. and S. Yamguchi .1987. A method of recognition of Arabic cursive handwriting, IEEE trans. PAAMI, Vol. PAMI-No,5,pp.715-722.

Al-Halag, A.1997. A Microcomputer Based Tool for Arabic Fonts Recognition, Abhath Al-Yarmouk, Yarmouk University.

Al-Sadon, H. and A. Amin.1995. A new structural technique for recognizing printed Arabic text. International Journal of pattern Recognition and Artificial intelligence,91:101-126.

Al-Taani, A. 1994. Toward The Automated Assessment of Pediatric bone Age using Computer Vision, Ph. D. Thesis, University of Dundee.

Al-Yousefi, H. and S. Upda .1992. Recognition of Arabic characters, IEEE Trans. PAMI, Vol.14. No.8,pp.853-857.

Amin, A. 1982. Machine recognition of handwritten Arabic words by IRAC II, Proc. 6th Int. conf. Patt. Recon., Munich, Germany.

Amin, A. 1986. Machine recognition of multi-font printed Arabic texts,Proc.ICPR,IEEE,pp.392-395.

Amin, A., Haton and R. Mohr ,1980. Handwritten Arabic character recognition by IRAC system, Proc. 5th Int. Conf. Recogn., Miami,FL,ppp.721-731.

Amin, A. and G. Masini. 1982. Machine recognition of Arabic Cursive words, SPIE Int. Soc. Opt. Eng., Vol.359,pp.286-292.

Amin, A. and Masini .1986. Machine recognition of multi-font printed Arabic texts, Proc. ICPR, IEEE, pp. 392-395.

Atic, A.1994. Segmentation, feature extraction and recognition of ottoman script, a master thesis in E. E. Eng. Middle East technical university ,Ankara Turk.

Bishop, C. 1995. Neural networks for pattern recognition, Clarendon Press, Oxford, UK.

Cao, J., M. Shidhar , F. Kimura and M. Ahmadi .1992. Statical and Neural classification of handwritten numerals: A comparative study. In proceedings of the international conference on document analysis and recognition, pages 643-646.

Casey, R. and E. Lecolinet. 1995. Strategies in character segmentation :Survey. In proceedings of international conference on document analysis and recognition, pages 1028.1033.

Cho, S. and Kim. 1995. Combining Multible Neural Networks by fuzzy Integeral for rubust Classification, IEEE Trans. Syst. Man, Cyber , pp.380-384

Damrah, A. 1985.“الخط العربي”جذوره وتطوره

Darpa.1988. Neural Network Study , AFCEA International Press, p. 60.

Davies and Plummer .1981. Thinning algorithm: A critique and a new methodology, pattern recognition, vol. 14, nos. 1-6,pp. 53-63.

El-Doskey. 1992. A handwritten Arabic Character Recognition Technique for machine Reader. International Journal of Mini and Microcomputer vol.14 no.2.

El-Gowely.1989. Recognition of multi-font photo script Arabic text, MS Thesis, Faculty of Engineering, Cairo University.

El-Melegy. 1996. Handwritten Arabic Character recognition, Assuit University.

Eric.1997. Cursive word Recognition: methods and strategies, E.N. telcom Paris, 46 rue Barrault, 75013 Paris, France.

El-Khaly, F. and Sid-Ahmad. 1990. Machine recognition of optically captured machine printed Arabic text, pattern recognition, and pp.1207-1214.

Freeman, J. and Skapura. 1992. Neural networks, algorithms, applications and programming techniques, Addison-Weslsley publishing Co.

Fujisaki, T., T. Chefalas, J. Kim and Wolf .1991. On-line run on character recognizer Design and performance," charter and Handwriting Recognition: Expanding Frontiers, World Scientific Series in Computer Science,Vol.30,pp. 123-137.

Gonzales, R. and P. Wintz .1987. Digital image processing, Addison-Wesley publishing Co. 2nd edition.

Guyon, I. 1991. Application of Neural Networks to character recognition. International journal of pattern recognition and artificial intelligence,5:353-382.

Guindi, R.1987. an Arabic text recognition system,M.S. thesis , Cairo university.

Habib. 1997.Analysis and Recognition of Persian and Arabic Handwritten Characters, university of Adelaide.

Hart, P. 1986. The condensed nearest neighbor rule," IEEE Trans. Information Theory, IT-14,pp. 515-516.

Haykin, S.1994. Neural Networks: A Comprehensive Foundation, NY: Macmillan, p. 2.

He, Y., M. Chen , and A. Kundu .1994. Off-line handwritten word recognition using HMM with adaptive length Viterbi algorithm. In proceedings of the international conference on pattern recognition, pages 460-462.

Hossain. 1997. Analysis and Recognition of Parisian and Arabic Handwritten Characters. University of Adelaide.

Huang, Y., K. Liu and C. Seun. 1995. The combination of Multiple Classifiers by neural network Approach, Int. j. Pattern Recogn. Artif. Intell, pp. 579-597.

Jain, A.1989. Fundamentals of digital image processing, prentice-hall Inc, 1989.

Jain, R., Kasturi and Shunck .1995. Machine Vision, McGraw-Hill Inc.

Khellah and Mahmoud. 1994. Recognition of Hexagonally Sampled Printed Arabic Characters, Arabian J. for science and engineering.

Mahmoud, S., I. Abuhiba And R. Green . 1991. Skeletonization of Arabic Characters using clustering based Skeletonization Algorithm CBSA, pattern recognition, pp. 453-464.

Majid, A. and Bayoumi.1994. Recognition of Arabic characters using Neural Networks, Proc. ICECS'94, Cairo, Egypt,pp.720-725.

Matsui, T., Noumi , Yamahhita , Wakahara and Yoshimuro . 1996. State of the art of the handwritten numeral recognition in Japan – the results of the first inputs character recognition competition. In proceedings of the international conference on document Analysis and Recognition, pages 391-396.

Michael, D. 1986. Automatic processing of digitized hand radiographs: an application of computer vision, Ms.c Thesis, Cambridge.

Mori, S., Suen and Yamamoto.1992. Historical review of OCR research and development. Proceedings of the IEEE ,80:1029-1058.

Nigrin, A.1993. Neural Networks for Pattern Recognition, Cambridge, MA: The MIT Press, p. 11.

Nouh, A., Sultan , and Tolba.1980. An approach for Arabic character recognition, J. Eng. Sci, Univ. Riyadh, Vol.6, No.2,pp. 185-191.

Nouh, A., and A. Sharaf Eldin . 1988. Boolean recognition technique for typewritten Arabic character set, Egyptian computer Society, pp.21-28.

Pao, Y.1989. Adaptive pattern recognition and neural networks, Addison-Wesley.

Patterson. 1996. Artifical Neural Networks Theory and Applications. Prentice Hall.

Ramisis, R., S. El-Dabi and A. Kamel . 1988. Arabic character recognition system, IBM Kuwait Scientific Center, Rep. KSC027.

Safabaksh, M. and Shaygan. 1995. A method for evaluation of readability of handwritten Persian nastaligh texts and data base design. In proceedings of the first annual CSI computer conference CSICC95,pages 88.95.

Saleh, A. 1994. A method of coding handwritten Arabic characters and its application to context-free grammar, pattern Recognition Letters, Vol.15,pp. 1265-1271.

Sharkawy ,M., Tolba, and Shaddad .1988. Fourier descriptors for printed Arabic character recognition, Egyptian computer Society, pp. 43-52.

Shunji, Ching and Kazuhiko Yammato.1992. Historical Review of OCR Research and Development, proceeding of the IEEE, Vol.80, No.7, pp. 1029-10558.

Stefanelli and Rosenfeld .1971. Some parallel thinning algorithms for digital pictures, "J. Ass. Comput. Mach., Vol. 18,No 2, PP.255-264.

Suen, C. 1982. Distinctive features in automatic recognition of handprinted characters. Signal processing,4:193-207.

Suen, C. and L. Lam . 1993. Building a new generation of handwritten recognition system. Pattern recognition letters, 144:303-316.

Suen, C., R. Shinghal and Kwan. 1997. Dispersion factor: A quantitative measurement of the quality of handprinted characters, Proc. Int. Conf. On Cybernetics and society, pp. 681-685.

Tappert, C., C. Suen and Wakahara. 1990. The state of the art in on-line handwriting recognition, IEEE trans. on pattern recognition and machine intelligent, vol. 12,No.8,pp. 787-808.

Trier, O. and T. Taxt. 1996. Feature extraction methods for character Recognition-survey, pattern recognition, 294, pp. 641-662.

Walker. 1996. Coparative Results for Arabic Character recognition using Artificial neural Networks. Brisbane Q 4001 Australia.

Yamany, S.1995. A complete analysis of an Arabic text Reader, Cairo University.

Yong .1997. Pattern recognition www.ph.tn.tudelft.nl/prin./prarea.html.

Zhang, T. and C. Suen. 1984. A Fast Parallel Algorithm for Thinning Digital Pictures, Commun. ACM27, pp.236-239.

Zurada, J. 1992. Introduction To Artificial Neural Systems, Boston: PWS Publishing Company.

Westall, J. and Narasimha. 1997. An evolutionary approach to the use of Neural Networks in the segmentation of handwritten numerals, Int. J. pattern recogn. Artif. Intell, pp.717-735.

الملخص العربي

التعرف على أحرف اللغة العربية المكتوبة يدوياً باستخدام الخلايا العصبية

إن التعرف على الحروف العربية يعد من الموضوعات التي تلاقي اهتماماً كبيراً نظراً لتزايد الاعتماد على الحاسب الآلي في معالجة البيانات. إن تقسيم الكلمات والمقاطع العربية إلى الأحرف التي تتكون منها عملية بالغة الصعوبة خاصة عندما تكون الكتابة يدويه وذلك لعدم وجود قواعد واضحة يمكن استعمالها . كما أن عدم وجود حجم أو شكل واحد للحرف بالأضافه للتداخل ما بين الأحرف يزيد من تعقيد العملية.وترجع أهمية هذا البحث إلى قلة البحوث التي تعرضت لهذا الموضوع , وصعوبة معالجة السمات الخاصة بالخط العربي التي تختلف عن باقي اللغات , والى التنوع الكبير في خطوط الأفراد.

مجال البحث في هذه الرسالة هو التعرف على الحروف العربية المكتوبة بخط اليد. في هذا البحث استخدمت خوارزمية تجميع عنقودي غائمة لإيجاد هياكل الكلمات والمقاطع العربية المكتوبة يدوياً.واستعملت هذه الخوارزمية في بحث سابق, وقد سجلت نجاحاً مميزاً في قدرتها على التعبير عن العلاقات التركيبية للحرف العربي, إلا أن البحث أشار إلى مشكلتين تعاني منهما الخوازميه وهما البط الشديد مقارنة مع طرق التمثيل الأخرى , ومشكلة الاختيار الابتدائي لعدد العناقيد. تم في هذا البحث اقتراح حل للمشكله الأولى بإضافة خطوة ترقيق للكلمة قبل تطبيق الخوارزمية. حيث استغرقت الحوازميه مع الترقيق حوالي 25% من زمن التنفيذ الأصلي,كذلك أدى الترقيق آلي تحسن في أداء الخوارزمية حيث يمتاز الهيكل الناتج بانه يعبر عن العناقيد التي تشكل المحور الوسطي للكلمة. وللتغلب على المشكلة الثانية, تم اقتراح اختيار عدد ابتدائي من العناقيد يتناسب طرداً مع حجم الكلمة أو المقطع , يتم بعد ذلك التخلص من العناقيد الزائدة والتي ليس لوجودها أية أهميه في التركيب الهيكلي للكلمة.

. ومع إن استعمال الشبكات العصبية لاقى نجاحاً واسعاً في تميز الأنماط وذلك لإجراء عملية التصنيف إلا أن استعمالها من اجل تقسيم الكلمات المتصلة أو الأرقام والأحرف ما زال قليلا في اللغة العربية.وتم استعمال الشبكات العصبية أيضا في عملية تميز الأحرف العربية المكتوبة بخط اليد.

ونتيجة تطبيق خوارزميات الشبكات العصبية لتميز النص المكتوب يدويا باللغة العربية يدويا كانت نسبة الدقة في مرحلة التقسيم تصل حوالي 89% . أما في مرحلة التميز فقد تراوحت

نسبة الدقة بين 89 إلى 97% عند تطبيق الشبكة على بيانات التدريب و 75 إلى 83% عند تطبيقها على بيانات الفحص. وتبين إن نتائج هذا البحث افضل من غيرها من الدراسات السابقة عند استخدام الشبكات العصبية لتميز 28 حرفاً مكتوبة بخط اليد. إلا أن النتائج كانت أقل كفاءة من نتائج الدراسات التي استخدمت طرق أخرى . ويمكن تعميم الخوارزميات المقترحة والشبكات العصبية لاستخدامها في لغات أخرى