Analysis and Recognition of Persian and Arabic Handwritten Characters

by

Habib Mir Mohamad Hosseini

B.Sc. in Electronic Engineering, Isfahan University of Technology (1988)
M.Sc. in Digital Electronic, Sharif University of Technology (1991)

Submitted to the Department of Electrical and Electronic Engineering
in fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

UNIVERSITY OF ADELAIDE

November 1997

Signature of Author
Department of Electrical and Electronic Engineering
30 November 1997

Certified by ....
Dr. Abdesselam Bouzerdoum
Associate Professor
Thesis Supervisor

Analysis and Recognition of Persian and Arabic Handwritten Characters

by

Habib Mir Mohamad Hosseini

Submitted to the Department of Electrical and Electronic Engineering
on 30 November 1997, in fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Though research for designing a machine which can read characters and numerals started more than 90 years ago, problems of recognition of handwritten texts are yet to be completely solved. Even for languages like English or Chinese, for which extensive research has been done, it is probably safe to say that no single scheme is likely to satisfy the requirements in real industrial applications. One of the main reasons is the great variability in handwriting.

The primary goal of this dissertation is to study potential problems of off-line recognition of Persian and Arabic handwritten texts. Specific characteristics of these languages do not allow a direct application of algorithms proposed for the recognition of other character sets. Our study is based on a carefully collected data set containing unconstrained handwritten samples of isolated characters, words, and text from 54 Persian and Arabic speaking writers. Sometimes printed characters and text were used either to analyze the handwriting or to show the difference in recognition of printed and handwritten patterns.

The thesis is divided into three parts. The first part is devoted to analyzing Persian and Arabic handwriting styles. It starts with an introduction to Persian and Arabic writing styles. Then, two of the main problems of a Persian and Arabic handwritten character recognition, namely *similarity* and *variability* of patterns, are addressed. To this end, a geometrical model for distortion analysis of handwritten patterns is introduced, and is then used to investigate the variation of the character patterns. In this model, each distortion source is represented by a transformation matrix operation. Both theoretical and experimental results show that various sources of distortion have different effects on individual characters. Distortion parameters are then estimated for collected handwritten samples of Persian and Arabic characters. This first part is concluded by a comprehensive review on the subject of recognition of printed and handwritten Persian and Arabic texts.

In the second part, we evaluate and test different approaches to feature extraction and classifier design. We also propose some algorithms for feature extraction; in our first approach, we introduced a complex logarithmic transformation technique for invariant feature extraction. This technique is similar to the way the receptors are

distributed in the human retina. This method of feature extraction is then applied to the recognition of both printed and handwritten isolated characters. This feature extraction technique is translation, scale and rotation invariant. For a set of printed Persian and Arabic isolated characters of different scales and rotation ranges, a high recognition rate of 97% was achieved, however, for handwritten characters the system showed a poor performance. The best recognition rates were obtained by using shadow features and a probabilistic classifier, 83% without rejection and 88% with an 11% rejection rate of ambiguous characters.

A new feature extraction technique was developed for recognition of unconstrained handwritten Persian and Arabic numerals. The best recognition rate achieved for a single classifier system was 80%, while using a combined system increased the recognition rate up to 91%. The study of the confusion matrices of the recognition systems revealed that most of the misclassifications were caused by similar digits. The recognition rate was increased up to 94% by rejecting 7% of the patterns.

The elastic matching is among other approaches that have been used to overcome the problem of pattern variation. In a second approach, we used elastic matching technique as a distance measure between the patterns of handwritten digits. Experimental results showed that even these techniques are not capable of completely resolving the problems of ambiguity caused by similar characters and variability of handwriting styles. Some characters become very similar when they are distorted, and hence even elastic matching technique fails to distinguish between these characters. To further improve the performance, context information should be included.

An experiment is done on human recognition of samples of isolated handwritten characters. The best reliability result for the human expert on the collected samples was 0.86. The interesting result is that the best proposed recognition system made almost the same mistakes as human experts; they showed a poor performance in distinguishing between similar patterns. This means that even a human expert is not able to resolve these problems without using context. This led us to the idea of using multiple experts or combination of multiple classifiers techniques to improve the recognition rate of handwritten samples.

In the third part, methodologies for classifier combination are studied. We evaluated three different systems for combining multiple classifiers: weighted voting, linear committee combiner, and a multi-label combiner. In all cases the experimental results showed that the combined system always outperforms all of the individual classifiers. By rejecting ambiguous patterns, both the recognition rate and the reliability improved. Using *a prior* information on the performance of individual classifiers for each class label increased the total recognition rate. The best recognition results achieved by the weighted voting combiner, linear committee combiner, and multi-label combiner were 94%, 96%, and 94% with rejection rates of 28%, 21%, and 24%, respectively.

Thesis Supervisor: Dr. Abdesselam Bouzerdoum
Title: Associate Professor

This work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by any other person, except where due reference has been made in the text.
I give consent to this copy of my thesis, when deposited in the university library, being available for loan and photocopying.

Habib Mir Mohamad Hosseini

November 1997

To my wife Zohreh

and

my lovely son, Amin

Acknowledgements

I would like to express my sincere gratitude to my thesis supervisor Assoc. Prof. Abdesselam Bouzerdoum (Salim). He introduced me the challenging area of pattern recognition. It was his guidance and support that has made this dissertation possible. I am very fortunate to have had the opportunity to work with him.

I also thank the ministry of culture and higher education (MCHE) of the Islamic Republic of Iran for providing the scholarship. Special thanks, however, goes to the very nice people of Iran whom I am indebted.

I enjoyed discussions with the pattern recognition group of the cooperative research centre for sensor signal and information processing (CSSIP). Special thanks to Prof. Robert E. Bogner, head of the CSSIP pattern recognition group for his comments and suggestions. I also thank postgraduate students of the group for listening to many of my presentations and offering useful suggestions. I am also grateful to my friend Mr. Carmine Pontecorvo. He spent long hours improving my writing abilities by a thorough reading of my thesis.

An acknowledgement is devoted to the CSSIP. Interaction with the research staff at CSSIP made my studentship a unique experience.

Many thanks goes to my friends, Iranian students in Adelaide, for their support. They have done a great job by helping me to collect the handwritten samples. Special thanks to my dear friend Dr. Alireza Moini for his invaluable helps in printing the thesis. I also thank Dr. Nasrollah Afjei for providing me samples of Persian calligraphy from Iran.

To my wife, Zohre, your understanding and encouragement helped me throughout this dissertation. You and my lovely son Amin fill my life with joy and happiness. I am really grateful to you.

To my parents, for their lifelong support and inspiration which always strengthened me through many tough times, God bless you.

Contents

List of Figures

List of Tables

List of Publications

1. H. M. Hosseini and A. Bouzerdoum, "A System for Arabic Character Recognition", *Proc. of second Australian and NewZeland Conf. On Intelligent Information Systems (ANZIIS'94)*, pp. 120-124, Brisbane, Australia, 29 Nov.- 2 Dec. 1994.

2. H. M. Hosseini and A. Bouzerdoum, "Arabic Character Recognition with Neural Networks", *Proc. of the sixth Australian conf. on Neural Networks (ACNN'95)*, pp. 261-264, Sydney, Australia, Feb. 1995.

3. H. M. Hosseini and A. Bouzerdoum, Modified Ring-Projection Transformation Method for Arabic and Persian Character Recognition, *Proc. of Computer Society of Iran Computer Conference (CSICC '95)*, , pp. 119-124, Teheran, Dec. 1995

4. H. M. Hosseini and A. Bouzerdoum,"A Scale and Rotation invariant method for Arabic and Persian character Recognition", *Proc. of the International conference of VISUAL'96*, pp. 328-334 ,Melbourne, Australia, Feb. 1996.

5. H. M. Hosseini and A. Bouzerdoum, "A Combined Method for Persian & Arabic Handwritten Digit Recognition", *Proc. Of ANZIIS'96 Conference*, pp. 80-83, Adelaide, South Australia, 18-20 Nov.1996.

6. H. M. Hosseini and A. Bouzerdoum, "Scale Invariance of a Combined Method for Recognition of Handwritten Persian and Arabic Digits", *Proc. of the International Symposium on Multi-Technical Information Processing (ISMIP'96)*, pp. 101-106 ,Hsin-Chu, Taiwan, Dec.1996.

Chapter 1

Introduction

As more of the world's information processing is done electronically, it becomes more important to make the transfer of information between people and machines simple and reliable. Thus, computers should be able to interact better with people and to act in a less constrained manner than has previously been possible. Handwriting is a natural means of communication which nearly everyone learns at an early age. It provides an easy way of interacting with a computer, requiring no special training other than that acquired in early education. In addition to a potential mode of direct communication with computers, another principal motivation for the development of *optical character recognition* (OCR) systems is the need to cope with the enormous flood of paper such as bank cheques, commercial forms, government records, credit card imprints and mail sorting generated by the expanding technological society. Many OCR systems have been developed for different applications including postal address reader device for handwritten and printed postal codes [110], telecommunication system as aid for the deaf [71], bank cheque reader and verifier [52, 29], and reading device for the blind [22].

Even though research for designing a machine which can read characters and numerals started more than 90 years ago [80], the problem of recognition of handwritten characters is yet to be completely solved. Even for languages like English or Chinese, on which extensive research has been done, it is probably safe to say that no single scheme is likely to satisfy the requirements in real industrial applications. The reason why the success of OCR has not carried over into handwritten recognition is the great variability in handwriting styles.

1.1 Character Recognition Systems

The conventional off-line character recognition process, shown in Fig. 1-1, may be modeled by the following sequence of operations: data acquisition, preprocessing, feature extraction, and classification. In the data acquisition stage, the document is scanned, digitized, and the resulting image is stored in a binary or gray-scale format. Preprocessing is intended to modify the data so that the extracted features are more amenable to classification. The segmentation stage separates the text lines and then splits them into characters. For some systems, however, this stage may be omitted, as indicated by the dashed lines. The feature extraction stage detects features of the input data for the purpose of recognition. The most important aspects of this stage is the selection of a feature set which can adequately discriminate between the patterns to be recognized. The larger is the number of similar patterns, the larger the number of features needed to discriminate between them. The function of the classifier stage is to make decisions based on the features. After the classification stage, there may be another stage for post-processing which uses other sources of information, such as contextual information, to verify and to correct mistakes in the OCR output.

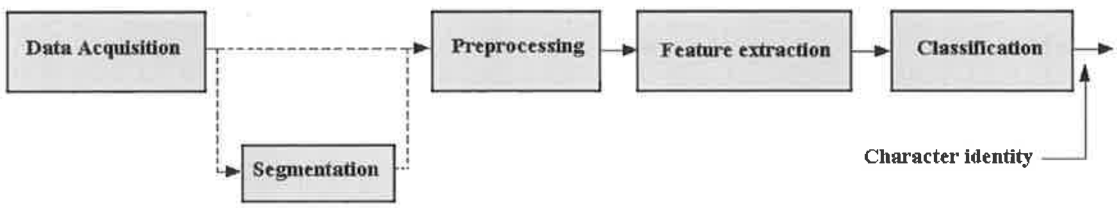


Figure 1-1: Block diagram of a typical OCR system.

The basic problem of character recognition is defined as assigning the digital image of a character into its symbolic class. The general term character recognition covers two categories: *on-line* and *off-line*. In on-line systems, the symbols are recognized as they are written, while in off-line systems recognition is performed after the writing or printing is completed. In terms of capability and complexity, the off-line schemes can be classified as [48]:

Fixed-font recognition systems which deal with the recognition of only a specific type of font.

Multi-Font recognition systems which can recognize more than one font.

Handwritten character recognition systems which deal with the recognition of unconnected handwritten characters.

Script recognition systems which recognize either connected or cursive unconstrained handwritten characters.

1.2 Persian and Arabic OCR

Research in the field of Arabic character recognition, as reported in [11], began in 1975 by Nazif [88]. This shows a very late start in the research as compared to earlier efforts in Latin dating back to the middle of 1940s [48]. The number of publications in Arabic and Persian character recognition indicates that there has been more research on Arabic character recognition than on Persian character recognition. However, almost all of the techniques used for Arabic character recognition are directly applicable to the Persian systems, although they may only need to be slightly modified to include four extra characters of Persian. As far as we are aware, the first publication for machine recognition of a printed Persian character set is a paper by Parhami and Taraghi [90].

The reasons that we have selected Persian and Arabic character sets for this research are:

- The amount of research done on the other languages is considerable when compared to the research on the Persian and Arabic character sets. Specially in the area of handwritten recognition, there have been very few serious works compared to the huge contributions in other languages such as Chinese and Latin.

- The specific characteristics of Persian and Arabic, i.e., the writing styles, similarity of characters, and cursiveness do not allow a direct application of algorithms proposed for the recognition of Latin and many other character sets.

- The cursive nature of Persian and Arabic handwriting texts and the unique characteristics of the character sets make the research in handwritten text recognition of these languages a challenging area for pattern recognition.

- Large variations in writing styles and existence of many similar characters make the recognition of unconstrained handwritten text a very difficult and still unsolved problem. Compared to English, there are more similar characters in Persian and Arabic. Moreover, due to the difference in shapes of a single character and mixture of fonts in handwriting, the range of handwriting variation is larger than for English handwriting.

- Arabic, Persian as well as other languages such as Urdu have a lot of similarities, hence the techniques for any of them is also of value for the recognition of the others as well. These very similar languages are spoken and written in a large area of two continents including countries in north Africa, middle east, central Asia, Pakistan, Afghanistan and parts of India. And if we add that Arabic is the official language of the Moslem community, the number of people using Arabic or Persian exceeds a billion.

1.3 Definition of the Problem

Despite a very late start for Persian and Arabic character recognition, many approaches in different countries have been tried to overcome special characteristics of Persian and Arabic writings which pose difficulties in the character recognition systems of these two languages. The lack of communication between the research groups, poor financial support, and the lack of standard data sets are big constraints for implementing commercial systems, as compared to the number of implementations of character recognition systems in other languages.

The primary goal of this dissertation is to study the potential problems of off-line recognition of Persian and Arabic handwritten texts. We believe that the two main obstacles for achieving higher recognition rates for Persian and Arabic handwritten recognition are the similarity between character patterns and the great variability in writing styles. In this dissertation, we will address these two problems by looking at their direct impacts on the character patterns as well as their effects on the recognition systems.

Comparing the number of publications in both printed and handwritten recognition of Persian and Arabic indicates that many researchers were interested in constrained handwritten or typed document recognition systems, we found there are very limited attempts to recognize unconstrained handwritten documents. The second goal of our studies is to test and explore different methods to overcome the two aforementioned problems. We will test different feature families and classification methods for both printed and handwritten characters. Some of these techniques have been already used by other researchers for Persian or Arabic character sets.

Many approaches for Persian and Arabic handwritten recognition reported high recognition rates; however, in almost all cases the high recognition rates reported were achieved on different data sets collected by the researchers themselves. Due to a lack of a standard data set, it is often impossible to compare the performances of different approaches for handwritten recognition systems. Our study is based on a carefully collected data set containing handwritten samples of isolated characters, words, and text from 54 Persian and Arabic speaking writers.

1.4 Thesis Outline

The first part of this dissertation is devoted to analyzing Persian and Arabic handwriting styles. It starts with an introduction to Persian and Arabic writing styles in chapter 2, followed by a comprehensive review of the related fields in chapter 3. Then, two of the main problems of Persian and Arabic handwritten character recognition, *similarity* of patterns of different characters and *variability* of handwriting styles, are addressed in chapter 4. To this end, a geometrical model for distortion analysis of handwritten patterns is introduced, and then used to investigate the variation of the character patterns. In this model, each distortion source is represented by a transformation matrix operation. Both theoretical and experimental results show that various sources of distortions have different effects on individual characters. Distortion parameters are then estimated for the collected handwritten samples of Persian and Arabic characters. These parameters can be used to build deformed prototypes or templates for individual characters or character sets.

In the second part of the thesis, different approaches for feature extraction and classification are studied. New feature extraction methods are proposed in chapter 5, including a complex logarithmic transformation technique for invariant feature extraction. A group of nine different feature families including the proposed feature extraction techniques are used for our studies. Different classification algorithms including MLP (Multi Layer Perceptron), PNN (Probabilistic Neural Network), KNN (K-Nearest Neighbor), and EMD (Euclidean Minimum Distance) are evaluated by using different features as inputs. The performances of different systems are compared for both printed and handwritten characters. Furthermore, the results of an experiment performed with human experts are presented in this chapter.

Chapter 6 is devoted to combination of multiple classifiers. In this chapter, we study the application of multiple classifier combination systems for Persian and Arabic handwritten isolated characters. Three methods of combining multiple classifiers are studied: weighted voting method, linear committee combiner, and our proposed algorithm, which is called *multi-label classifier combiner*. For each combined system we show by experiments that the combination of multiple classifiers always has a higher recognition rate than that of a single classifier.

In chapter 7 recognition and analysis of handwritten Persian and Arabic numerals are studied. We introduce a line segment model to represent all the characteristics of Persian and Arabic numerals. The extracted features are also used to demonstrate the similarity between digits and variation in writing styles. A new method of combination of multiple classifiers is also examined for increasing the overall recognition rate of the system. We use the elastic matching technique for recognition of handwritten Persian and Arabic numerals. The background theory followed by our proposed system for extracting the stroke sequences from off-line data are presented. This chapter is completed by presenting the recognition results of both the elastic matching and combination of multiple classifiers. Finally, this dissertation is concluded with a summary and discussion of future directions.

Chapter 2

Persian and Arabic Handwritings

2.1 Introduction

Several methods have already been developed for the recognition of Latin and many other character sets [82, 46, 84, 27]. However, intrinsic differences between the writing styles of Persian & Arabic and other languages do not allow a direct application of these algorithms to Persian and Arabic handwritings. The key to high recognition performance of handwritten characters of any language is the ability to detect and utilize the distinctive characteristics of the characters of the language. Persian and 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. In the first part, which is devoted to language description, we introduce the Persian and Arabic character sets, and some popular fonts. In the second part, we first present the primitive strokes from which all Persian and Arabic characters can be built, then in the following section we introduce the problems associated with handwritten recognition systems of Persian and Arabic texts, from primarily a pattern recognition point of view.

Table 2.1: Comparison of various languages

Characteristics	Persian	Arabic	English	Hebrew	Hindi
Number of characters	32	28	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 shapes of a character	1-4	1-4	2	1	1
Diacritics	Yes	Yes	No	No	Yes

2.2 Language Description

2.2.1 Character Sets of Persian and Arabic

Both the Persian and Arabic languages are similar to English in that they use letters, numerals, punctuation marks, as well as spaces and special symbols for mathematical expressions. However, they differ from English in their character sets and writing direction. In addition, the structure of the Persian and 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.

There are 28 characters in the Arabic character set; these characters and four extra ones make up the Persian character set (Table 2.2). This does not mean, however, that in both character sets there are only 28 or 32 unique shapes. The reason is that there are a 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 both languages, there exist different shapes for some characters, depending on their position in a word; some characters have up to four different shapes (see Table 2.2).

The writing styles of Persian and Arabic are almost the same, however they use different fonts. Texts are written from right to left and numerals are written from left to right. Both languages use 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 a

Table 2.2: Persian and Arabic alphabet with the different shapes of characters, depending upon the character position in a word. Characters marked with (*) are used only in Persian.

	Position of character in a word			
Name	Isolated	End	Middle	Beginning
ALEF	ا	ـا	-	-
BA	ب	ـب	ـبـ	بـ
PA (*)	پ	ـپ	ـپـ	پـ
TA	ت	ـت	ـتـ	تـ
THA	ث	ـث	ـثـ	ثـ
JEEM	ج	ـج	ـجـ	جـ
CHA (*)	چ	ـچ	ـچـ	چـ
HEH	ح	ـح	ـحـ	حـ
KHA	خ	ـخ	ـخـ	خـ
DAL	د	ـد	-	-
ZAL	ذ	ـذ	-	-
RA	ر	ـر	-	-
ZA	ز	ـز	-	-
ZHA (*)	ژ	ـژ	-	-
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
GHAIN	غ	ـغ	ـغـ	غـ
FA	ف	ـف	ـفـ	فـ
GHAF	ق	ـق	ـقـ	قـ
KAF	ک	ـک	ـکـ	کـ
GAF (*)	گ	ـگ	ـگـ	گـ
LAM	ل	ـل	ـلـ	لـ
MEEM	م	ـم	ـمـ	مـ
NOON	ن	ـن	ـنـ	نـ
WAW	و	ـو	-	-
HA	ه	ـه	ـهـ	هـ
YA	ی	ـی	ـیـ	یـ

word. However, some other 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 nine connected characters while the last one in (d) has four isolated characters.

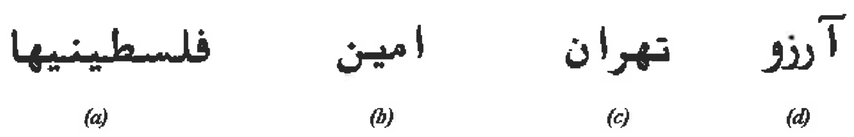


Figure 2-1: Persian and Arabic words with: (a) one subword (Palestinian), (b) two subwords (Amin), (c) three subwords (Tehran), and (d) four subwords (wish).

2.2.2 Printed and Handwritten Fonts

There are many different fonts used in Persian and Arabic handwritten 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 handwritings. Some fonts differ in the shapes of some of the characters, but other fonts differ also in the combination of connected characters. Sometimes those characters which are not connectable from the left are connected (Fig. 2-5 and Fig. 2-7). In addition, character sizes are different for different fonts, and the characters in some fonts include a hook-like curvature at the end or at the beginning. Table 2.3 shows ranking of some popular fonts. The ranking in this table is extracted from [40], and is accepted by calligraphy experts. Among these fonts, Naskh is the most popular font in machine printed documents. In the next few paragraphs, we briefly introduce some of these fonts.

Kufi

It is known as the first official font of writing in Islamic texts (Fig. 2-2). Thus this font was invented before the origination of Islam. The characters use more straight

Table 2.3: Rankings of Persian and Arabic fonts.

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

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*.

Figure 2-2: Arabic text written in Kufi font [40].

Naskh

This font is the most frequently used font in printed documents and the second most frequently used font in handwritten texts (Fig. 2-3). Invented in the seventh century by Ibn Moghalled Bayzavi by inspiration from the Kufi font, it is one of the simplest fonts in handwritings. There are two versions of this font for Arabic and Persian. The Arabic version of this font is called *Yaghooti* and the Persian one is called *Neyrizi*. 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.

Nastaligh

This font was invented by Mir Ali Tabrizi in the 14th century. It is the first most popular font used in normal handwritings (Fig. 2-4). However, due to special cases

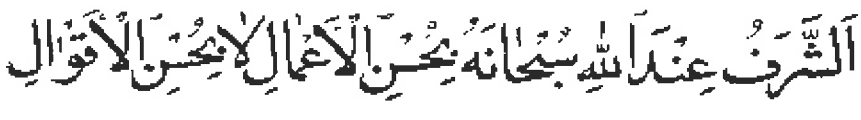


Figure 2-3: Arabic text written in Naskh font [40]. Here, diacritics are included.

and exceptions, it is not an easy task to implement this font in machine printed documents.

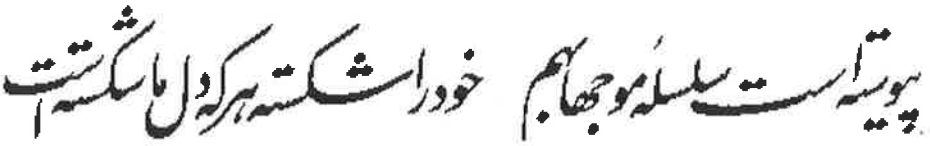


Figure 2-4: Persian text written in Nastaligh font [40].

Broken

This font (Fig. 2-5) is a subset of Nastaligh font which was invented to speed up handwriting. The reason that this font is suitable for fast writing is that the building strokes of characters have the smallest complexity among all the fonts. Some of the characters in this font are extended by long curves at the end. Because of unusual connections between characters, segmentation in this font is the most difficult in comparison to other fonts.

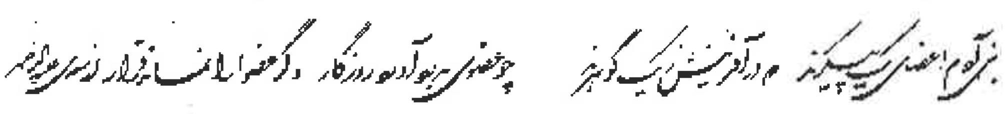


Figure 2-5: Persian text written in Broken font [40].

Mohaghegh

This font is one of the earliest fonts extracted from the Kufi font and was a very popular font in Islamic countries (Fig. 2-6). Another extension of this font called

Rayhan was mostly used for writing religious texts.

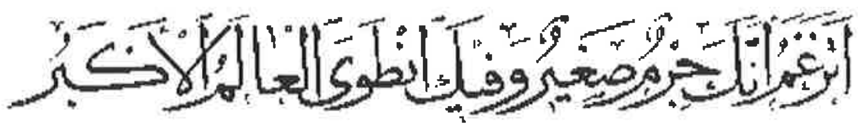


Figure 2-6: Arabic text written in Mohaghegh font including the diacritics [40].

Thuluth

This is also an old font which is sometimes called the '*Mother font*'. This font was also extracted from the Kufi font but it uses more circular lines than Kufi. Characters and words written in this font are condensed and brought close to each other (Fig. 2-7), hence there are more crossings between subwords. Thus segmentation for this font is very difficult.

Figure 2-7: Arabic Text written in Thuluth font including diacritics [40].

Diwani

As the name Diwani stands for "governmental", this font was very popular in government and royal offices (in about the 14th century governments started to use this font). It has less decorative shapes than the other fonts (Fig. 2-8), but there are still more connected words and subwords than in both the Naskh and Nastaligh fonts.

Figure 2-8: Arabic text written in Diwani font [40].

Roqa

This font is more popular in Arabic than Persian. It has the same rank as Broken font in terms of simplicity of reading and writing. A sample Arabic text written in Roqa is shown in Fig. 2-9.

Figure 2-9: Arabic text written in Roqa font [40].

2.3 Problems of Handwriting Recognition

In the previous section we described the characters and language writing characteristics of Persian and Arabic. In this section we present the characteristics of Persian and Arabic texts from a pattern recognition points of view. Like any other pattern recognition application, 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 also look at the characteristics and difficulties of a recognition system for handwritten Persian and Arabic texts.

2.3.1 Segmentation Problem

One of the most prominent difficulties in Arabic and Persian 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 Persian and Arabic writings, segmentation is a very difficult task, even for printed texts.

Handwriting Types

As described in [116], handwriting in Latin and English languages 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.

The above categories are listed in the order of increasing difficulty of recognition. Fig. 2-10 shows different types of handwriting texts in Latin and English languages. In Persian and Arabic, there exist only 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 this problem is interaction of segmentation with recognition.

Overlap in Handwritten texts

Horizontal spaces between the subwords or between discrete characters is of 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 Persian and Arabic handwritten texts there are different types of overlap between characters. We categorized them into three classes: overlap without touching, overlap with touching, and overlap caused by unusual touching.

Overlap without touching: In this case the subwords of a word 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;

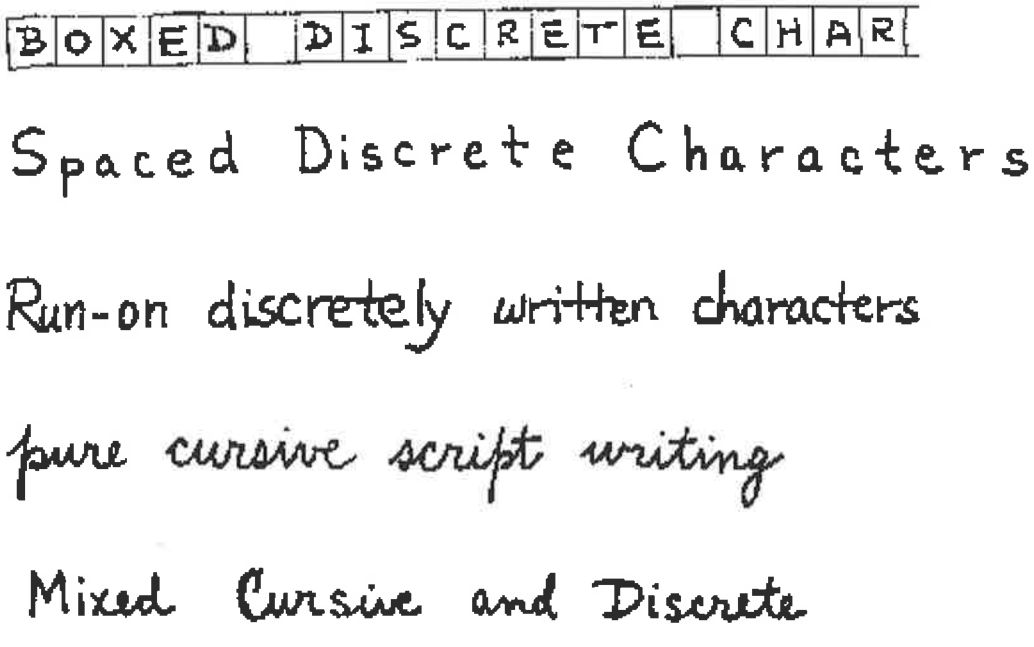


Figure 2-10: Different types of handwriting (from [116]).

this is marked as type a in Fig. 2-11. In this figure, a letter in a circle connected to a vertical line shows the location and 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 Nastaligh a character may connect to its succeeding character from the top (marked as type b in Fig. 2-11). In this case these two characters align vertically.

Overlap caused by unusual touching: This often occurs in broken font. In this case, sometimes those characters which are not connectable from the left or right side, are connected (marked as c in Fig. 2-11).

All of these types of connections are very common in normal handwritings, and for none of them are vertical projection and base-line approach for segmentation is applicable. Fig. 2-12 shows an example of using vertical projection. This figure shows that projection method does not provide adequate information for segmenting the words into their building characters. For a survey on segmentation methods in handwritten texts refer to [25].

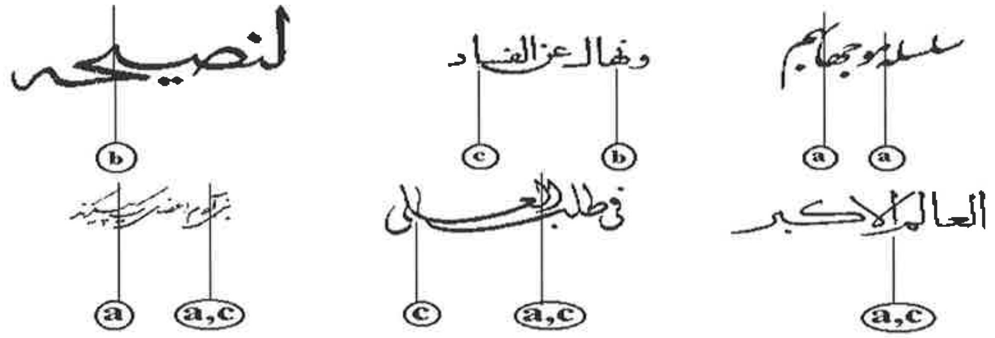


Figure 2-11: Different types of overlap between characters. "a" denotes vertical without touching, "b" vertical with touching, and "c" unusual touching.

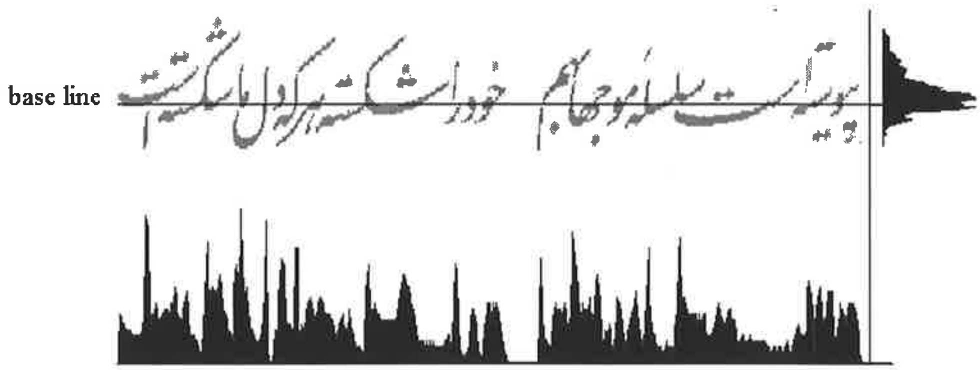


Figure 2-12: Vertical projection of a line of Persian text. This method is not applicable in Persian and Arabic segmentation.

2.3.2 Character Primitives

People learn to combine so called *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 primitive strokes. The primitive strokes of characters of Naskh font are shown in Fig. 2-13.

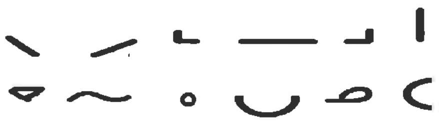


Figure 2-13: Primitive strokes of Persian and Arabic Nask font.

2.3.3 Number of Classes

Because Persian and 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 can be finally combined with the outputs of a character body recognizer system at the post-processing stage. With these assumptions, the actual number of patterns to be recognized is always different from the number of characters in a font.

2.3.4 Handwriting Variability

Handwriting is a free-form process, and 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 their 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 preferences.

Handwriting characteristics 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 individuals are taught to write in the same way. When these individuals are first learning to write, there are differences in their ability to perform the task, and the results are not all the same, but the true individual writing style differences appear only over time. As we grow and mature 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.

2.3.5 Confusion of Similar Characters

One of the most important limits for achieving a high recognition rate for handwritten Persian and Arabic characters is confusion between the very similar characters. As was mentioned before, there are groups of similar characters which only differ in position and number of dots like JEEM, CHA, 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 can become similar. We divide the problem of similarity of characters into five categories:

Similar shapes: this is a group of characters which have the same body shape. Regardless of what type of feature extraction technique we use for these character we always have a large probability of confusion. Fig. 2-14 shows some groups of similar isolated characters of the Naskh font.

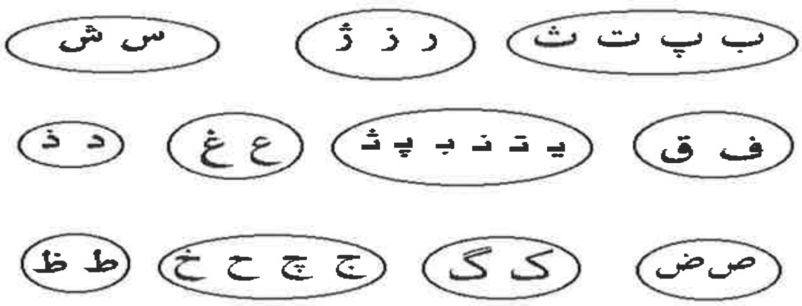


Figure 2-14: Groups of some similar isolated characters for the Naskh font.

Similar when rotated: the body of some characters become similar when one of them is rotated. For example if the character "MEEM (beginning) " rotates 90° anti-clockwise, it becomes very similar to the character "HAA (end)" (Table 2.2). For these characters, any extracted feature which is rotation invariant causes them to be confused.

Similar when scaled: for these characters, the scaled version of one becomes similar to the other. 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. We will discuss this topic in a later chapter.

Similar feature vectors: Depending on the method of feature extraction, some characters may have the same feature vectors. We will also discuss this in greater detail in a later chapter.

2.3.6 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.

2.3.7 Problems of Dots and Diacritics

As we defined before in this chapter, diacritics are the marks which sometimes, specially in Arabic, are added to a letter to indicate a special pronunciation. Another problem with handwritten Persian and Arabic documents is that of dots and their locations. Many Persian and Arabic characters have a number of dots. There are different number of dots which are located in different positions within the character (see Table 2.2). Dots are considered as complimentary characters [7]; any erosion or deletion of these complimentary characters results in a misrepresentation of the whole character. This is specially 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 difficulty with recognition of dots in handwritten documents can be attributed to the following causes:

Misplacement: 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 clues such as context to recognize the actual location of a dot or a group of dots.

Change in shape: There are 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. Fig. 2-15 shows a word using character SHEEN with three dots, but of different shapes.

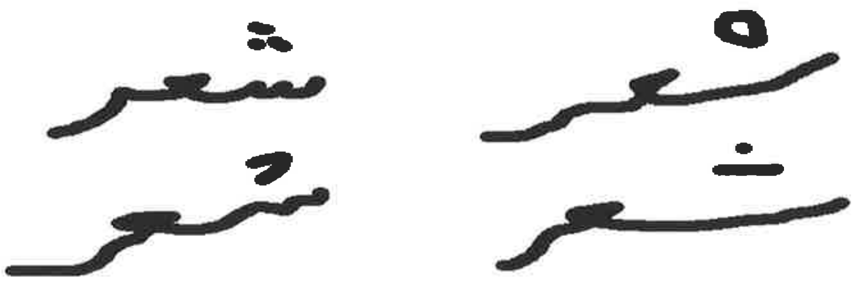


Figure 2-15: A Persian word using the character "SHEEN" with different shapes for its three dots.

2.3.8 Lack of Handwritten Data

One of the main problems of Persian and Arabic handwritten recognition systems is that there are no standard data sets to evaluate the developed algorithms. All research done for these two languages 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 [98]:

- Should the data set include segmented words or isolated characters?
- What is the minimum number of words that covers all the possible practical patterns?
- What is the criteria for readability of patterns and what are the conditions for which a pattern should be rejected?

2.4 Summary

In this chapter, we have introduced Arabic and Persian character sets and their fonts for handwriting. A wide variety of fonts and mixture of fonts pose difficulties for designing a general purpose multi-font handwritten recognizer for Arabic and Persian documents. We have presented some of these difficulties by showing some examples. Even by using the best feature extraction technique and the best classification methods, it is impossible to completely resolve the problem of similarity between characters. This means that context information must be used at a later stage for accurate word and text recognition.

Chapter 3

Review of the Literature

3.1 Introduction

This chapter is devoted to a brief introduction to the current trends in character recognition, and also a comprehensive review of the research undertaken for Persian and Arabic character recognition. We review papers published about the different approaches used by the researchers for different parts of a recognition system for Persian or Arabic texts, including segmentation, preprocessing, feature extraction, recognition, and post-processing. In this review, we only include those publications to which we had access.

The chapter starts with a brief study of the current research directions for character recognition systems in the next section, and then presents review of the research work conducted for different units of a recognition system for Persian or Arabic languages.

3.2 Research Directions

Many OCR systems have been developed over the last two decades [86], but more work is still required to attain results close to human recognition abilities. This is especially 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 feature extraction and classification of both handwritten and printed texts [64, 46, 84, 111]; however, no simple scheme is likely to solve all the problems associated with the largely variable input data in handwritings.

Of the different strategies for feature extraction, 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. 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 [112].

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 [53]. 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 deficiencies of conventional pattern recognition techniques, the application of different types of neural networks in the area of handwritten character recognition has been increasing [66, 33]. For a comparison of statistical and neural classification techniques for recognition of handwritten numerals see [24]. Different types of neural network architectures including back propagation [74, 73], Neocognitron [43], associative memory networks [130], and ART networks [44], have been employed in character recognition systems.

Variability of handwriting introduces a kind of fuzziness in handwritten recognition systems. Neural networks and fuzzy logic are complementary tools to deal with this problem. Fuzzy sets allow their members to belong to them partially. The membership defines how much an element belongs to a set. A new era of using fuzzy set theory for character recognition has commenced and many have successfully applied fuzzy logic concepts to different parts of the OCR system. These approaches include fuzzy graph theoretic approach [2], unsupervised character classification [26], self-organizing maps and fuzzy rules [28], feature extraction [78], decision operator [79], allograph modeling for cursive script recognition [91], and fuzzy integration [129].

Although Hidden Markov Models (HMM) have been widely used especially for on-line cursive recognition as well as speech recognition[45], some researchers applied the HMM technique for off-line systems, mostly in word recognition application. HMM was used for representation of printed characters in noisy document images [37], and for handwritten word recognition [57].

Despite the large number of algorithms developed for character recognition, the problem is not yet solved completely. Currently available systems, even commercial ones, have a set of limitation on handwriting styles or print quality. In the future, emphasis will be on the recognition of unconstrained handwriting. The potentials for OCR algorithms seems to lie in the combination of different techniques and also greater usage of contextual information. It has been proved that the performance of a handwritten recognition system can be improved by combining the outcomes of multiple classifiers [61]. Neural networks can not only be used as excellent feature extractor and classifiers, but also as trainable, and good classifier combiners [62].

3.3 Persian and Arabic Character Recognition

Research in the field of Arabic character recognition, as reported in [11], began in 1975 by Nazif [88]. This shows a very late start in the research as compared to earlier efforts in Latin dating back to the middle of 1940s [48]. The number of publications in Arabic and Persian character recognition indicates that there has been more research on Arabic character recognition than on Persian character recognition. However, almost all of the techniques used for Arabic character recognition are directly applicable to the Persian systems, although they may only need to be slightly modified to include four extra characters of Persian. To the best of our knowledge, the first publication for machine recognition of a printed Persian character set is a paper by Parhami and Taraghi [90].

In this section, we present a review of off-line recognition of handwritten and printed Persian and Arabic documents. There have been few survey papers on the recognition of Arabic characters[12, 67, 104]. The latest comprehensive review on the recognition of Arabic characters, to the best of our knowledge, was made by Badr and Mahmoud [6]. We will try, however, to cover more recent papers in our review and also include more publications on research for the Persian character set. We start with brief review on psychology of reading Persian words, followed by a review of data collection and analysis. OCR systems are then reviewed in two sections: one dealing with segmentation, the other with recognition.

3.3.1 Psychology of human word recognition

Although there is a large number of research publications in the field of psychology of reading words and letters (see for example [83]), the only paper which studied the psychology of Persian words and letters, to our knowledge, is a paper by Baluch and Shahidi [20]. In a section of their paper on the type of information the beginner or less skilled readers in Persian use to recognize words, they explained:

> "There seems to be agreement that recognition of a word is possible through at least two independent routes: an assembled route, based on rule-based conversions of subword orthographic units onto phonological units which are appropriate for oral reading or semantic recognition, and an addressed or orthographic route in which a word's meaning or pronunciation is directly looked up in a mental lexicon."

For skilled readers, the addressed orthographic route was concluded to play a more important role in reading than the assembled route; however, there is no complete agreement between the researchers about the role of these two routes in beginners or less skilled readers. Baluch and Shahidi in their paper address this issue and by running an experiment concluded that the beginner readers of Persian engage more in phonological coding for the recognition of words. Thus beginners rely more on diacritic information than skilled readers. For skilled readers of Persian (and Arabic), the transparency of a word's spelling is not crucial to the route used in oral reading or lexical decisions.

3.3.2 Data Collection and Analysis

As we mentioned in the previous chapter, one of the problems of handwritten recognition of Persian and Arabic texts is the lack of a standard data set. Such a data set could be used for comparison between all algorithms developed for the recognition of printed and handwritten Persian and Arabic documents. The only publication which deals with this problem, and we had access to, was a paper by Safabaksh and Shayghan[98]. In this paper, they presented a set of rules for evaluation of data sets for the Nastaligh font. They also collected three sets of handwritten words called AMIR-KABIR-1, AMIR-KABIR-2, and AMIR-KABIR-3, each set was divided into training and testing sets. AMIR-KABIR-1 included 5740 words, 1800 signs, and 1600 numerals stored as gray-scale images. Binary version of AMIR-KABIR-1 is called AMIR-KABIR-2. They considered 16 rules of connecting characters of Nastaligh font, and selected those words from AMIR-KABIR-2 data set which satisfied these rules, and then segmented these words into characters and stored these characters in a collection called AMIR-KABIR-3. Unfortunately, we did not have access to any of these data sets.

The criteria they used for measuring the readability of samples, was an empirically-derived formula (Eq. 3.1). This equation was introduced by a group of famous Iranian calligraphers to evaluate a piece of handwritten text. The readability is estimated by

$$J = 0.25S + 0.125Y + 0.225A + 0.2F + 0.2M \tag{3.1}$$

where S is a score for normality of the shape of characters within the text, Y is a measure for similarity between the shape of the same characters in the text, A is a score for size and aspect ratio coherency of characters, F is a measure for correct distances between subwords and between words, and finally M is a score of relative distance of the words from the base-line. Since all of the variables in the right side of the equation lie in the range $[0, 100]$ $J \in [0, 1]$. The readability score (J), is usually assigned by professional calligraphers, and the equation is an estimation for readability. There is no mathematical proof for this estimate, however, the results showed that values for J obtained from both the equation and professional calligraphers have a high degree of agreement. This measurement criteria was used by Safabaksh and Shayghan[98] for selection or rejection of a handwritten sample.

3.3.3 The Segmentation Problem

As we mentioned before, one of the most difficult part of a text recognition system for Persian and Arabic is its segmentation unit. This unit is responsible for breaking the texts into words, and then splitting the words into subwords, and finally splitting the subwords into individual characters. For the recognition of Persian and Arabic documents, five recognition strategies have been proposed to date [6] :

1. Segmentation-free recognition. The input is recognized as a whole word without any segmentation.
2. Recognition of already segmented characters. There is no need for segmentation.
3. Segmentation of words into characters. None of the methods reported are robust to handwriting style variations, and they are usually designed for printed text.
4. Segmentation of subwords or words into primitive strokes smaller than a character. In this approach, the primitive strokes are usually reconnected to each other to form a character or word in a later stage.
5. Recognition and segmentation working together. In this approach, the segmentation is a by-product of recognition stage.

Segmentation-Free Recognition

In this method, the characters of a connected subword are recognized without any segmentation in advance. El-Badi and Ramsis in [35] started from the extreme right of a subword and examined a set of columns of the image and tried to recognize the set as a character. If that fails, they add columns to the set until a character is recognized. Once a character is recognized, the set of columns are removed from the subword and the process is repeated until all the image columns of the subwords are examined. There are two problems when using this approach for handwritten texts. One is that if the recognition system fails to recognize a separated part, it will affect the recognition of the reminder of the subword. The other limitation of this approach is that in handwritten texts occasionally characters touch each other vertically. This means that for a set of columns, there may be more than one character to be recognized (Fig. 4-1).

Al-Badr and Haralick [5] called their technique a segmentation free approach although in their method they divide Arabic words into a set of primitives. The primitives and their locations are then detected using mathematical morphology operations. At the time of recognition the detected primitives are combined into characters. As they stated, the proposed system is dependent on the font type and size [4]. The reason is that in order to recognize new fonts, a new set of primitives needs to be produced.

Some proposed systems recognize a word as a whole unit of a pattern[16]. This approach is limited to recognizing a small set of predefined words, e.g., a computer program written in a particular software language.

Segmentation Approaches

The vertical projection histogram of a text line has been widely used as a common method for subword separation [13]. As we discussed in a previous chapter, this method, however, fails when a vertical overlap occurs between characters.

The angle that each character forms when it is joined with another character at the base-line was used by Bushofa and Spann [23] to choose the correct position of segmentation of an Arabic word. In their technique, the lower part of a subword contour, the part which falls below the base-line, is first examined for any possible touching characters or an "(end) YEH". These two cases were segmented before the main procedure started. The upper contour was then examined for candidate points for segmentation. By tracing from left to right, starting from the first contour point above the base-line, any minimum in contour coordinates between two peaks , is considered as a segmentation point. A point in the contour is considered as a peak if its value is greater than a threshold. If no peak point is found after a minimum point, the point is neglected. Furthermore, if two or more minima between two peaks satisfy the threshold condition, the point nearest to the first peak was taken and the remaining points are neglected. Bushofa and Spann applied this method to segment text scanned from books and newspapers in two fonts and four different sizes.

Al-Sadoun and Amin [7] presented a complete system for the recognition of Arabic text, including preprocessing, thinning, binary tree construction, segmentation, and recognition stages. They introduced a binary tree segmentation technique to split a subword into its characters. After preprocessing and thinning of a word image, a binary tree was built which included all the information describing the structure of the image. This involved tracing the image by a 3×3 window and recording the structure of the traced part by a set of image primitives. These primitives were the eight Freeman codes [42]. After the binary tree was generated, it was smoothed to minimize the number of nodes in the tree and the length of Freeman codes, and also to reduce the effect of noise in the thinned image. The next step was segmenting the tree of a subword into its characters, which was performed by traversing the binary tree and using a set of rules for segmenting the subword.

Line segmentation in [56] is the process of splitting a line of text into so-called subword glyphs and secondary glyphs. Hassibi defined a subword glyph as the bitmap representing a connected set of letters describing a subword, a word, a letter, or a ligature (a special type of connected characters). A secondary glyph was defined as the bitmap representing a dot or a group of dots, and diacritical marks. Each subword glyph was the input to a preliminary segmentation process, where each subword was broken into so-called *Meta character glyphs*. A Meta character glyph is defined as a glyph representing a single valid Arabic character, a valid ligature, or a character stroke.

A segmentation technique for Arabic words was proposed by Amin [14] and was applied to the binary skeleton of the word. The original image of the word was pre-processed in order to produce the skeleton. Then the thinned image was traced to construct a binary tree with all the information describing the structure of the image. Using 8 Freeman coding primitives and two primitives for loops and double loops, a binary tree was built and then smoothed to reduce the number of nodes and the Freeman code string in the information field of the nodes, and to eliminate or minimize any noise in the thinned image. The binary tree was divided into several subtrees such that each subtree represents a character.

Instead of splitting a subword into characters, Almuallim and Yamaguchi [10] segmented connected characters into a set of strokes. After classifying these strokes, their relative positions were used to combine them into a string of characters. In order to reduce the complexity of the combination process caused by the large number of primitive strokes, they attempted to define strokes so that the number of the strokes of a word became as small as possible. Strokes were clustered into one of five groups of strokes, and then within the clustered group the stroke was classified by using a set of 7 geometric features.

In a similar fashion to the work of Almualim [10], Goraine and Usher [47] used Freeman coding to segment the words into principal strokes, which are strings of coordinate pairs, and secondary strokes which are additions to the principal strokes. By using 8 directions for stroke coding, they introduced ten primitives codes which were defined as references.

Parhami and Taraghi [90] presented a technique for the segmentation of printed Persian texts. The algorithm was based on a fundamental property of the Persian script. Persian font design is done by using a rectangular-tip pen having a length much greater than its width. As the designer moves the pen at certain angles to generate each symbol, lines with varying thicknesses appear. At the unique connection point of two adjacent symbols, the pen moves horizontally on the connection axis to produce a line with maximum script thickness. Also, there is no symbol overlap at the connection point. They used this characteristics of the Persian text for the segmentation of printed texts in a newspaper title font.

Another segmentation method for Arabic typewritten texts was implemented by El-Sheikh and Guindi [36]. The segmentation process was essentially based on the calculation of the distance between the two extreme intersections of the outer contour with a vertical line. If in the recognition phase, a character was rejected, then the subword would be re-segmented with new parameters.

3.3.4 Recognition Systems

The two main parts in a character recognition system are the *feature extraction* and *classifier* units. The feature extraction technique is usually applied to the isolated characters and it selects a set of features that uniquely identifies that character. The selected features should efficiently discriminate between patterns of different classes, but should also be similar for patterns within the same class. Suen [111] has a good survey on different methods of feature extraction employed for hand-printed character recognition. Al-Badr and Mahmoud [6] reviewed the different stages of a character recognition system for the Arabic character set. They categorized different feature types into the following four main groups:

Structural features describe a character by its geometry and topology, either by local or global properties[18].

Statistical features use statistical measures of the character matrix as features[35].

Global transformations transform the character matrix from pixel representation into a more abstract level with lower dimensionality. Chain codes of skeletons or contours are examples of this method[15].

Template matching and correlation use pixel-by-pixel comparison of the character and a set of templates.

Of these methods, template matching is the most sensitive to distortion and noise; both the structural and statistical features are more tolerant to distortion and noise.

We review the recognition systems in two parts: printed, and handwritten recognition. In the first part we review the research on the recognition of printed Persian or Arabic characters and texts, and in the second part we present Persian and Arabic handwritten recognition systems.

A. Recognition of Machine Printed Characters

The Fourier spectrum of the character's projection on the X and Y axes was used by Saleh et al. [11] to recognize printed characters of the Naskh font. Once computed, the feature vectors were compared to the model feature vectors representing each individual Arabic character. Classification was based on a minimum distance measurement between unknown character's and model feature vectors. A recognition rate of 99.94% was obtained for one dimensional slice technique using features for the X- and Y-projections.

The average gray-scale threshold of the background image and the character images was used by Fathi and Broumand-Nia [39] to separate the characters from the background in a binary format. The character matrix was then divided into non-overlapping rectangular regions, and for each region some simple features like the ratio of black to white pixels were calculated. These features were then applied to a multi-layer Perceptron classifier. There was no report of the recognition rate of the system, however, the system was reported to be under test for industrial applications.

Al-Sadoun and Amin [7] presented a complete system for recognition of Arabic text. It consisted of preprocessing, thinning, binary tree construction, segmentation, and recognition sections. After preprocessing and thinning of a word image, a binary tree was built which included all the information describing the structure of the image. This involved tracing the image with a 3×3 window and recording the structure of the traced contours by a set of image primitives. These primitives were the eight Freeman codes [42]. After the binary tree was generated, it was smoothed to minimize the number of nodes in the tree, and the length of the Freeman codes, and to reduce the effect of noise in the thinned image. The next step was segmenting the tree of a subword into its characters. In the recognition phase, the binary tree of the subword was transformed into a single string according to a set of defined rules depending on the number of nodes for each character. This string was then matched with those which have already been computed and stored in a dictionary to find the character class. A recognition rate of 93.38% for printed characters and 76% for an old book was achieved.

Hassibi [56] used a neural network in the recognition phase of a machine printed Arabic OCR. The segmentation process produced Meta character glyphs which may be a single character, a valid ligature, or a character piece. Meta character glyphs were then recognized using classical classification techniques and a neural network was used to recognize the more difficult cases. Contextual information was used to join Meta characters into words, and in the post-processing stage, lexicons were used to improve the recognition rate. A neural network trained using training set derived from 350 images achieved a 99% recognition rate.

Al-Yousefi and Upda [8] introduced a statistical method for Arabic character recognition. In the first step, the character was segmented into primary and secondary parts (dots and zigzags). The dots and zigzags were isolated and identified separately. The features were extracted from the normalized moments of vertical and horizontal projections, and were then classified by a Bayesian classifier. For the isolated-form printed characters of three different fonts and five different sizes a classification rate of 85.5% was achieved by using linear discriminant analysis, while using quadratic discriminant analysis increased the recognition rate to 99.5%.

SARAT, which stands for Segmentation And Recognition of Arabic printed Text, was introduced by Mărgner[81]. The system was based on features of the upper contour. The reason for using this contour was that the upper contour of the main body of each character in Arabic contains most of the information about the character. By defining a set of geometrical features, Mărgner classified the characters by using a statistical minimum distance classifier. With 4110 characters of the laser printer font, a recognition rate of 99% was obtained, while for the inputs from documents printed by a dot matrix printer font the recognition rate decreased to 96.9%.

Goraine and Usher [47] used Freeman coding to segment Arabic words into strokes. The process of classification was done in two stages: in the primary stage the primitive type, the dot number, dot position, and loops were used as features, and in the secondary stage strokes were combined to form a character. They then employed a technique to solve any ambiguities between pairs of characters. For a total number of 830 printed characters of different sizes, they reported a recognition rate of 92%.

After segmenting the words into characters, El-Sheikh and Guindi[36] extracted the features by using a set of Fourier descriptors derived from the coordinate sequences of the character's outer contour. Topological features such as the height, width and the number of black pixels of a stress mark were used to classify the different stress marks. The developed system achieved a recognition rate of 99%.

To the best of our knowledge, the first publication for machine recognition of a printed Persian character set is a paper by Parhami and Taraghi [90]. In their paper they presented a technique for the automatic recognition of printed Farsi (Persian) texts. It was based on certain geometric properties such as relative width, existence of concavities and loops. They used 20 geometric features to form a 24-bit feature vector for each symbol. The feature vector obtained was matched against templates for the Persian symbols. In some cases where an exact match was not found, the algorithm looked for a best match in which the more reliable features were examined first. For newspaper headlines (big fonts), the recognition rate was 100%.

B. Recognition of Handwritten Characters

Dehghan and Faez[31] applied a set of moment invariants to recognize a pre-selected set of hand printed Persian characters. The moments which have been used were Zernike moments, Pseudo Zernike moments, and Legendre moments. They achieved a recognition rate of 96.92%, although, the data set they used was collected from samples of five famous Iranian calligraphers which are usually of a good quality.

After a well presented introduction to the characteristics of Persian character sets and handwriting, Fahimi and Sani [38] presented a neural network system for the recognition of handwritten characters. Due to difficulties of a general recognition system, they placed some constraint on the input patterns. The first stage of their research was to design a form to collect the data. Using a histogram of vertical and horizontal projections, each character was separated from the form boxes. To extract the features from the 25×25 character matrix, the value of the pixels at the crossings between ten selected rows and ten selected columns of the image were applied directly to a neural network classifier. The columns and rows were carefully selected to include all the essential information of the characters. A recognition rate of 90% for a set of isolated characters was reported.

Abuhaiba et al. [3] presented an automatic off-line character recognition system for recognition of segmented handwritten Arabic characters. They used a cluster-based skeletonization algorithm (CBSA), which they had already developed in [77], to convert the characters into a tree structure. They used the fuzzy set theory to model isolated handwritten characters as fuzzy attributed graphs. The graphs for an unknown input character were then compared to those of the models. They reported a 100% recognition for 420 character samples of a single writer, although, the recognition rate dropped to 73% when the system was tested with samples of a second writer. They then fine-tuned the system for the second writer to increase the recognition rate up to 97.4%.

An algorithm was developed by Abuhaiba and Ahmed [1] in order to restore the temporal information in off-line Arabic handwriting so that an on-line recognition algorithms may be used to advantage. After segmenting the secondary strokes that touch the main stroke, each subword was traversed from the starting vertex to the end vertex by solving the Chinese postman's problem for the subword graph. By applying special rules, the temporal information in the subword was restored. For a total of 1605 strokes, freely written by two writers, good performance of restoration of temporal information was reported.

3.4 Summary

In this chapter we presented a comprehensive review of the field of character recognition of Persian and Arabic languages. Despite the large number of recognition systems introduced in journals or conference publications, there is still much to do to achieve a reliable system. It is completely agreed that the performance of a machine which can recognize handwritten texts is still far from that of humans in terms of reliability, but they are much faster than humans.

Despite a very late start for Persian and Arabic character recognition, many approaches in different countries have been tried to overcome special characteristics of Persian and Arabic writings which pose difficulties in the character recognition systems of these two languages. The lack of communication between the research groups, poor financial support, and the lack of standard data sets are big constraints for implementing commercial systems, as compared to the number of implementations of character recognition systems in other languages.

Comparing the number of publications in both printed and handwritten recognition of Persian and Arabic indicates that many researchers were interested in the constrained typed document recognition systems, and there are very limited attempts to recognize unconstrained handwritten documents. Those few approaches for handwritten recognition reported high recognition rates, however, in almost all cases the high recognition rates rely only on collected data sets. Due to a lack of a standard data collection, no one can compare the outputs of different approaches for handwritten recognition systems. Research directions in Persian and Arabic character recognition systems is becoming more consistent and there are clear signs of a new era in this field. Research on Persian and Arabic character recognition is also becoming more intensive than before and commercial systems are becoming available. Because many papers in this field are written in either Persian or Arabic languages, survey papers can save time and effort for the beginner, especially those publications in well-known journals and conferences, written in English, which can attract more researchers, and more financial support for the field.

Chapter 4

Analysis of Handwriting

4.1 Introduction

Handwriting originates by forming a mental picture of letters and words in the brain. A signal is then sent to the hand through the muscles and nervous system to draw this mental picture. Due to the existence of an infinite number of ways of writing even the simplest letter combination, the output letter is almost never an exact match of the original mental picture. It is true to say that nobody can ever write a word exactly the same way twice in an entire lifetime. Every person has a handwriting variation determined by his or her physical writing ability, training, psychological status, and many other factors such as injury, illness, medication, drug or alcohol use, stress, the writing surface, the writing instrument, and personal preferences. We do not exactly know what is the original signal that is sent by the brain to the human motor system to produce a particular pattern. We humans, however, can cope with a variety of handwritings surprisingly well. This is not the case with machine recognition of cursive scripts.

Many researchers have attempted to model the biomechanical system of humans to investigate the problem of variation of the handwriting patterns; see, for example [101, 75, 118].Ward and Kuklinski presented a predictive model for many variability effects observed in handwritings [124]. The problem of handwritten variation is of special importance for Persian and Arabic due to the large variety of writing styles and personal preferences, which makes it almost impossible to include all types of handwriting styles in a single recognition system.

In this chapter we address two of the main problems associated with the recognition of Persian and Arabic handwritten characters, namely *similarity* and *variability*. We first introduce the components which cause different writing styles. Then a geometrical model for deformation is presented. The model introduces a range of possible distortions that may occur in patterns of handwritten characters. We show that any one of these deformations has a different impact on the individual characters. This model is then used to show the effects of various distortions on different Persian and Arabic characters. Although it is not possible to exactly calculate the distortion parameters from samples of handwritten characters, an attempt is made to estimate the order and the value of deformation parameters for the handwritten samples of the characters by using the distortion model. In the second part of this chapter, we address the problem of pattern similarity. By using the model presented for deformation, we show that some characters become more similar when they are distorted. Current approaches to these two problems of variability and similarity are reviewed in the final section, followed by some concluding remarks.

4.2 Pattern Variability

Recognition of unconstrained handwritten texts involves numerous problems. People are taught to write, at an early age, by copying the patterns of characters. The writing style is determined by many factors such as the brain's motor control, speed of writing, personal preferences, effort and fatigue [94]. Sometimes people use completely different shapes of characters from the shape and writing rules that they were taught. People often tend to minimize the writing effort, and they frequently produce illegible writings. The result is enormous variability in handwriting. All these factors make the recognition of handwritten characters very difficult.

4.2.1 Components of Handwriting Style Variability

A sample of a person's handwriting contains various global subject-specific parameters. However, these global parameters do not contain any information about the identity of the characters. Therefore, the handwriting patterns have to be normalized in terms of orientation, vertical size, and slant [119]. The wide diversity inherent in handwritten characters results from factors such as regional styles, differing writing instruments and psycho-motoric effects [123]. Some possible components of the handwriting style variability are:

- **Instrumental**. Various writing devices may produce different outputs. The types of variations made by different writing devices include line thickness, and salt and pepper noise.

- **Cursiveness**. Despite the fact that almost the same rules apply for cursive writing of both printed and handwritten Persian and Arabic texts, there exist, sometimes unwanted, cases of touching characters. In addition, different fonts have slightly different rules of touching.

- **Slant**. Slant is usually defined as the general direction of the vertical down strokes in handwritten characters [100]. In [51], handwriting slant is defined as deviation between the principle axis of characters and the vertical axis.

- **Shapes and length of ascenders and descenders**. The vertical size of a word or text line consists of three components: body, ascender, and descender heights relative to the base line. However, in handwritten Persian and Arabic text, there are many cases in which a character is written in a place quite different from its usual place in a word. In Persian and Arabic handwriting, ascenders and descenders may consist of a hook-like shape.

- **Connection between letters**. This usually happens because of the differences between different fonts. Fig. 4-1 shows an example of different ways for characters to touch each other in a word.

Figure 4-1: Two different version of connected characters in the word "MAHJOOB".

- **The base-line.** It is defined as an imaginary horizontal line upon which a text is written. Usually the base-line is a straight horizontal line; however, in Persian and Arabic handwriting the base-line may deviate from the horizontal line.

- **Size and aspect ratio.** A change in size may occur globally in all parts of the text or locally in a section of the text.

- **Orientation.** In some handwritten texts there may be a slight rotation for some characters; the range of rotation is usually small.

4.2.2 Deformation Model

In this section, we introduce a model for deformation analysis of characters. A function is then presented for each geometric transformation. Each pixel $P_{x,y}$ in the character matrix is represented by its Cartesian coordinates (x, y) relative to the matrix centre (as the origin). (x', y') is then the new location of the pixel after the transformation. More details about document image defects and perturbation models can be found in [19, 54].

The proposed model for deformation of the characters is shown in Fig. 4-2. In this figure, cascaded blocks represent different types of geometric perturbations applied to each pixel of the character matrix. Each $d_i(x_i, y_i; \zeta_i)$ is a function which operates on the image pixel at (x_i, y_i) and produces the transformed position of the pixel (x_{i+1}, y_{i+1}). Each ζ_i represents a distortion parameter for deformation of type i.

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = d_i(x_i, y_i; \zeta_i) \tag{4.1}$$

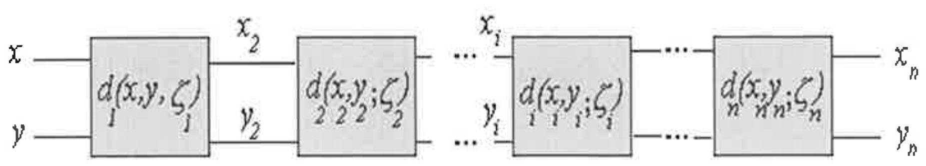


Figure 4-2: Block diagram of geometric transformation model for deformation.

At each stage of the deformation model, each pixel of the character matrix may have a different distortion parameter, but for simplicity we assume that in each stage, all the pixels in the matrix are transformed by the same deformation matrix.

The distortion model explains different types of deformation of the original pattern. Not all the possible sources of distortions can be modeled as a geometrical transformations. For instance the stroke thickness, which is another source of variation in handwritings, is not included in this model and should be considered separately. Moreover, in all cases we assume a uniform distortion for all the pixels when in reality the distortion usually has a non-uniform nature. Furthermore, the effect of random salt and pepper noise is not covered by this model. One also should note that the order by which the pattern is distorted by various transformations is very important; for each character we may have a different order of distortions, and if we change the order, we may get different patterns.

According to [54], the different geometric transformations are expressed by a second order polynomial transformation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \end{pmatrix} \begin{pmatrix} x \\ y \\ xy \\ x^2 \\ y^2 \end{pmatrix} \tag{4.2}$$

We now discuss some common sources of distortion applicable to patterns of handwritten characters.

A. Rotation

Rotational transformation is a well-known equation in standard mathematics. Each point in the matrix is transformed to a new point in the rotated image by the following equation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_1(x, y; \zeta_1) = \begin{pmatrix} \cos(\zeta_1) & -\sin(\zeta_1) \\ \sin(\zeta_1) & \cos(\zeta_1) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.3}$$

where ζ_1 is the distortion parameter which in this case represents the rotation angle in radians.

B. Slant

Slant is a very common distortion in human handwriting. Whereas humans deal with it without any problem; it makes the machine recognition of handwriting considerably more difficult. In general slant can affect the image both vertically and horizontally. Horizontal slant transformation (also called shear transform), can be defined as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_2(x, y; \zeta_2) = \begin{pmatrix} 1 & \zeta_2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.4}$$

where ζ_2 is the distortion parameter; in this case, it is the amount of horizontal deviation of the image. According to this equation, deviation of x depends on y. Fig. 4-3 shows images of character "HEH" deformed by horizontal slant with ζ_2 in the range $[-2, 2]$. The image at the centre ($\zeta_2 = 0$) shows the original image of the character.

For vertical slant transformation, the x-coordinate does not change but y linearly changes with x. The deformation transformation for vertical slant is:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_3(x, y; \zeta_3) = \begin{pmatrix} 1 & 0 \\ \zeta_3 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.5}$$

One may combine the two slant distortions in the following matrix operation:

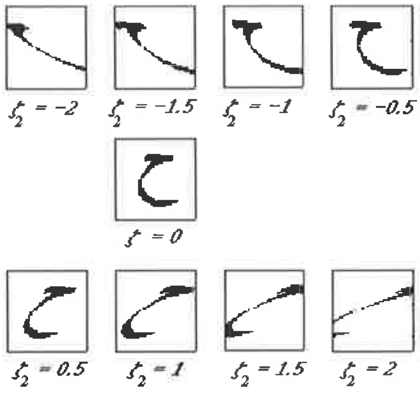


Figure 4-3: Patterns of the character 'HEH" deformed by horizontal slant.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_s(x, y; \zeta_3, \zeta_3) = \begin{pmatrix} 1 & \zeta_2 \\ \zeta_3 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.6}$$

Fig. 4-4 shows patterns of character "HEH" deformed vertically by the slant transformation.

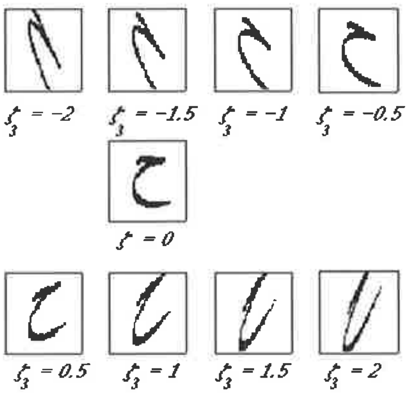


Figure 4-4: Patterns of the character 'HEH" deformed by vertical slant.

C. Perspective

Perspective transformation matrices are obtained by using the coordinates of the four corners of a square and its distorted version. The vertical, horizontal, first diagonal and second diagonal perspective transformations are given by, respectively,

Horizontal perspective

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_4(x, y; \zeta_4) = \begin{pmatrix} 1 & 0 & 0 & \zeta_4 \\ 0 & 1 & \zeta_4 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ xy \\ x^2 \end{pmatrix} \tag{4.7}$$

Vertical perspective

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_5(x, y; \zeta_5) = \begin{pmatrix} 1 & 0 & \zeta_5 & 0 \\ 0 & 1 & 0 & \zeta_5 \end{pmatrix} \begin{pmatrix} x \\ y \\ xy \\ y^2 \end{pmatrix} \tag{4.8}$$

First diagonal perspective

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_6(x, y; \zeta_6) = \begin{pmatrix} 1 & 0 & \zeta_6 & 0 \\ 0 & 1 & 0 & \zeta_6 \end{pmatrix} \begin{pmatrix} x \\ y \\ x^2 \\ y^2 \end{pmatrix} \tag{4.9}$$

Second diagonal perspective

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_7(x, y; \zeta_7) = \begin{pmatrix} 1 & 0 & \zeta_7 & 0 \\ 0 & 1 & 0 & -\zeta_7 \end{pmatrix} \begin{pmatrix} x \\ y \\ x^2 \\ y^2 \end{pmatrix} \tag{4.10}$$

Fig. 4-5 shows the output of the various perspective transformation of the character "HEH" using different distortion parameters.

D. Shrink

These transformation matrices look very similar to the perspective matrices. For example, the x-coordinates of the pixels in the horizontal perspective transform

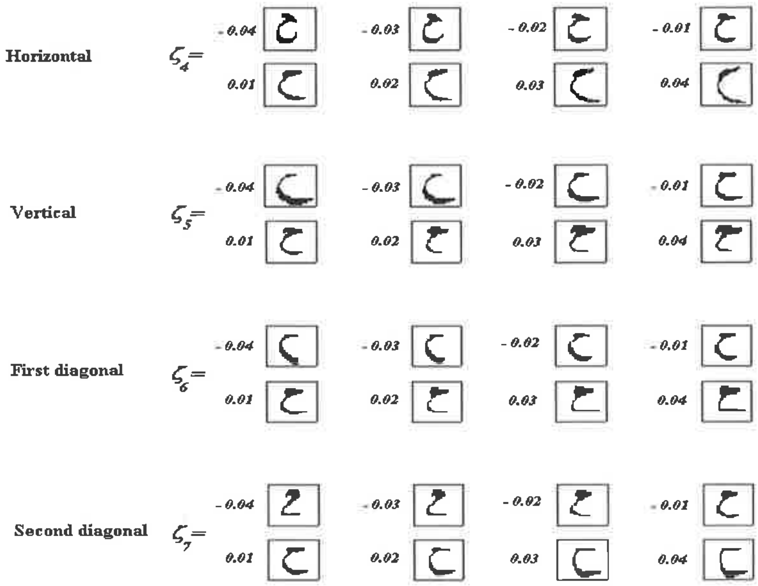


Figure 4-5: Patterns of the character 'HEH" deformed by horizontal, vertical, first diagonal and second diagonal prespective transformation with different distortion parameters.

have an extra quadratic term (x^2) which makes it different from the horizontal shrink transformation. Again there are four transformations corresponding to shrinkage in different directions:

Horizontal shrink

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_8(x, y; \zeta_8) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & \zeta_8 \end{pmatrix} \begin{pmatrix} x \\ y \\ xy \end{pmatrix} \tag{4.11}$$

Vertical shrink

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_9(x, y; \zeta_9) = \begin{pmatrix} 1 & 0 & \zeta_9 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ xy \end{pmatrix} \tag{4.12}$$

First diagonal shrink

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_{10}(x, y; \zeta_{10}) = \begin{pmatrix} 1 & 0 & \zeta_{10} & -\zeta_{10} \\ 0 & 1 & -\zeta_{10} & \zeta_{10} \end{pmatrix} \begin{pmatrix} x \\ y \\ x^2 \\ y^2 \end{pmatrix} \tag{4.13}$$

Second diagonal shrink

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_{11}(x, y; \zeta_{11}) = \begin{pmatrix} 1 & 0 & \zeta_{11} & -\zeta_{11} \\ 0 & 1 & \zeta_{11} & -\zeta_{11} \end{pmatrix} \begin{pmatrix} x \\ y \\ x^2 \\ y^2 \end{pmatrix} \tag{4.14}$$

Fig. 4-6 shows various versions of the character "HEH" shrunk with different distortion parameters. To fill any pixel discontinuities caused by the discrete nature of the images after the transformations, a morphological filter is used. The filter performs as a "close" operator. The reason to use this filter is its simplicity, and there are other methods such as "bilinear curve fitting" which may give a better results but are more complex and time consuming.

E. Scaling

It is considered as a scaling of x- and y-coordinates with different distortion parameters:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_{12,13}(x, y; \zeta_{12}, \zeta_{13}) = \begin{pmatrix} 1 + \zeta_{12} & 0 \\ 0 & 1 + \zeta_{13} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.15}$$

In this equation if ζ_{12} and ζ_{13} are different, there will be also a change in aspect ratio of the image. For ζ_{12} and ζ_{13} greater than 0, the image becomes larger and for $-1 < \zeta_{12}, \zeta_{13} < 0$, the image is reduced. A 1 is added to the distortion parameters to obtain an undistorted pattern when the distortion parameters are zero, consistent with all previous distortions.

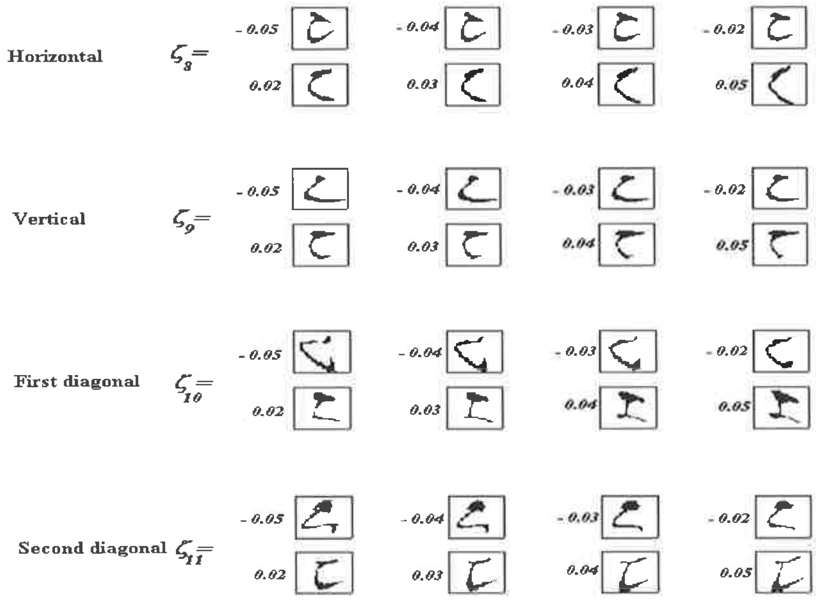


Figure 4-6: Patterns of the character 'HEH" deformed by horizontal, vertical, first diagonal and second diagonal shrink transformations with different distortion parameters.

F. Translation

As translation transformation does not change the shape of the image; it is different in this regard to previously defined distortions. However, translation can be considered as a geometrical transformation. Thus, we define the following matrix transformation for translation. For different ζ_{14} and ζ_{15} we have different translations in the x- and y-coordinates.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = d_{14,15}(x, y; \zeta_{14}, \zeta_{15}) = \begin{pmatrix} 1 & 0 & \zeta_{14} \\ 0 & 1 & \zeta_{15} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.16}$$

4.2.3 Variation Analysis of Persian and Arabic Characters

In this section, the effect of each of the above distortions on individual characters is investigated. Our proposal is that various sources of distortion have different impact on different characters. The reason is that the distribution of the character's pixels within the character matrix and a relative size of strokes differ from one character to another, and the effects of each transformation vary in different regions of the character matrix.

For this experiment we use samples of isolated printed Persian characters, each presented in a 48×48 binary matrix. The criterion we used for measuring the distortion is the average per pixel Euclidean distance between the original pattern and its distorted version. We define $S_i(x, y; \zeta_i)$ as the distance surface for a distortion with parameter ζ_i . Each point $S_i(x_j, y_j; \zeta_i)$ represents the Euclidean distance between a pixel (x_j, y_j) in the original pattern and the corresponding transformed pixel (x'_j, y'_j) in the distorted pattern

$$S_i(x_j, y_j; \zeta_i) = \left| \sqrt{(x_j - x'_j)^2 + (y_j - y'_j)^2} \right| \tag{4.17}$$

Define the normalized Euclidean distance between two patterns (d_i) as

$$d_i = \frac{\sum_{j=x_{\min}}^{x_{\max}} \sum_{k=y_{\min}}^{y_{\max}} S_i(j, k; \zeta_i) \cdot C(j, k)}{n} \tag{4.18}$$

where C is the binary character matrix, and n is the number of black pixels in the original character matrix. The characters are represented by black pixels on a white background.

Fig. 4-7 shows the surface plot of $S_i(x, y; \zeta_i) \cdot C(x, y)$ for the character (isolated) HEH for a horizontal slant distortion with $\zeta_2 = 0.1$. This figure shows the amount of distortion in different regions of the character using the Euclidean distance measure. The proposed distance measure, however, is not equal to the elastic matching distance of the two patterns which we will discuss later in a following chapter. The reason is that the number of pixels in the distorted and original patterns are not always the same and there is a possibility of insertion or deletion of points in the distorted pattern. Deletion of character pixels occurs when two or more points in the original pattern map to a single point in a distorted version, and insertion is performed by the process of filling the gaps caused by a geometric distortion. We penalize the insertion and deletion of pixels by adding the following term to Eq. (4.18)

$$\frac{(N_d + N_I) \cdot d_i}{n} \tag{4.19}$$

where $(N_d + N_I)$ represents the total difference between the number of black pixels of original and distorted patterns caused either by insertion or deletion. The final equation for average distance per pixel between the two character patterns is defined as

$$D_i = d_i + \frac{(N_d + N_I)}{n} \cdot d_i = (\frac{n + N_d + N_I}{n}) \left(\frac{\sum_{k=x_{\min}}^{x_{\max}} \sum_{j=y_{\min}}^{y_{\max}} S_i(k, j, \zeta_i) \cdot C(k, j)}{n} \right) \tag{4.20}$$

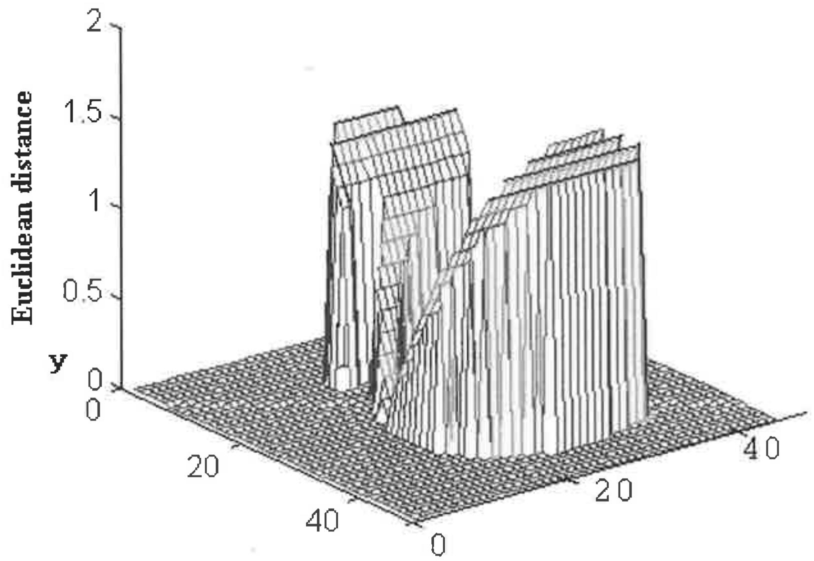


Figure 4-7: Surface plot of $S_2(x, y, \zeta_2) \cdot C(x, y)$ for the character "HEH isolated" for horizontal slant distortion ($\zeta_2 = 0.1$).

A set of 21 characters is selected (see Fig. 4-8). These characters are especially selected so as to cover all possible stroke shapes including straight vertical and horizontal lines, loops, and circular strokes. Each character is distorted by different distortion transforms described above for a range of distortion parameters. The average distance per pixel of each character is then plotted against the distortion parameter of each transformation.

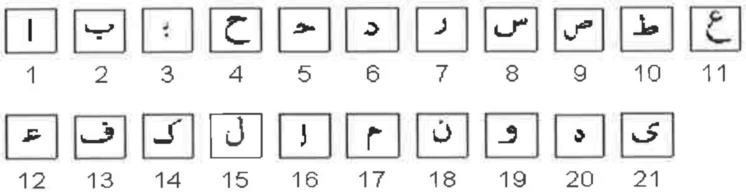


Figure 4-8: Character set used in variation analysis of Persian and Arabic characters.

A. Rotation

Rotational distortion changes the location of each pixel according to the distance surface shown in Fig. 4-9. The bigger the rotation angle, the larger the slope of the surface. Only the centre pixel has no displacement from its undeformed location. Moving from the centre, the amount of distortion becomes larger, and pixels that are far from the centre have the largest distortion. For small range of rotation, characters like "(beginning) HA ", whose pixels are spread evenly around the centre, have less distortion. Rotation has a more significant effect on characters like "AYN" , "HEH", and "GHAYN" which have pixels located far from the character's centre.

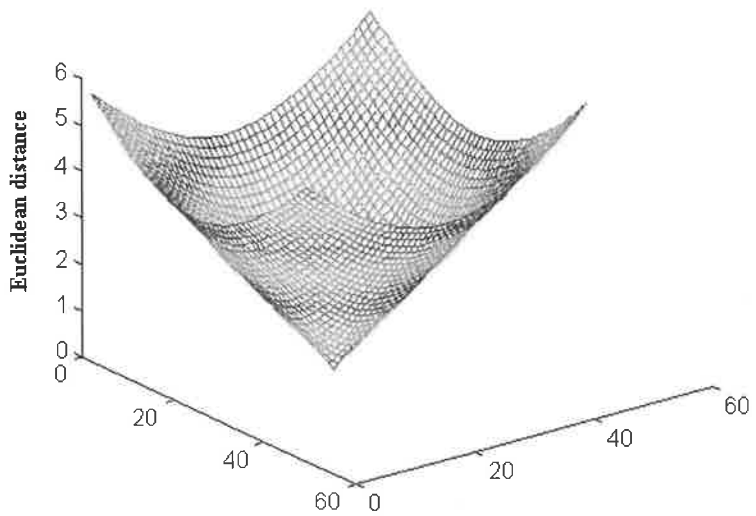


Figure 4-9: Euclidean distance of the pixels of the character matrix from their corrsponding deformed pixels in the rotated version with $\zeta_1 = \pi/18$.

The average distance per pixel of all the 21 selected characters, caused by rotational distortion, is shown in Fig. 4-10. The distortion parameter varies in the

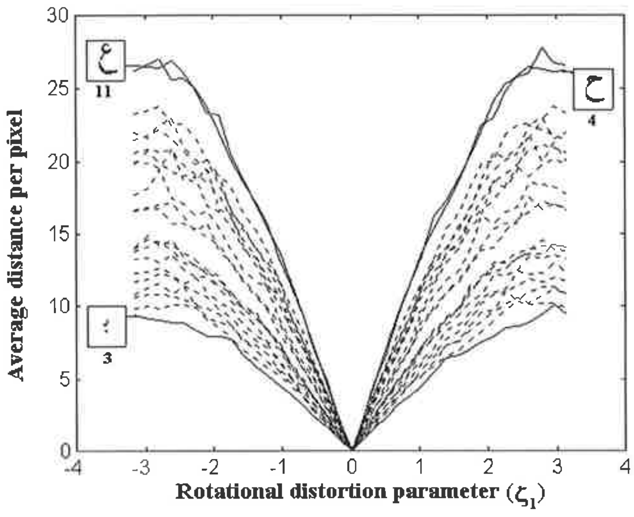


Figure 4-10: The average distance per pixel caused by rotational distortion of the selected Persian characters.

range $[-\pi, \pi]$. As shown in this figure, small characters like "(beginning) BEH " (#3 in the figure) have less sensitivity to rotational distortion than large characters like "HEH" (#4), and "AIN" (#11).

B. Slant

Horizontal slant has a greater effect on those characters like "(isolated) ALEF", which are constructed by vertical lines than characters like "(isolated) BEH", whose pixels are on horizontal lines near the origin and are thus more affected by horizontal slant. As shown in Fig. 4-11-a, vertical lines are affected more by horizontal slant distortion. Vertical slant, however, has more effect on characters like "(isolated) BEH" than characters like "(isolated) ALEF", as shown in Fig. 4-11-b. In both distance surfaces, the bigger the distortion parameter the larger the slope of the distance surface.

Figure 4-12 shows the effect of horizontal and vertical slant deformations on a set of Persian and Arabic characters. As discussed before and shown in this figure, the maximum distortion caused by horizontal slant is for the character "(isolated) AIN" as most of its pixels are located far from the centre of the matrix. For small characters like "(beginning) BA" and for the character "(isolated) BA", which looks

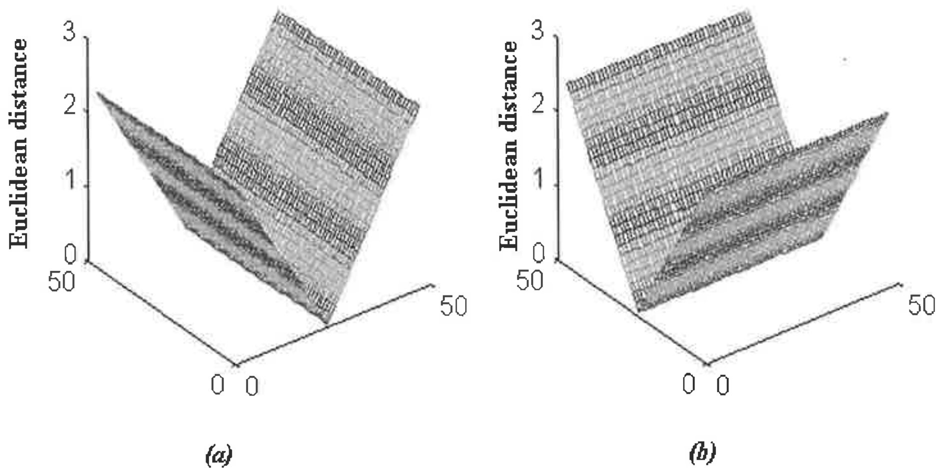


Figure 4-11: Euclidean distance of the pixels of the character matrix from their corresponding defromed pixels: (a) horizontal slant distortion with $\zeta_2 = 0.1$, and (b) vertical slant distortion with $\zeta_3 = 0.1$.

like a horizontal line about the centre, the effect of horizontal slant is minimal. Vertical slant, however, has the maximum effect on the character "(isolated) FA" and has minimum impact on characters like "(isolated) ALEF" and "(beginning) LAM", which consist of a vertical line near the centre.

C. Perspective

As shown in Fig. 4-13, both horizontal and vertical perspective transformations affect the characters in a very similar fashion as to horizontal and vertical slant distortions, respectively. The difference is that the distance surfaces are not as flat as for slant (Fig. 4-11). Diagonal perspective distortion distance surfaces are very similar to rotational transformation surfaces, and have more impact on characters whose pixels are more located further from the centre of the matrix.

Figure 4-14 shows the effects of various perspective deformations on different characters. Horizontal and vertical perspective have almost the same effect as horizontal and vertical slant distortions, respectively; however, the first and second diagonal deformations have more effects on characters like "(isolated) HEH" (#4) , "(isolated) FA" (#13), and "(isolated) AIN" (#11) which have more pixels far from the centre of the matrix than on the smaller characters like "(beginning) BA"

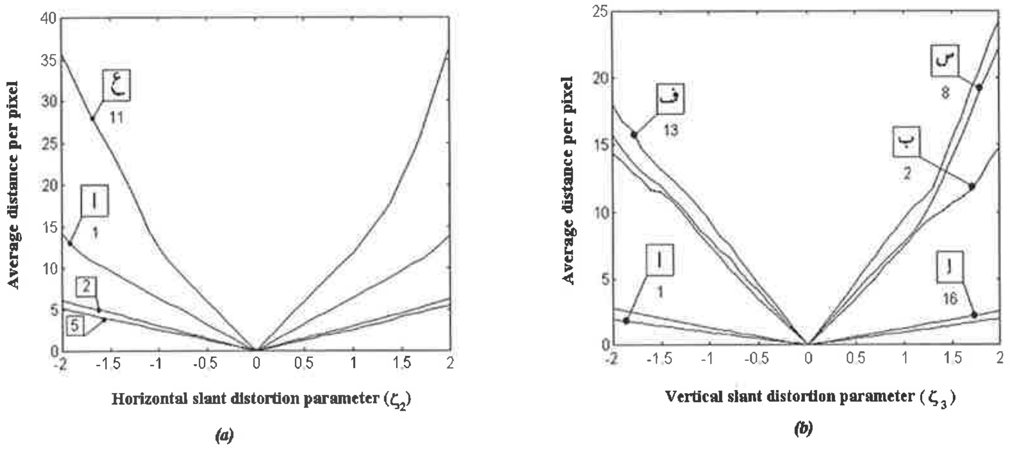


Figure 4-12: Effects of (a) horizontal and (b) vertical slant deformations on selected Persian and Arabic characters.

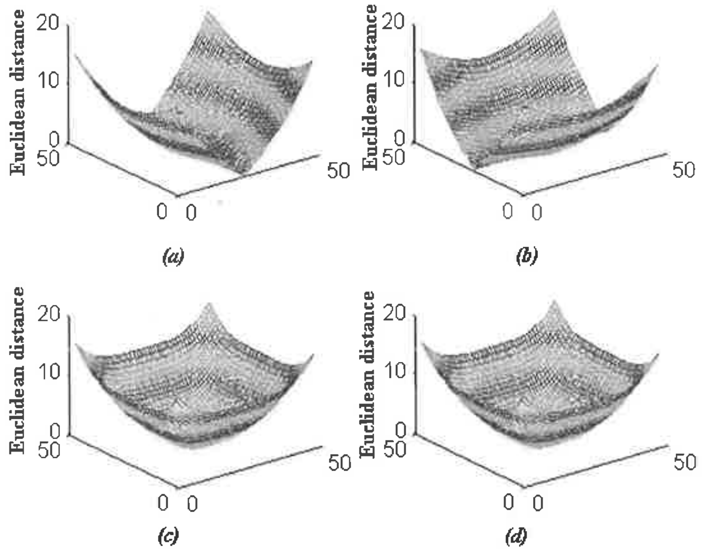


Figure 4-13: Euclidean distance of the pixels of a 48×48 character matrix from their corresponding deformed pixels: (a) horizontal perspective distortion with $\zeta_4 = 0.02$, (b) vertical perspective distortion with $\zeta_5 = 0.02$, (c) first diagonal perspective distortion with $\zeta_6 = 0.02$ and (d) second diagonal perspective distortion with $\zeta_7 = 0.02$.

(#3), "(beginning) HEH" (#5) and upright character like "(isolated) ALEF" (#1). Smaller characters such as "(beginning) BA" (#3), "(isolated) HA" (#20) are less affected by either first or second diagonal perspective distortions.

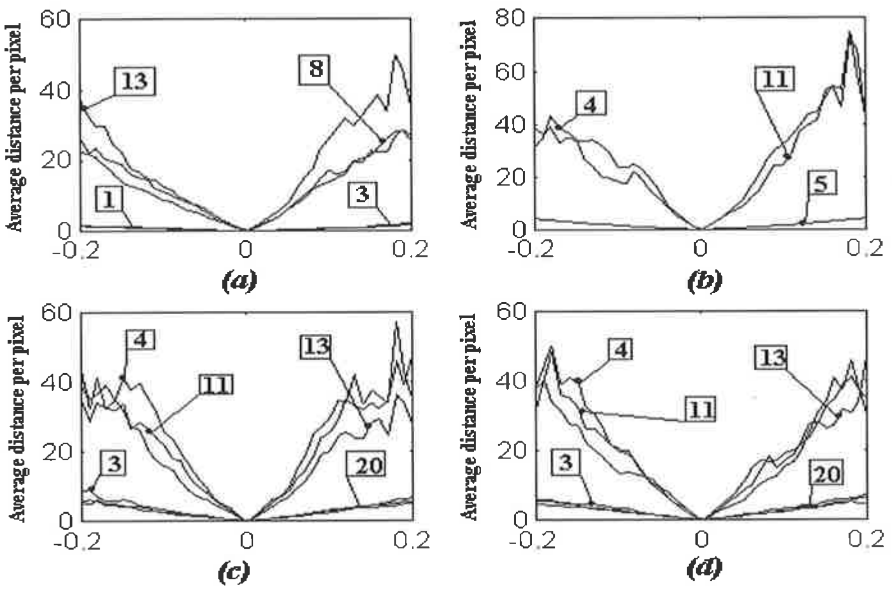


Figure 4-14: Average distance per pixel of selected Persian and Arabic characters deformed by *a)* horizontal, *b)* vertical, *c)* first diagonal, and *d)* second diagonal perspective transformations.

D. Shrink

Shrink distortion, as shown in Fig. 4-15, is slightly different from the previously mentioned distortions. It has different impacts on different regions of the character matrix, but in general, it has less impact on the regions near the centre of the matrix. Again, increasing the distortion parameters increases the slope of the surfaces. Figure 4-16 shows the effects of various shrink deformations on a set of Persian and Arabic characters.

4.2.4 Estimation of the Parameters

As we mentioned before, the order in which the pattern is distorted by various transformations is very important. By rearranging the order of distortion transforms in

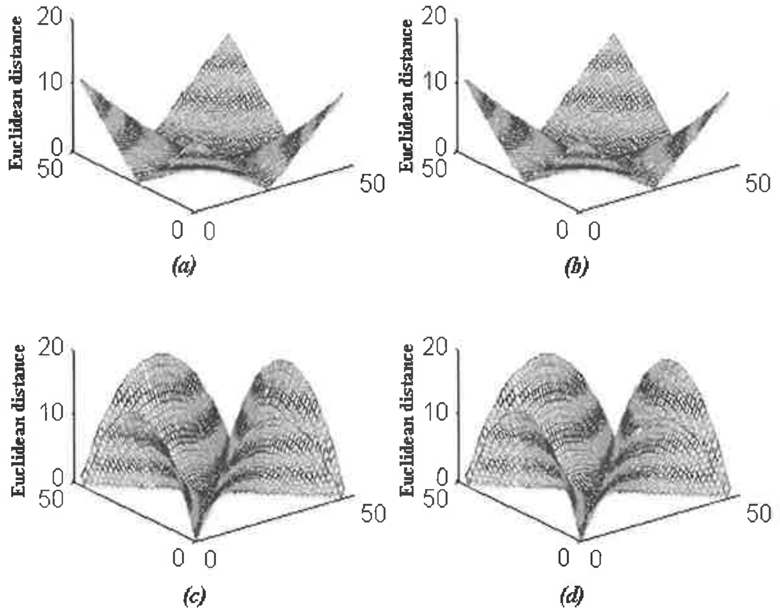


Figure 4-15: Euclidean distance of the pixels of the character matrix from their corresponding deformed pixels: (a) horizontal shrink distortion with $\zeta_8 = 0.02$, (b) vertical shrink distortion with $\zeta_9 = 0.02$, (c) first diagonal shrink distortion with $\zeta_{10} = 0.02$ and (d) second diagonal shrink distortion with $\zeta_{11} = 0.02$.

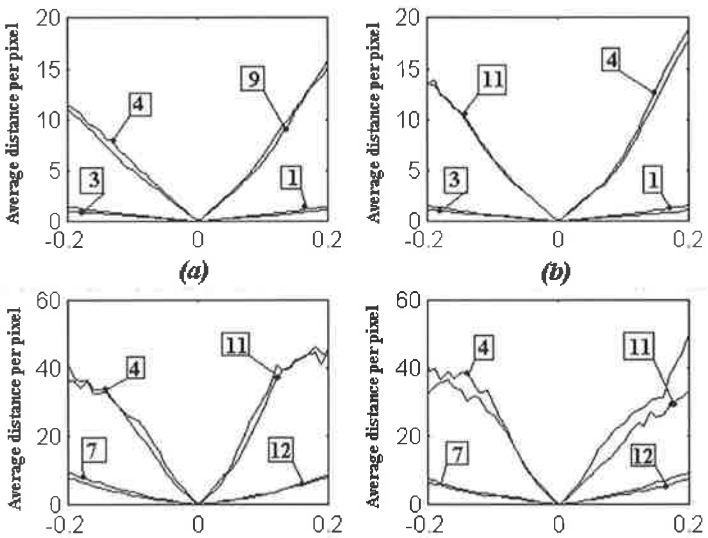


Figure 4-16: Average distance per pixel of selected Persian and Arabic characters deformed by: *a)* horizontal, *b)* vertical, *c)* first diagonal, and *d)* second diagonal shrink transformations.

Fig. 4-2, different distorted patterns are produced. If we assume that each handwritten character sample is a distorted version of an original pattern, then the problem is to find the best model of distortion which describes the deformation process from the original pattern to the final distorted pattern. The process can also be done in reverse, starting from a samples of handwritten character and distorting it with the model until the best match between the distorted version and the original template is found. To estimate this model, first we should find the order of distortion types, and then adjust the distortion parameters so that the model's output matches the original template of the character.

To find the order and to estimate the relevant parameters, we use a mixed mode distortion model. In this model, each handwritten character goes through a series of distortion stages. At each step, we evaluate all distortion types, and select the one which gives the maximum likelihood between the distorted and the template pattern. In this study, we use the pixel correlation between the distorted and final patterns as the likelihood function. Depending on the type of distortion, we assign a small value to the parameter, e.g. $\zeta_i = \pi/18$ for rotational distortion. These values, which are determined by trial and error, are used as steps in distorting a pattern in each stage. To find the best and smallest step size for each parameter, we start from $\zeta_i = 0$. A printed sample of characters are then distorted by changing the distortion parameter and the step size is the first small ζ_i which creates a visually sensible deformation on the character. We also could use the correlation between the original and distorted character and choose the step size from the smallest parameter that gives a certain correlation values.

For each type of distortion, both positive and negative values of the parameter (ζ_i) are examined. After deforming the pattern at each stage, the distorted pattern is applied as the input to the next stage, and the same process is repeated for the next stage. This process is continued until there is no further increase in the likelihood value. The output of the estimator is a string of codes, where each code represents a distortion type. If the same distortion occurs for consecutive stages, all of these stages can be replaced by one step of the same distortion with a parameter equal to the sum of the parameters of these stages. Fig. 4-17 shows an example of the parameter estimation process. An original handwritten sample of the character "(isolated) RA" undergoes a series of distortions indicated by the numbers in boxes. These numbers represent the distortion type in each step (see Table 4.1). As shown in this figure, the final version of the characters looks more similar to the template than the original handwritten sample, i.e. correlation coefficient of the samples increased from 0.12 to 0.627.

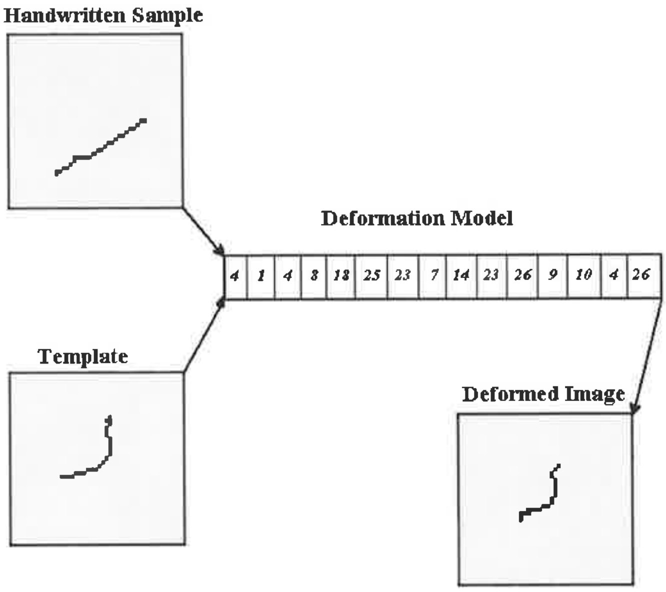


Figure 4-17: Estimated distortion parameters of the character "(isolated) RA" by using the distortion model.

Since each distorted pattern undergoes a series of distortion transformations to become as close as possible to the original pattern, the estimation process discussed above can also be considered as a *"warping"*. This technique increases the similarity and correlation between handwritten samples of characters and their corresponding printed samples, which we use as template patterns.

Distortion Characteristics

We now use the deformation model for real samples of Persian and Arabic handwritten characters. If we consider the negative and positive distortion parameters separately, then, according to Table 4.1, there are 26 different parameters to be es-

Table 4.1: Codes and step values of the parameter of various distortions used for estimating distortion parameters of handwritten samples.

Code	**Parameter**	**Step**	**Code**	**Parameter**	**Step**
1	ζ_1	$\pi/18$	14	ζ_7	-0.01
2	ζ_1	$-\pi/18$	15	ζ_8	0.01
3	ζ_2	0.1	16	ζ_8	-0.01
4	ζ_2	-0.1	17	ζ_9	0.01
5	ζ_3	0.01	18	ζ_9	-0.01
6	ζ_3	-0.01	19	ζ_{10}	0.01
7	ζ_4	0.01	20	ζ_{10}	-0.01
8	ζ_4	-0.01	21	ζ_{11}	0.01
9	ζ_5	0.01	22	ζ_{11}	-0.01
10	ζ_5	-0.01	23	ζ_{12}	-0.1
11	ζ_6	0.01	24	ζ_{12}	0.1
12	ζ_6	-0.01	25	ζ_{13}	-0.1
13	ζ_7	0.01	26	ζ_{13}	0.1

timated. For each sample of a handwritten character, we estimate the parameters by the method discussed in the previous section. The estimation process produces a vector of 26 elements whose elements are the distortion parameters for deforming a sample to get the maximum correlation with the printed character template. For each character we average these vectors over all the handwritten samples. The resulting vectors show the distortion characteristics of characters.

Fig. 4-18 shows the distortion characteristics of some of the characters. As the results show, handwritten samples of the character "(isolated) ALEF", which consists of a vertical line segment, has been distorted mostly by the vertical perspective (ζ_5) and vertical scaling (ζ_{13}) distortions. On the other hand, horizontal scaling (ζ_{12}) has more effect on the character "(isolated) HA". More graphs of distortion characteristics are presented in appendix B.

4.2.5 Deformable Models

According to the way *a priori* information of the pattern shape is used, shape matching methods can be classified into two main groups: *data-to-model* and *model-to-data*. In data-to-model methods, the raw image data is analyzed by feature extraction and the features are compared with the model. Conversely, in model-to-data methods, we start with a model (or template), and search the image for evidence

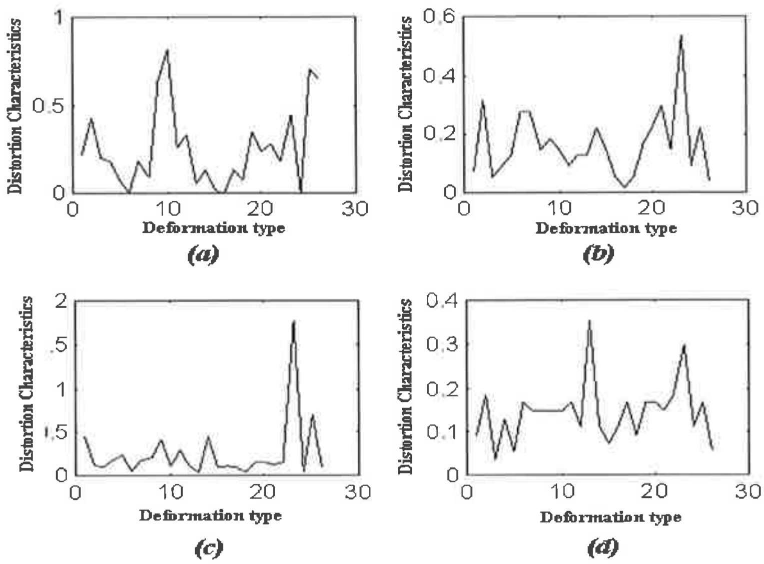


Figure 4-18: Distortion characteristics of different Persian characters: (a) "(isolated) ALEF", (b) "(isolated) BA", (c) "(isolated) HA", and (d) "(isolated) KAF".

supporting the existence of the model.

A major problem with the deformable models is that the fitting procedure between an unknown pattern and the model images is very computationally expensive and is usually an iterative task. One way to categorize the range of model-based approaches to handle the wide diversity inherent in handwritten documents is to consider the complexity of the procedure used to match the model to an unknown pattern. As shown in Fig. 4-19, moving from left to right across the spectrum, matching complexity increases while the number of matches decreases.

The basic idea in using deformable models for handwritten character recognition is that each character has a model which we call an original template. Each unknown character is classified by finding the model which is most likely to have generated it. The two important terms in assessing the fit of an unknown image (U) and a model (M) are the prior probability distribution for the distortion parameters (ζ) of a model ($P(\zeta \mid M)$), and the probability distribution over possible images given the distortion parameters ($P(U \mid M, \zeta)$). This framework has been used by many authors, e.g. [126, 50]. The probability of recognition of the unknown patterns of U

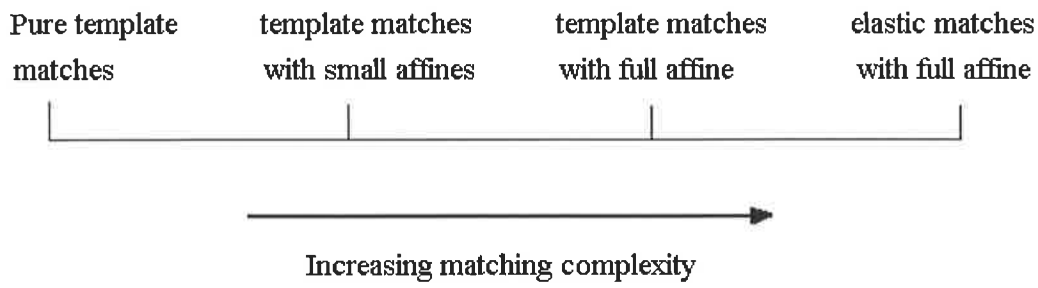


Figure 4-19: A spectrum of approaches to handling diversity in handwritten character recognition (from [96]).

as model M is calculated as

$$P(U \mid M) = \int_{\zeta} P(\zeta \mid M) \cdot P(U \mid M, \zeta) \cdot d\zeta \tag{4.21}$$

where the integration is calculated over the whole parameter space. The second term inside the integral represents the likelihood between the unknown pattern and the model M which is distorted by the parameter ζ. For large dimensions of distortion parameters, as in our case, the evaluation of this integral is computationally expensive. However, $P(U \mid M)$ peaks for a certain set of values of the distortion parameters (ζ^*).

Proposed System

Because at each stage, $2k$ distorted patterns are calculated (one negative and one positive parameter for k distortions), and for each distorted pattern the correlation coefficient must be examined, the proposed method for estimation of the distortion parameter is computationally expensive. We tested the method for real samples of handwritten characters. The aim here is to see, if the method increases the correlation between a handwritten sample and its template.

The deformation technique is the same method we used for the estimation of the distortion parameters (see previous section). Figure 4-20 shows the average pixel correlation coefficients between handwritten samples of the selected characters and their corresponding original templates before and after applying the proposed warping technique over all handwritten samples. As shown in this figure, in all cases the deformation technique has significantly increased the correlations.

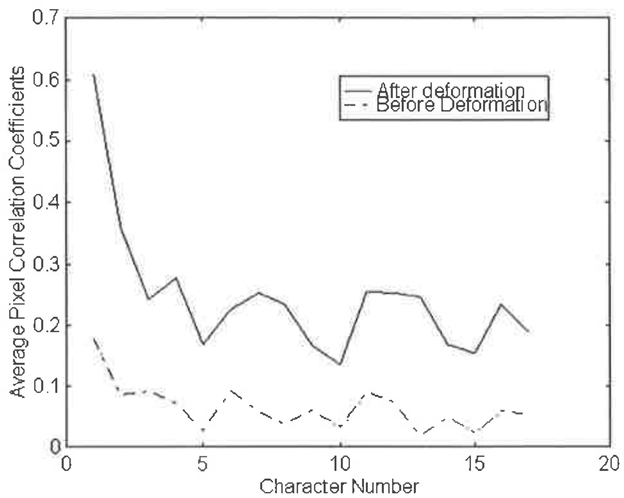


Figure 4-20: Average pixel correlation coefficients between handwritten samples of the characters and their templates (printed characters), before and after using the deformation model.

4.3 Pattern Similarity

As Tappert described in [117], the fundamental property of writing which makes communication possible is that differences between various characters are more than differences between different drawings of the same character. We humans can easily distinguish between similar patterns, and furthermore use contextual information to resolve any ambiguity, but for machines intended to recognize off-line handwritten documents the situation is quite different. The only available information is the image, and a machine should extract as much information as possible from that image. Increasing the number of similar patterns in different classes increases the number of potentially confusing patterns for the system.

Some Persian and Arabic characters are written very similarly. In a previous chapter, we gave examples of similar characters that are similar in shape. In this section, we investigate the similarity between the characters by using the distortion model. Using this model, we want to show that there are some characters whose distorted versions become very similar to the distorted version of other characters.

4.3.1 Similarity and Confusion

Large similarity between the patterns of two characters increases the probability of confusion between them during the classification stage. The similarity measure between any two patterns is directly proportional to the distance between them; the smaller the distance, the greater the similarity is. As shown in Fig. 4-21, if the distorted version of a sample of character C2 becomes too close to the cluster centre or prototype of another character (C1), regardless of the type of classifier or feature extraction technique we use, the probability of classifying C2 as C1 will increase. The actual probability of confusion, however, depends on the decision boundaries of the classifier. The similarity measure between any two patterns $P1$ and $P2$ is defined as:

$$S(P1, P2) = \alpha \cdot d(P1, P2) \tag{4.22}$$

where $d(P1, P2)$ is the distance between the two patterns and α is a normalization coefficient. If y is the output of the recognition system, then for an input pattern x' which belongs to the class $C2$, the probability of classifying it as belonging to class $C1$ is

$$P_{confusion} = P(y = C1|x' = C2) \propto S(x', O2) \tag{4.23}$$

where $O2$ is the cluster centre or the prototype of class $C2$. The above equations show that if the distance between an unknown pattern of a class to the prototype of another class decreases, it is more probable that it will be classified incorrectly during the classification stage.

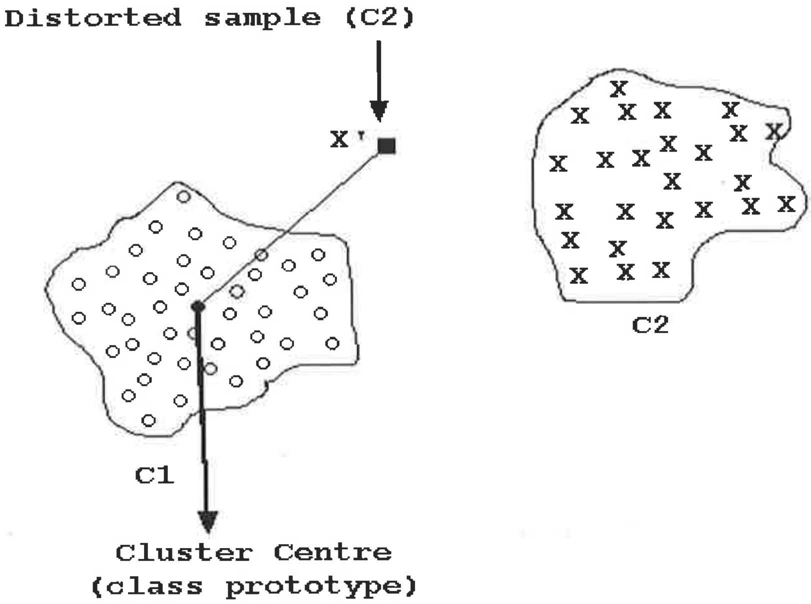


Figure 4-21: Similarity of distorted character of class C2 to the patterns of class C1.

4.3.2 Similarity and The Deformation Model

To show the similarity between distorted versions of some Persian and Arabic characters, we use the deformation model presented before in this chapter. Printed samples of a character are deformed by different distortion parameters, and the correlation coefficient between the distorted pattern and the prototype of another character (printed sample) is used as the similarity measure. We present examples of the correlation coefficient for rotation (Fig. 4-22), horizontal slant (Fig. 4-23), and horizontal perspective (Fig. 4-24) distortions. The characters in the examples are selected from similar and confusing characters. In all these examples, the correlation coefficient between the distorted pattern of a character and prototype the pattern of another character increases for a non-zero value of the distortion parameter.

4.4 Approaches to Handwriting Style Variation

Researchers in the field of handwriting recognition have been trying to minimize the effect of variability by employing two main strategies: standardization of the raw data by normalization, and particularization of the problem by limiting the number

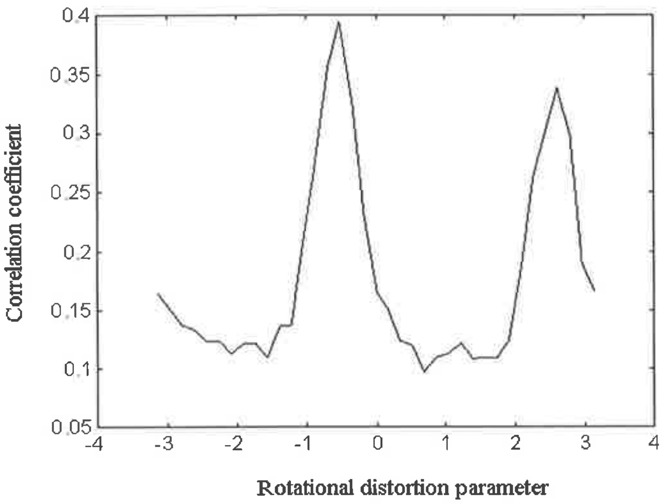


Figure 4-22: Correlation coefficients between the character "(beginning) ALEF" deformed with different rotational distortion parameters, and the character "RA".

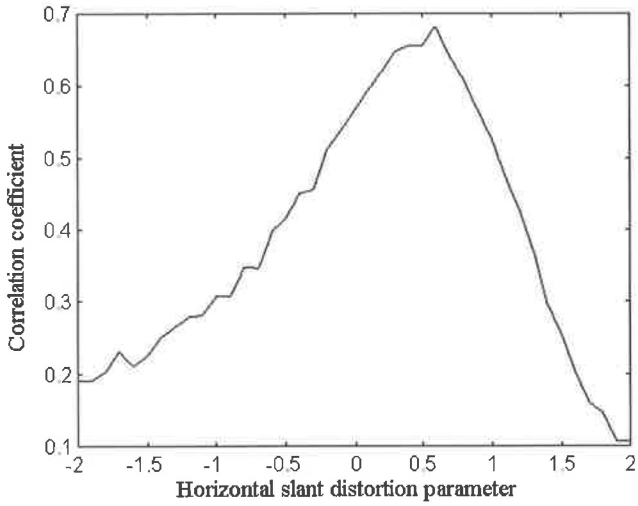


Figure 4-23: Correlation coefficients between the character "DAL" deformed with different horizontal slant distortion parameters, and the character "RA".

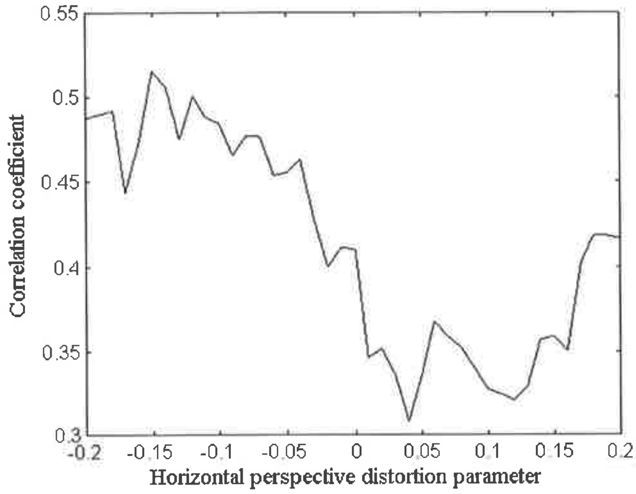


Figure 4-24: Correlation coefficients between the character "DAL" deformed with different horizontal perspective distortion parameters, and the character "RA".

of objects to be recognized [30]. In many cases, patterns are normalized in the preprocessing stage. Normalization attempts to remove random irrelevant variations from the characters while preserving the differences between patterns of different classes. Normalization of handwriting patterns may include deslanting (or deskewing), base-line drift correction, and normalization of size and component length. For example, Nagy and Tuong [87] have described a technique of normalization using perspective transformation. They found the four points where a string drawn tightly around the character passes through $\pm 45°$ to the horizontal, then, by using the coordinates of these points, they normalized the characters.

We humans have little difficulty in recognizing patterns irrespective of their size, position, deformation and orientation in our field of view. How can we get a computer to do this? One approach is to extract functions and features from the pattern that are invariant to the transformation made by these changes. The theory and practice of such invariant image features are presented in [95] for planar objects.

Rigid templates cannot account for deformations which frequently arise from diversity and irregularity of patterns. Since the degree of deformation is also unknown in advance, rigid templates for a range of deformation cannot produce satisfactory results for all cases. Deformable models are also an attractive way for characterizing handwritten patterns since they have relatively few parameters, are able to capture many topological variations, and incorporate much prior knowledge [96].

Another approach to the problem of variation in writing style and similarity of the patterns is the idea of using multiple expert systems. Combining multiple classifiers has the advantage that the features and the classification procedure of individual classifiers can be used to complement one another and improve the overall correct recognition rate. A multiple classifier system consists of a set of classifiers and a decision making unit which acts on the outputs of the individual classifiers. Each classifier uses a particular descriptor of an input pattern. The outputs of the individual classifiers are then combined to derive a final decision.

To read a piece of text, we humans use many more sources of information than just the image. Even a good feature extraction technique cannot distinguish between very similar patterns. To deal with similar patterns of the character, it is wise to use any contextual information. In this case, rather than trying to classify very similar characters in different classes, the system can put them in the same class. In the word recognition stage, however, a dictionary lookup system may be used to resolve the possible problem of similar characters.

4.5 Conclusions

In this chapter we studied two main problems of Persian and Arabic handwritten characters: variation of handwriting and similarity between characters. After discussing the components of variation in handwriting style, a model was presented to describe the various distortions by geometric transformations. We used a mixed mode distortion model and then used it to study the effect of each deformation on the individual characters.

If we consider that any sample of a handwritten character is a distorted version of a template, i.e., the corresponding printed character, then we can use the distortion model as an inverse transformation or warping function from the handwritten sample back towards the original template. This method represents the distortion characteristics of any handwritten sample by a set of parameters. This set of parameters are related to the writing style used by the writer.

It was shown that various sources of distortions have different effects on individual characters. This implies that different normalization procedures are needed for individual patterns. As shown in various examples, depending on the shape of the characters, individual sources of distortions affect the pattern of the character with a different range of parameters.

Distortion parameters of individual characters were estimated by a simple warping technique. By using this technique as a normalization process, the correlation between the handwritten samples of Persian and Arabic characters and their corresponding templates, in this case printed samples, significantly increased. Also, a new concept called *distortion characteristics*, was introduced to represent the impact of geometric transformations on real samples of handwritten characters.

The problem of pattern similarity was also presented in detail in this chapter. Some characters may not look very similar in normal shapes, but they could become similar when one or both of them are distorted. We use the distortion model to visualize this problem. Most of the missclassification is due to very similar characters.

Finally, the current techniques used to overcome the problem of handwriting variation and similar patterns were reviewed. They included either standardization of the raw data by normalization, or particularization of the problem by putting a limit on the number of objects to be recognized. Using invariant feature extraction techniques, which we will discuss in the next chapter, we can also increase the performance of recognition systems. We also introduced the combination of multiple classifiers as a possible solution to the problems of variability and similarity. Later in a following chapter we will discuss this issue in more detail. However, we should always keep in mind that for Persian and Arabic handwritten recognition, contextual information is still of great importance.

Chapter 5

Feature Extraction and Character Recognition

5.1 Introduction

The primary goal of this dissertation is to study potential problems of off-line recognition of Persian and Arabic handwritten characters. In the previous chapters, we presented two of the characteristics of Persian and Arabic characters, namely variability and similarity. In this chapter we study the problems of recognition of Persian and Arabic characters. Our main goal in this chapter is to use the same collected handwritten samples with different systems and evaluate their performances.

In the first section, the processes of data collection and preprocessing are explained. A new thinning algorithm for binary images of the characters is proposed in this section. In the following section, the feature extraction techniques used in this study are reviewed. These techniques can be divided into two groups: the first group includes feature extraction methods reported in the literature for Arabic or Persian character recognition systems; the second group are feature extraction techniques we proposed for either printed or handwritten character recognition, which include the modified ring projection transformation (MRPT), foveated retina log-polar mapping, and chain code histogram. The results of recognition systems for Persian and Arabic characters are then presented in two parts. In the first part the performance of selected invariant feature extraction techniques are tested for printed characters and in the second part the recognition performances of the studied feature extraction techniques are compared by using different classification schemes. The recognition performances of single classifiers were generally unsatisfactory for handwritten characters. By rejecting ambiguous patterns, the recognition rate of an individual classifier increases slightly, but the system reliability improves significantly. We also performed an experiment on the recognition ability of human experts of the preprocessed data. The results show that even human experts need to use more evidence, including contextual information, to perform a good recognition. This chapter is ended by some concluding remarks.

5.2 Data Acquisition

As we discussed in chapter 2, one of the problems in evaluating Persian and Arabic character recognition systems is the lack of standard data sets. Therefore, it is difficult to compare the results of the different systems implemented for Persian and Arabic texts. To collect the necessary data for our study we designed three forms as shown in appendix A. Forms were filled by a group of 54 Iranian students from three South Australian universities. The forms were then scanned and stored by using a digital scanner with a 300dpi resolution.

Preprocessing plays an important role in any pattern recognition system. Not only does it affect the shape of the resulting digital patterns, but also the features to be extracted afterwards. Two preprocessing steps may take place in a preprocessor: smoothing and normalization. Smoothing usually consists of filling or thinning a pattern. Smoothing algorithms are mostly based on some technique which slides a small window (e.g. 3×3) over the entire binary character matrix, and compares the state (1 or 0) of the central element with its neighbores to decide whether this state should be maintained or changed. In the processing of binary patterns, skeletonization or thinning consists of iterative deletions of the ON pixels (black pixels for binary images) along the edges of a pattern until the pattern is thinned to a one pixel width boundary drawing. More details on thinning algorithms can be found in [85].

We developed a thinning algorithm for use in our recognition systems. In this method, pixels which satisfy certain conditions are removed from the boundary of the character body. The main objective of the method is that the thinned image must not lose the structural information included in the original image. The boundary is traversed clockwise, and pixel removal is continued until there are no more pixels which can be deleted.

After finding one black pixel, the direction of the boundary at this point is calculated according to the codes shown in Fig. 5-1. In this figure, the darkest pixels are the ones under test for deletion, pixels marked as '×' are *"don't care"* pixels which do not matter if they are black or white, and the gray ones are black neighboring pixels. The black pixel under test will be removed if and only if there is another black pixel on the boundary or inside the body of the character by which the same direction can be represented, and its removal does not create a discontinuity. This is done by a set of logical equations used as conditions. In any case, the deletion of a pixel should not affect the curvature of the boundary. Figure 5-2 shows an example of a Persian character thinned by the proposed algorithm.

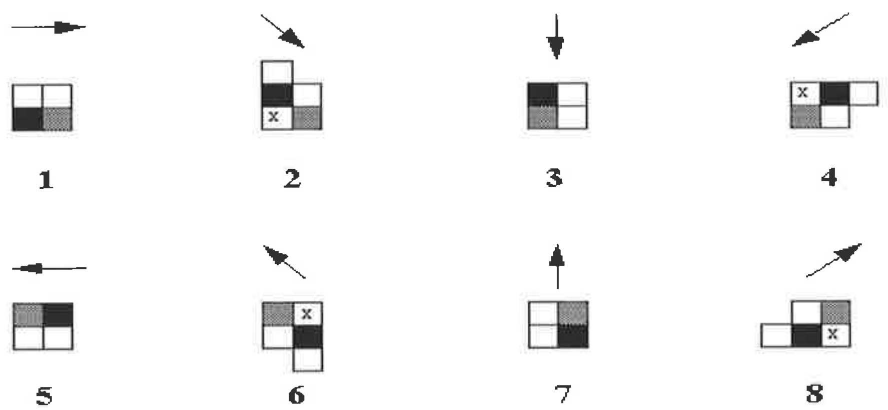
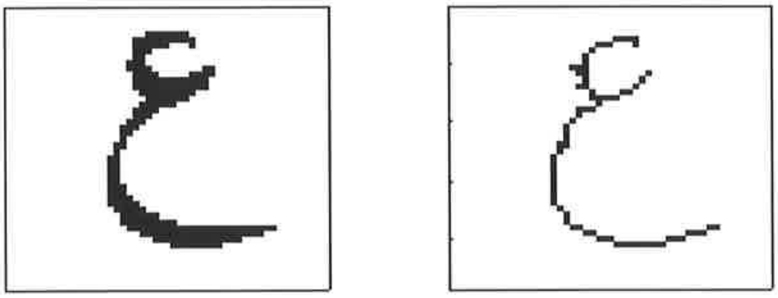


Figure 5-1: Direction codes used for the thinning algorithm.

The normalization stage, if necessary, includes size normalization, shifting the centroid of the character to the center of the matrix, removing gaps and isolated pixels by morphological filters, and boundary smoothing. For boundary smoothing,



(a) (b)

Figure 5-2: Images of a) character "(isolated) AIN", and b) thinned with the proposed algorithm.

a simple three point averaging operator was used. In this method, each point of the boundary, p_i, is replaced with

$$p'_i = \frac{p_{i-1} + p_i + p_{i+1}}{3} \tag{5.1}$$

where p_{i-1} and p_{i+1} are the two neighboring points of p_i. In Eq. 5.1, i is the index of the point on the boundary.

5.3 Art of Feature Selection

Pattern recognition, and character recognition in particular, has been attempted with many different systems and algorithms. Different recognition systems usually differ in their feature extraction units. The greatest difficulty in handprint recognition is the infinite number of possible shapes of characters generated by different writers. Nevertheless, we humans can still recognize most of them, partly due to experience and partly due to the existence of additional cues (features) to identify handwritten characters.

Suen et al.[111] divided the feature extraction techniques for character recognition into six feature families, which are derived from two main feature detection schemes: *global analysis* and *structural analysis*. Global analysis includes the distribution of points, transformation, and physical measurement feature families. Measuring the distribution of the pixels in the character image may include extracting the positional information, density, distance of certain elements from predetermined reference points, and crossings. Transformation features, on the other hand, may be derived by converting the image matrix into a series of numbers, a vector, or a spectrum. Structural analysis use the line representation of a character to derive the features, which may include certain line segments, edges, or the outline of a character.

In this section, we begin by briefly reviewing techniques for invariant pattern recognition. We describe two feature extraction systems based on the moment invariants whose applications for Arabic character recognition have already been reported in the literature ([8] and [35]). We then present two methods of invariant feature extraction that we have proposed and used successfully for the recognition of printed Persian and Arabic characters: the modified ring projection transformation (MRPT) and the foveated retinal log-polar mapping. Another feature extraction method, named *Chain Code Histogram*, which is based on the boundary information of the characters is proposed and introduced in this section.

5.3.1 Moment Invariant Features

We humans have little difficulty recognizing objects irrespective of their size, position and orientation in the field of view. This means that we may use features of objects, such as handwritten characters, which are invariant to these transformations. Extracting mathematical functions from an image that are invariant to size, orientation, position, and affine transform would thus provide us with a technique for recognizing objects using computers, as well as providing us with a possible model for part of human vision [95].

There are two main approaches to forming feature invariants: one involves normalization to a standard version of the image which is invariant to the transformations mentioned above; the other is finding invariant functions of the image.

Table 5.1: Various correlation and moment invariant functions of an image.

Correlation	T	$g_k(a_1,\ldots,a_{k-1}) = \int_{-\infty}^{+\infty} f(X)f(X+a_1)\ \ldots\ f(X+a_{k-1})dX$
	R	$g_k(a_1,\ldots,a_{k-1},r) = \int_0^{2\pi} f(r,\theta)f(r,\theta+a_1)\ldots f(r,\theta+a_{k-1})d\theta$
	S	$g_k(\ln a_1,\ldots,\ln a_{k-1}) = \int_0^{+\infty} f(X)f(a_1X)\ \ldots\ f(a_{k-1}X)\frac{dX}{X}$
Moments	T	$\mu_{pq} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}(x-\ \overline{x})^p(y-\ \overline{y})^q f(x,y)\ dx\ dy$
	R	$c_{pq} = \int_0^{2\pi}\int_0^{+\infty} r^{p+q+1}e^{j(p-q)\theta}f(r,\theta)\ dr\ d\theta$
	S	$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{(\frac{p+q}{2}+1)}}$

Invariance to image translation, rotation and changes of scale can be dealt with either using image correlation or using image moments.

Some invariant features selected from the correlation and moment families are presented in Table 5.1. In this table, the symbols "**T**", "**R**", and "**S**" stand for translation, rotation, and scale invariants, respectively. The image of the character is represented either by $f(x,y)$ or by $f(X)$, where $X^T = [x \quad y]$. The moment's order varies for different applications; however, higher order moments are very sensitive to noise and small changes in pixel position. $g_k(a_1,\ldots,a_{k-1})$ is the kth order correlation and $a_i^T = [\alpha_i \quad \beta_i]$ is a 2-D vector. μ_{pq} represents the central moment of order $(p+q)$.

From the moment based features, we selected two sets to be tested for handwritten character recognition; the applications of these two moment invariant sets for printed Arabic character recognition have already been reported in the literature. The first group of moment based features is the one used by Al-Yousefi and Udpa in[8]. For each character, they proposed a feature vector, f, with 9 elements calculated from the normalized moments of horizontal or vertical projection of the image as follows:

- Measures of kurtosis[76], which represents the flatness of the distribution:

$$\begin{cases} f(1) = \frac{\mu_4^V}{(\mu_2^v)^2} \\ f(2) = \frac{\mu_4^H}{(\mu_2^H)^2} \end{cases} \tag{5.2}$$

  where μ_k represents the kth central moments of the vector, and the superscripts V and H indicate vertical or horizontal projection.

- Measures of skewness, which represent the asymmetry of the distribution:

$$\begin{cases} f(3) = \frac{\mu_3^V}{(\mu_2^v)^{1.5}} \\ f(4) = \frac{\mu_3^H}{(\mu_2^H)^{1.5}} \end{cases} \tag{5.3}$$

- Measures of normalized skewness and kurtosis which show the symmetry to flatness of the distribution:

$$\begin{cases} f(5) = \frac{\mu_3^V}{(\mu_4^v)^{0.75}} \\ f(6) = \frac{\mu_3^H}{(\mu_4^H)^{0.75}} \end{cases} \tag{5.4}$$

- Ratios of vertical and horizontal moments:

$$\begin{cases} f(7) = \left|\frac{\mu_1^V}{\mu_1^H}\right| \\ f(8) = \frac{\mu_2^V}{\mu_2^H} \\ f(9) = \frac{\mu_4^V}{\mu_4^H} \end{cases} \tag{5.5}$$

The next group of moment invariants we analyze in this study are those used by El-Dabi et al. [35] for recognition of typed Arabic text. This group of moment invariants was first introduced by Hu [59]. The first four of Hu's translation, scale and rotation moment invariants are given by:

$$\begin{cases} \psi_1 = \eta_{20} + \eta_{02} \\ \psi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\ \psi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\ \psi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{12} + \eta_{03})^2 \end{cases} \tag{5.6}$$

where the η_{pq} is defined in Table 5.1.

5.3.2 Modified Ring Projection Transformation

In projections along straight lines, such as the horizontal or vertical line, the result of the projection varies with the orientation of the given pattern. The ring projection transformation (RPT) method was first introduced in [114], and is one of several rotation invariant techniques that have been used in pattern recognition. Before we start describing our modified ring projection transformation method, some definitions are in order [114].

Definition 1 *The ring-extraction panel is the triplet* $\Phi = (r, \theta, \delta)$, *where* $\theta \in [0, 2\pi]$, $r \in I$ *(*$I = \{0, 1, ...\}$*) and* δ, *the ring extraction function, is a function of* r *and* θ, *i.e.* $\delta = P(r, \theta)$ *where*

$$\forall_{k \in I}[P(r,\theta) = P(r, \theta + km)] \tag{5.7}$$

where m is the number of spokes. A graphical representation of the ring-extraction panel is shown in Fig. 5-3. In this figure, there are n concentric circles and m spokes. Each cross between a ring of radius r_i and a spoke S_j is called a sample point $P(i, j)$.

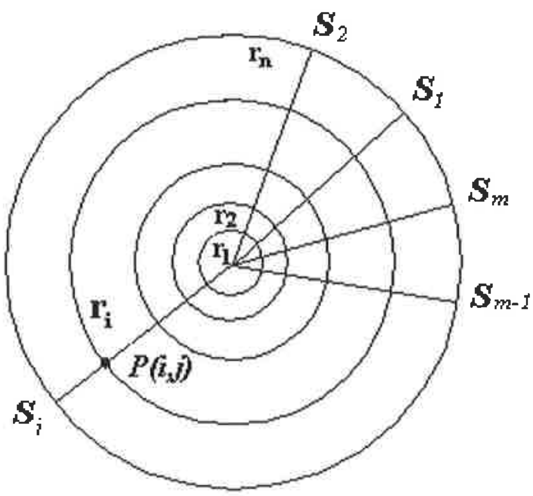


Figure 5-3: Ring extraction panel [114].

Definition 2 *A ring-projection vector* V *extracted by the ring-extraction panel* Φ *is given by*

$$V = [V_{r1} \quad V_{r2} \quad \dots \quad V_{rn}]^T \tag{5.8}$$

where n is the number of rings, and for discrete image patterns, V_{ri} is calculated as

$$V_{ri} = \sum_{j=1}^{m} P(i, j) \tag{5.9}$$

where m is the number of spokes.

Definition 3 *A ring r_i is called a zero ring if $V_{r_i} = 0$, otherwise it is called a non-zero ring.*

The ring-projection vector is sensitive to the round-off errors caused by the discrete nature of a binary pattern. In order to reduce the error caused by the shift in the centroid, Tang et al. used an accumulation operation [114]. Instead of using the feature vector defined in Eq. (5.8), they used

$$V' = \begin{bmatrix} V_{r1} \\ V_{r1} + V_{r2} \\ V_{r1} + V_{r2} + V_{r3} \\ \vdots \\ V_{r1} + V_{r2} + \cdots + V_{rn} \end{bmatrix} \tag{5.10}$$

To increase the stability of the feature vectors due to small changes in the centroid, we present the Modified RPT (MRPT) technique. In the modified version of ring projection, rather than adding the values of pixels at the sample points (Eq. 5.9), we calculate the area of black pixels in each ring. A pixel is assumed to have a square shape; therefore, only a fraction of it belongs to a ring with radius r_i , the remainder of the pixel belongs to the ring with radius r_{i+1} (see Fig. 5-4).

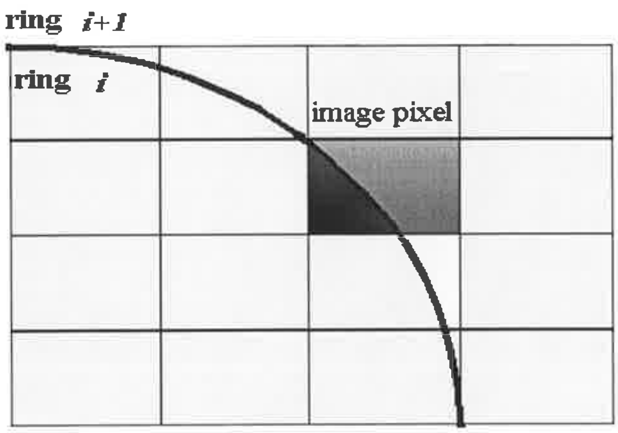


Figure 5-4: Rectangular black pixel which is located between two consecutive rings.

While the MRPT method is rotation invariant, it is neither scale nor translation invariant. The problem of translation can be solved by moving the character to a reference point in the matrix. The centroid of the character or the centre of gravity is the best candidate for a reference point to be moved to the centre of the matrix. The coordinates of the centroid of a pattern are given by

$$(x_c, y_c) = \left(\frac{m_{10}}{m_{00}} \quad , \quad \frac{m_{01}}{m_{00}}\right) \tag{5.11}$$

where m_{pq} represents the moments of the pattern of order $p + q$. In order to have a size-independent feature extraction technique, the size of the character should be normalized prior to the ring-projection operation.

5.3.3 Foveated Retina

The first studies carried out by Whitteridge and Daniel [125] on visual systems pointed out that the retinal topology can be optimally described in terms of ρ (radius) and θ (orientation). Research on the biological visual system and the anatomy of the eye has revealed that the photoreceptors are not uniformly distributed over the entire retina. The density of the receptors peaks in the centre of the visual field and decreases towards the *periphery*. The small central area, which is called the *fovea*, can resolve line widths at least equal to the inter-receptor spacing. To have this high resolution, an area of constant density spaced receptors is required. Outside the fovea, in the region known as the periphery, the density of photoreceptors decreases as a function of the radial distance from the center of the retina [107] (see Fig. 5-5).

Based on psychophysical experiments, researchers have characterized the image transformation performed by the visual pathway in mathematical terms. Studies mainly by Schwartz[102], yielded the well known analytical formulation of mapping that occurs between the retina (ρ, θ) and the visual cortex (η, γ). (ρ, θ) represent the polar coordinates of a point in the image and (η, γ) represent the corresponding point in the transformed space. This nonlinear image transformation is known as *log-polar* or *foveated* mapping.

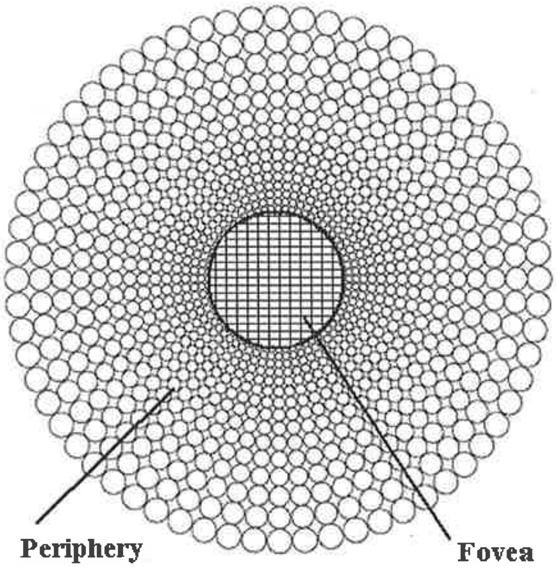


Figure 5-5: Distribution of the photo receptors in a foveated retina.

$$\gamma = \ln \frac{\rho}{\rho_0} \quad , \quad \eta = q\theta \tag{5.12}$$

where ρ_0 corresponds to the radius of the innermost circle of the log-polar layout; ρ and θ are given by

$$\rho = \sqrt{x^2 + y^2} \qquad \text{and} \qquad \theta = \arctan(y/x) \tag{5.13}$$

For a discrete image

$$\begin{aligned} \gamma_i &= \ln \frac{\rho_i}{\rho_0} \qquad i \in [1, \ldots, N_{circ}] \\ \eta_i &= q\theta_j \qquad j \in [1, \ldots, N_{ang}] \end{aligned} \tag{5.14}$$

ρ_0 is obtained on the basis of the dimensions of the smallest receptive field and the desired number of cells N_{ang} for each circle; N_{circ} is the number of cells for each radius. $(1/q)$ is the minimum angular resolution and is calculated as

$$\frac{1}{q} = \frac{2\pi}{N_{ang}} \tag{5.15}$$

Lemma *In a foveated mapping, an object keeps its perceived shape regardless of rotation or scaling.*

Proof. If we represent a point (z) in the image by its polar coordinates (ρ, θ), then after mapping the retinal plane by the log-polar transformation the corresponding transformed point (w) will have coordinates (γ, η) where

$$\begin{cases} \gamma = \ln \frac{\rho}{\rho_0} \\ \\ \eta = q\theta \end{cases} \tag{5.16}$$

If one scales the object by a scale factor k, and rotates the object by an angle φ, then a point of the image (z) will be transformed to a new point (z') with new coordinates (ρ', θ') as

$$\begin{cases} \rho' = k\rho \\ \\ \theta' = \theta + \varphi \end{cases} \tag{5.17}$$

the log-polar transformed image is

$$\begin{cases} \gamma' = \ln(\frac{\rho'}{\rho_0}) = \ln(\frac{k\rho}{\rho_0}) \\ \quad = \ln(\frac{\rho}{\rho_0}) + \ln(\frac{k}{\rho_0}) = \gamma + C_1 \\ \\ \eta' = q\theta' = q(\theta + \varphi) \\ \quad = q\theta + q\varphi = \eta + C_2 \end{cases} \tag{5.18}$$

This means that every change in scale or orientation of the original image is represented by a shift in log-polar coordinates.

In our proposed system, first the character is thinned and centered by moving its centroid to the center of the matrix. The image is then mapped into the logarithmic space according to the following transformation,

$$
\begin{cases}
\gamma_i = 10 \ln \rho_i & i \in [1, \ldots, N_{circ}] \\
\\
\eta_j = 10\theta_j & j \in [1, \ldots, N_{ang}]
\end{cases}
\tag{5.19}
$$

where $\rho_0 = 1$, and a coefficient 10 is used for scaling of both axes. For a black pixel in the original image, we first calculate θ_1, θ_2, ρ_1, and ρ_2 (Fig. 5-6). The area in the region between θ_1 and θ_2, and ρ_1 and ρ_2 is then transformed into the log-polar transformation according to Eq. (5.19); that is, any black pixel in the original image is transformed to a rectangular area (γ_i, η_i) (see Fig. 5-6) defined by:

$$
\begin{cases}
10 \ln \rho_1 < \gamma_i < 10 \ln \rho_2 \\
\\
10\theta_1 < \eta_i < 10\theta_2
\end{cases}
\tag{5.20}
$$

This will decrease the effects of the discrete nature of the character matrix. In our approach, we have used a 2×2 fovea at the center of the matrix.

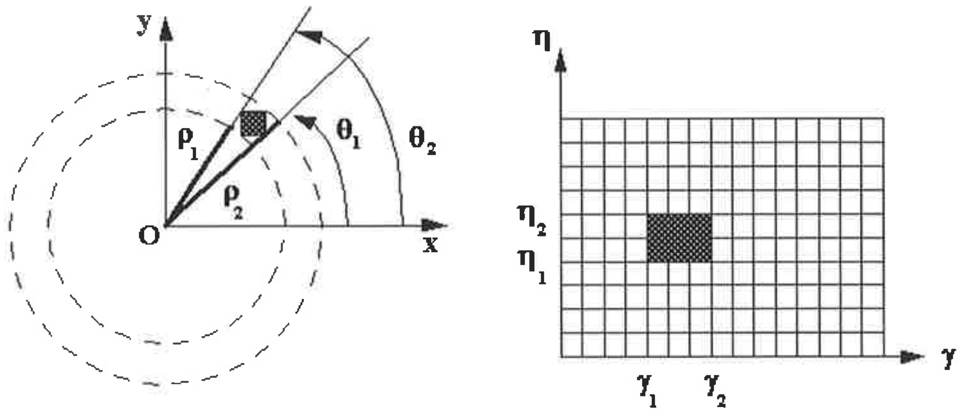


Figure 5-6: Proposed log-polar mapping of the character image.

Fig. 5-7 shows three rotated and scaled versions of the character "(isolated) AIN" and their corresponding images in the logarithmic space. As shown in this figure, scaling and rotation are converted to a shift of the mapped image in the log-polar space.

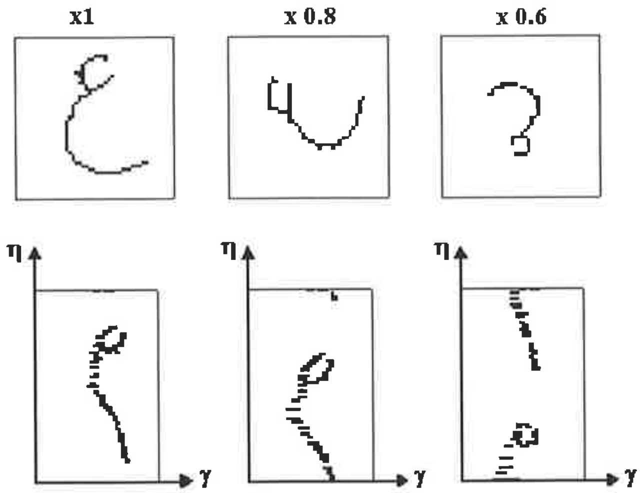


Figure 5-7: Three rotated and scaled versions of the character "(isolated) AIN" and their corresponding log-polar images.

If the image in the logarithmic space is projected onto the η axis, the resulting feature will be scale invariant, and if it is projected onto the γ axis, the feature vector will be rotation invariant. If the original pattern is normalized against scale change, then projection of the corresponding transformed image onto the γ axis will be both scale and rotation invariant.

Another possible way to get both scale and rotation invariant features is to project the image in the logarithmic space onto one axis, then, to resolve the shift on the other axis, and move the resulting features to a reference point. In the proposed system, since the image of the characters are normalized to a standard size prior to feature extraction, we only use the projection onto the γ axis as the feature vector.

5.3.4 Chain-Code Histogram

As a structural feature extraction technique, this method uses the information about the boundary of the character. The idea is very similar to the Freeman chain coding of boundary [42]. Depending on the status of its neighboring pixels, each black pixel in the image may have a code representing one of the 8 Freeman direction codes (see

Fig. 5-8a).

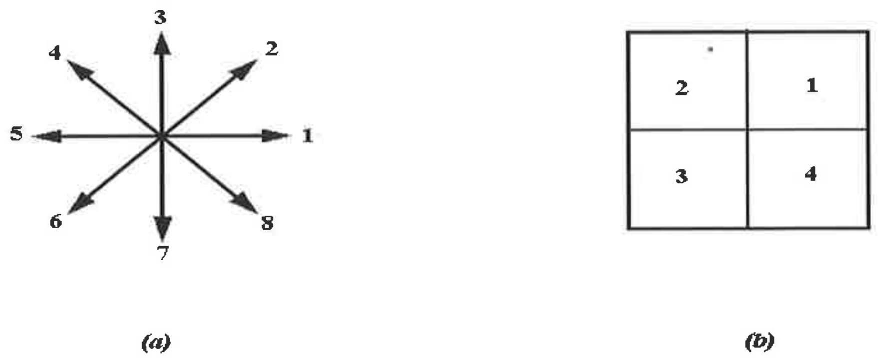


Figure 5-8: a) Freeman codes used in the proposed system b) quadrants of the character matrix.

After thinning the character, depending on the state of the neighboring pixels a code is assigned to each black pixel of the image. After this process, all the black pixels ("ON" pixels) of the image are replaced by their corresponding chain codes. Fig. 5-9 shows an example of the chain code histogram for the character "(isolated) ALEF". The structure of the character is well represented by this feature vector, as in this figure the character is mainly built by straight vertical line segments, i.e., chain codes 3 and 7 in Fig. 5-8a.

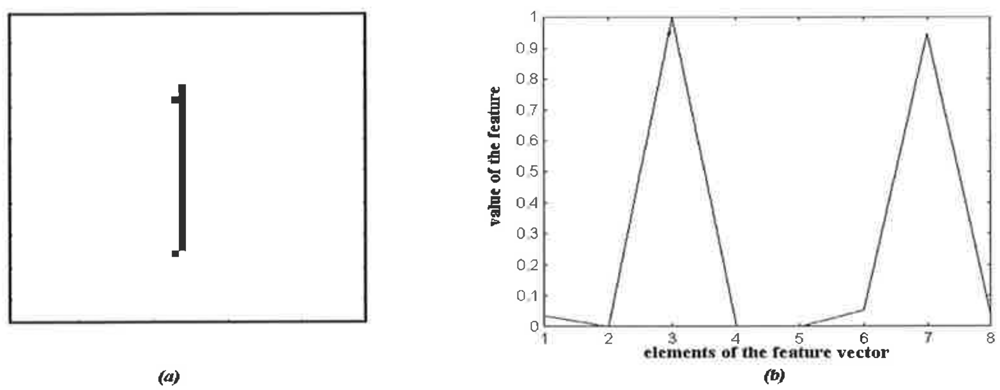


Figure 5-9: (a) Thinned image of character "(isolated) ALEF", and (b) normalized chain code histogram for the character without dividing into quadrants.

In the proposed system, the character matrix is divided into four quadrants as shown in Fig. 5-8b. For each quadrant, a histogram of the pixel codes is used as a feature vector; i.e., the feature vector of each quadrant has a length of 8, each element representing the total number of each code in the quadrant. The final 32 element feature vector is a concatenation of four vectors of the regions 1 to 4 of the image.

5.4 Experiments on Single Classifier Systems

In this section. we present the results of the proposed recognition systems on Persian and Arabic characters. Our aim is to evaluate the performances of the different features in a single classifier scheme. First, the different classifier types we used in our study are reviewed, then the results of the classification for both the printed and handwritten characters by the individual classifier systems are discussed.

5.4.1 Classifier Design

In this section we review the different classification techniques we used for our study. Each individual classifier is represented by a bank of discriminant functions, one for each class. An unknown input pattern is then assigned the class associated with the discriminant function of highest value.

Euclidean Minimum Distance classifier (EMD): This is one of the simplest classifiers. Its discriminant function is of the form

$$D_i(x) = -d^2(x, \mu_i) \tag{5.21}$$

where μ_i is the mean feature vector, or centroid, of the ith class. and x is the input feature vector to the classifier. $d^2(x, \mu_i)$ represents the Euclidean distance between the input feature x and μ_i. A feature vector x is assigned the class i if $D_i(x)$ has the highest value of discrimination among all the classes.

Quadratic Minimum Distance classifier (QMD): In this classifier the training set is used to produce sample covariance matrices (S_i) and to estimate the mean feature vector (μ_i). The discriminant function is then defined as:

$$D_i(x) = -(x - \mu_i)^T \; S_i^{-1} \; (x - \mu_i) \tag{5.22}$$

For a discriminant function to exist, the inverse of the covariance matrix must exist. This implies that the rank of the covariance matrix should not be less than n, where n is the dimensionality of the feature vectors.

K- Nearest Neighbor classifier (KNN): For $k = 1$, the class of the unknown input in this classifier is simply the class of the nearest training sample. The discriminant function for $k = 1$ is defined as:

$$D_i(x) = - \min_{1 \leq \; j \; \leq \; M_i} \; d^2(x, x_j^{(i)}) \tag{5.23}$$

where M_i is the number of training samples for the ith class, and $x_j^{(i)}$ is the jth training sample of class i. For $k > 1$, the class of an unknown pattern is assigned by voting on the classes of the k closest prototypes.

Multi-Layer Perceptron classifier (MLP): In this classifier, which is also known as feed-forward neural network, the training set is used for adjusting the weights of the inputs of each neuron. This process is known as learning and there are many different learning rules for this type of neural network. Backpropagation is the most commonly used learning rule [97]. For a one hidden layer MLP the discriminant functions can be define as:

$$D_i(x) = f\left(B2_i + \sum_{j=1}^{N_2} W2_{ij} \; f \left(B1_i + \sum_{k=1}^{N_1} W1_{jk} \; x_k \right)\right) \tag{5.24}$$

where $f(x)$ is the transfer function of the neurons, Wk_{ij} is the connection weight between the ith node of kth layer and the jth neuron of $(k-1)$th layer, and Bk_i is the bias weight of the ith neuron of kth layer. N_k is the number of neurons in the kth layer. The parameters Wk_{ij}, and Bk_i are adjusted by the training procedure.

Probabilistic Neural Network classifier (PNN): In this classifier, which was proposed by Specht [106], each training sample is considered as a center of a kernel function which has a maximum at the sample point and reduces gradually as one moves away from the sample point in the feature space. For each unknown input pattern x and for each class i, the sum of the values of the class kernels at x is computed. There are a number of possible kernel functions, however, radially symmetric Gaussian is the most commonly used kernel function. The resulting discriminant function for a probabilistic neural network classifier is:

$$D_i(x) = \sum_{j=1}^{M_i} \exp\left(-\frac{1}{2\sigma^2}\, d^2(x, x_j^{(i)})\right) \tag{5.25}$$

where σ is a scalar called *"smoothing parameter"*, which is usually optimized by trial and error. $d^2(x, x_j^{(i)})$ represents the Euclidean distance between the input feature vector x and feature vector of the jth training sample of class i ($x_j^{(i)}$).

Before testing the above mentioned feature extraction techniques for handwritten samples of Persian and Arabic characters, we test some of them for printed characters. We want to demonstrate that despite the fact that there are many feature extraction techniques proposed for printed Persian and Arabic characters, which performed successfully in recognizing machine printed documents, these techniques have very poor performance for handwritten characters.

5.4.2 Results on Printed Characters

Here we test the performance of the two invariant feature extraction techniques we proposed for the recognition of printed Persian and Arabic characters. The first method, the MRPT technique is tested for invariance against rotation distortion, and the second, the retinal log-polar transformation technique, is tested for both scale and rotation invariance.

Modified Ring Projection Transformation

For a set of patterns consisting of 58 different isolated Persian and Arabic printed characters, we use MRPT as the feature extraction technique. The characters, which are presented in 48×48 binary matrices, are first normalized; i.e., their centroids

Table 5.2: Recognition rates of MRPT method for roatetd printed Persian characters.

	Test range			
Training range	$[-10^\circ, 10^\circ]$	$[-15^\circ, 15^\circ]$	$[-30^\circ, 30^\circ]$	$[0^\circ, 360^\circ]$
$[-10^\circ, 10^\circ]$	96%	94%	89%	87%
$[-15^\circ, 15^\circ]$	96%	94%	90%	89%
$[-30^\circ, 30^\circ]$	95%	94%	91%	90%
$[0^\circ, 360^\circ]$	95%	93%	92%	91%

are transferred to the center of the matrix and then their sizes are normalized to a standard size. We used a multilayer Perceptron with one hidden layer as a classifier. This classifier was then trained by the feature vectors obtained from 10 rotated samples of each character, and 100 randomly rotated versions of each character as test patterns. The results of this experiment is shown in Table 5.2. This table shows the recognition rate for different ranges of rotation for both training and testing sets.

Most misclassification errors are caused by similar characters. For example the character "(beginning) MEEM" has the same shape as the character "(end) HA" when it is rotated.

Foveated Retina

The foveated mapping feature extraction was tested by samples of 10 selected printed Persian and Arabic characters. The characters were selected so as to include different shapes. Each character and its scaled versions (0.5, 0.75 scale factors) were randomly rotated to create 100 samples for each scale. Thus, for each character 300 samples with different rotations and scales were created. After log-polar mapping, the resulting image in the logarithmic space was projected onto the γ axis. To create scale invariant features, the centroids of the resulting vectors were circularly shifted to the middle of the feature vector. The feature vectors of 100 samples (10 for each character) were used for training of a MLP, and the remaining samples were used as a test set. The network consisted of (40, 10, 10) nodes in the input, hidden, and output layers, respectively. A correct classification rate of 97% was achieved for the test set.

Table 5.3: Recognition rates of different systems for handwritten characters.

	Classifier					
Sys. Id.	**EMD**	**QMD**	**KNN(1)**	**KNN(3)**	**KNN(5)**	**PNN**
PM	22%	**50%**	34%	34%	37%	33%
PR	73%	NA	75%	73%	**76%**	69%
RT	48%	NA	48%	48%	**53%**	50%
NM	21%	**63%**	35%	30%	31%	34%
SF	79%	NA	79%	78%	80%	**83%**
MI	39%	**52%**	46%	46%	48%	46%
CC	**63%**	NA	59%	62%	62%	59%
LP	**43%**	NA	40%	41%	42%	40%
LC	78%	NA	79%	**80%**	**80%**	79%

5.4.3 Results of Tests on Handwritten Characters

Now we present the performances of different systems for the recognition of handwritten Persian and Arabic characters. For all the following systems we used the same training and test sets. We divided the entire samples into two groups: *training set,* which consists of samples of selected handwritten characters of 25 persons, and the *test set,* which consists of the samples of the remaining 27 persons.

Recognition Without Rejection

Table 5.3 shows the results of different recognition systems we used for handwritten characters. Features for the systems are extracted from V&H projection moments (**PM**), V&H projection (**PR**), modified ring projection transformation (**RT**), normalized moments (**NM**), shadow features (**SF**), moment invariants (**MI**), chain code histogram (**CC**), log-polar mapping (**LP**), and line crossings (**LC**), respectively. This table shows in boldface the best results achieved for each individual system. The number in parenthesis for the KNN classifiers shows the value for k. Because for some systems, the inverse of the covariance matrix does not exist the QMD classifier is not applicable (shown by "NA" in Table 5.3).

Observations:

- The features which have a good recognition results on printed characters, may not necessarily have the same performances on handwritten patterns.

- Those features based on the pixel distribution, e.g. MRPT, proved to be less effective than the features based on the structure of the patterns, e.g. line crossings. The reason is that the former features are not invariant to deformation of the patterns.

- Rotation and scale invariances of features, e.g. retina model, could not help to overcome the other sources of pattern deformation. This means that the other sources of variations are more prominent than rotation or scale.

- Almost all moment based features have a poor performance. The reason is that moments, especially higher order moments, are very sensitive to pixel distribution.

- Topological features, e.g. shadow features, resulted in a higher recognition accuracy than moments and projection based features.

- Features built by dividing the image into quadrants and combining the individual features of the quadrants show better results than features derived from the whole character image; however, they will increase the computation time, and hence the complexity of the recognition system.

- Most of time, the preprocessed patterns, e.g. by thinning, show a better performance than the raw image.

- Different classifiers show different performances on individual features; for **RT** and **LP** systems which showed good performances on the printed characters, perform relatively poor on handwritten character recognition.

- Although it was claimed that some of the techniques reported in the literature achieved high recognition rates, they performed poorly on the handwritten data set we collected. This could be due to many factors such as environments, constraints, and fine tuning of the system which are not generally described in the published works. Here we tried not to impose any constraints on the writers.

Rejection of Patterns

The results in Table 5.3 present the recognition rates of systems without any rejection. In real-world, however, not all the samples are acceptable and there should be a mechanism for rejecting them. By rejecting ambiguous patterns the reliability of the system increases. The reliability is defined as

$$Reliability = 1 - \frac{E}{100\% - R} \tag{5.26}$$

where R is the rejection rate, and E is the error rate of the classifier.

There are many approaches to rejection including the two following methods [105]:

1. **Acceptance on Cleanness of Output:** Let y_k be the label corresponding to the maximum output of the classifier, i.e., $y_k = \max_i(y_i)$. The label y_k is accepted iff

$$\{y_k > clean\_top\} \ \wedge \{\min\{|y_k - y_i|\} < clean\_bottom \quad \forall i \neq k\} \tag{5.27}$$

   where *clean_top* and *clean_bottom* are two threshold values which are set to force the output to have a form of a clean target class.

2. **Rejection on Dirtiness of Output:** the label y_k with maximum output is rejected if

$$y_k < dirty\_bottom \tag{5.28}$$

   where *dirty_bottom* is a threshold representing the dirtiness of the output.

The above mentioned methods are not always practically applicable, and their efficiencies depend on the output levels of the classifiers. For example a QMD classifier may produce an output vector in which the labels have very close output values. The threshold values are usually determined by trial and error to get the best reliability for the system.

As we discussed earlier in this thesis, one of the most important problems of Persian and Arabic handwritten character recognition is the similarity between pat-

Table 5.4: Recognition Rate of the proposed systems after rejection of ambiguous patterns.

System	Thr.	Recognition	Rejection	Reliability	Reliability (No Rejection)
PM	0.002	56%	38%	0.29	0.50
PR	0.05	80%	10%	0.78	0.76
RT	0.01	60%	20%	0.5	0.53
NM	0.005	68%	28%	0.56	0.63
SF	0.25	88%	11%	0.87	0.83
MI	0.001	58%	29%	0.41	0.52
CC	0.03	72%	30%	0.6	0.63
LP	0.02	57%	34%	0.35	0.43
LC	0.3	88%	21%	0.85	0.80

terns. This implies that for similar patterns the outputs are very close. Because we use different types of classifiers, and for similar patterns they may produce small outputs, we reject a pattern based only on the closeness of the two top outputs of the classifier. This means that from Eq. 5.27 we are only using the second condition. A pattern is rejected if

$$\min_i |y_k - y_i| < closeness\_threshold \tag{5.29}$$

where the *closeness_threshold* is determined for each classifier by trial and error to give the best recognition rate with the least rejection rate. The above equation shows if the two top outputs are closer than a threshold, then the classifier may confuse between two class labels. The rejection criterion in Eq. (5.29) can also be combined with other rejection criteria like the one in Eq. (5.28).

Table 5.4 shows the experimental results for recognition and rejection rates and the reliability of different systems. The classifier systems in this table are the best ones from Table 5.3, based on their recognition rates. The recognition rates in the third column of the table show the rate of correct classification on the patterns which are not rejected. Because different classification systems have different output levels, the closeness threshold is different.

Observations:

- As shown in the table, in most of the cases a small threshold is needed to reject a large percentage of the patterns. This means that before applying the rejection mechanism, similar patterns could easily be confused by the classifiers, and a large portion of misclassifications may be caused by similar patterns.

- Although the recognition rates are improved, but in some systems, the reliability after rejection is less than the reliability without rejection. These systems have high rejection rates; hence, according to Eq. 5.26 high rejection rates will reduce the reliability.

- By rejecting ambiguous patterns, the total recognition rate of all the systems have increased.

- Comparing with the last column, which represents the reliability of the systems without rejection, the reliability of some systems are improved. Those systems with low recognition rate obviously have a lower reliability. For a system without rejection, the reliability is the same as recognition rate (i.e. $R = 0$ in Eq. 5.26).

- Without rejection, some of the rejected patterns might be correctly recognized by the classifiers but due to high rate of the closeness between the correct answer and the next close output, it may be considered as a random selection. In other words, the classifier does not have a high rate of discrimination between similar patterns and it may distinguish the correct answer by chance.

- Reliability can be used in applications where misclassifications are particularly injurious. For example, cheque reader systems should be very close to 100% reliable. Reliability can also be used as an evaluation measure for selecting the potential candidates for combining multiple classifiers.

5.4.4 Most Confusing Characters

Depending on the feature extraction technique, each classifier may confuse two or more character classes. For example, for a rotation invariant feature extraction method, it is most likely that patterns which are similar when rotated would be confused by the classifier.

In this section, we present pairs of confused characters for the proposed systems. These similar characters cause most of the errors of the recognition systems. Some of the confusions are obviously caused by similar shapes while some others are due to the preprocessing and normalization processes. Handwritten samples of some of the confused characters are shown in Fig. 5-10. The first two pairs of characters that are confused almost by all the systems are "HEH" and "AIN", and "SEEN" and "SAD". The character "SEEN" may also be confused with character "YA".

There are groups of characters which are not very similar, however, preprocessing and normalization make them more similar. For example, characters "YA" and "(isolated) KAF" are not very similar, but when size normalized they become more similar. The same problem happens for characters "LAM" and "NOON".

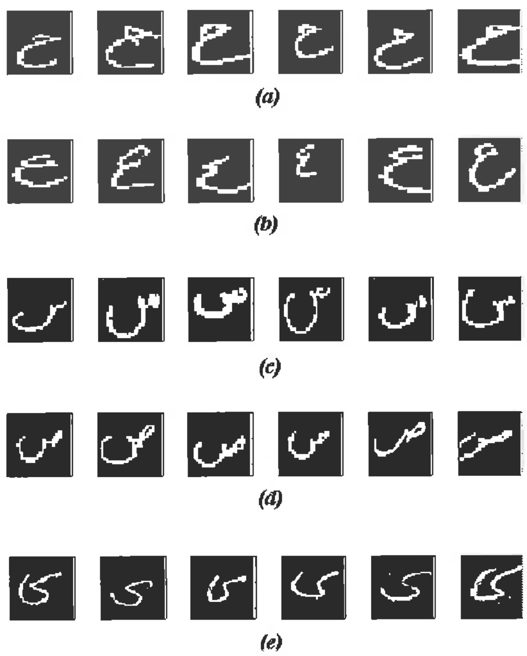


Figure 5-10: Handwritten samples of confused characters: a) "HEH" b) "AIN" c)"SEEN" d) "SAD" e) "YA".

Table 5.5: Recognition results of human experts on the data set.

Subject No.	Correctly classified	Rejected	Reliability
1	81%	3%	0.80
2	86%	3%	0.86
3	81%	1%	0.81
4	82%	10%	0.80
5	80%	6%	0.79
6	81%	5%	0.80
7	76%	9%	0.74
8	78%	26%	0.70
Average	**80.6%**	**7.8%**	**0.79**

5.5 Character Recognition By Human Experts

As we mentioned before, no standard data set exists for referencing and comparing different systems developed for Persian or Arabic characters. It is not even an easy task to evaluate the quality of the data. To evaluate our results for character recognition, we test the performances of human experts on the collected samples. Another reason for running this test is to show that even human expert readers, who use contextual information to increase the correct recognition rate, have problems recognizing characters without using much contextual information.

We presented a set of 400 randomly selected isolated Persian and Arabic characters to a group of Persian and Arabic speaking people. A sample form which was specially designed for this study is shown in Appendix A. All the patterns were selected after the preprocessing stage, e.g., thinning. Table 5.5 shows the recognition and rejection rates for 8 different subjects. The second column shows the correct recognition rate on the patterns which are accepted by the subjects.

Observations:

- Compared to the machine recognition results, human experts achieve a higher reliability with less rejection rates.

- The machine recognition system and human experts have similar confusion patterns. For example, samples of the characters "SEEN" and "SAD" or characters "HEH" and "AIN" are also confused by human experts (see Fig.

5-10).

- As the results show, without using contextual information even human experts have problems in recognizing unconstrained handwritten characters.

- The last two subjects were Arabic speakers and had problem with Persian characters like "(isolated) GAF", resulting in a lower recognition rate than the other subjects.

5.6 Conclusions

In this chapter we studied different methods for extracting features from both printed and handwritten samples of Persian and Arabic characters. The process of data collection and preprocessing was discussed, and a new technique for thinning characters was introduced. We then discussed the feature extraction techniques we used for our study. These techniques include both the feature extraction methods that were reported in the literatures, for Arabic or Persian character recognition systems, and those techniques we proposed ourselves. The new techniques included modified ring projection transformation (MRPT), the foveated retinal log-polar mapping, and the chain code histogram.

The performances of different classification systems have also been evaluated for both the recognition of the printed and handwritten samples. The results showed that features have different performances for printed and handwritten samples, and a good recognition rate on printed characters may not necessarily result in a good performance on handwritten samples. The reason, as we discussed in a previous chapter, is the high variation in the patterns of handwritten characters. Features based on pixel distribution proved to be less successful than features based on the structure of the characters.

In chapter 4, we discussed different sources of deformation that may happen to the handwritten samples of a character. Rotational and scaling distortions were part of the distortion model; however, experimental results showed that using only rotation and scale invariant features is not enough to resolve the variation problem of handwritten patterns. This implies that the other sources of deformation have more impact on the patterns than the rotational or scaling distortions.

We also studied different classification techniques. Different classifiers showed different performances on the individual features, however, even the best recognition rate of a single-classifier system is far from the results obtained for the printed characters. By choosing a good training set the recognition rates can be significantly improved for any classification system.

Some of the techniques reported in literature performed poorly on the handwritten data set we collected. This could be due to many parameters including writing constraints; here we tried not to impose any constraints on the writers.

Furthermore, we introduced a rejection criterion based on the closeness of the highest two outputs of a classifier. This is a measure of pattern similarity, which is a main problem in Persian and Arabic characters. By adding the rejection mechanism, the reliability of the classifiers increased; however, in many systems a large number of input patterns were rejected. Reliability can be used in applications where misclassifications have a very high cost, e.g. cheque reader systems.

We also derived a test on the recognition performances of human experts on the collected and preprocessed data. Human experts showed a high reliability. The interesting result is that the machine recognition systems made almost the same mistakes as human experts; they all showed a poor performance in distinguishing between similar patterns. In conclusion, we need more evidences, including contextual information, to achieve a good recognition rate.

Chapter 6

Multiple Classifiers Combiners

6.1 Introduction

Despite the success of handwritten character recognition in constrained domains, the problems in the application areas that involve recognition of distorted, and unconstrained data still remain unsolved. A range of recognition systems with high recognition rates have been reported; however, as we showed in the previous chapter, when dealing with a wide range of variations in handwriting styles, almost none of these systems could achieve a satisfactory performance. Due to inadequate training data, noise, and high variability in the data, most of the single classification strategies often perform significantly below the Bayesian error limits. There is a new trend in classification, namely *the combination of multiple Experts*, to improve the performance of handwritten recognition systems; see for example [63, 120, 92]. It is shown that even humans combine the independent features encoded in parallel with a special attentional mechanism to recognize patterns [121]. As a multiple classifier system allows for the simultaneous use of different feature descriptors and different decision boundaries of the classifiers, it is often the preferred solution to complex pattern recognition problems.

In this chapter, we study the application of multiple classifier combination systems for Persian and Arabic handwritten isolated characters. After a brief discussion of the history and background theory of combining multiple classification systems, three methods of combining multiple classifiers that we use for our study are described. The first combiner is the commonly used weighted voting method. We use the recognition rates of individual classifiers for each class on its training samples as the respective weight. This increases the probability of correct decision made by the multiple expert system. The second system is a linear committee combiner in which each individual classifier outputs a value for each class. A weighted sum of the classifier outputs is then formed using the same weight factors for the voting system. A third system, which we call *multi-label classifier combiner*, will be introduced. This system works by combining the ranked classifier outputs. For all three systems, we introduced a method for rejecting confusing input patterns. For each combined system we show by experiments that the combination of multiple classifiers always has a higher recognition rate than a single classifier. The chapter is then completed by some concluding remarks.

6.2 Background

In the field of handwritten character recognition, there has been an increasing interest in using combined classifiers to increase the performance [61, 113]. The combination of classifiers allows for the simultaneous use of feature descriptors of many types, corresponding measures of similarity, and many possible classification procedures. Combining multiple classifiers is based on the idea that different methodologies of classification can complement each other. One of the potential pitfalls of using a combination of multiple classifiers is the higher computational cost. However, by using parallel computing and processing techniques, this problem can be significantly reduced.

Finding the best classifier and the optimal selection of features for classification is not always possible beforehand. Concatenating different feature descriptors into a single vector is undesirable for many reasons, including:

- designing an accurate classifier for features with higher dimensions is more complex than for smaller vector sizes.

- larger size of the input vector complicates the training phase and parameter selection.

- by mixing qualitatively different features, the relative importance of the most discriminant features may change in the resulting combined feature vector.

Using a committee of decision makers is an old idea which was discussed as early as 1960's [89]. Theoretical analysis of the systems that use majority votes of n (odd integer) independent decision makers on a two-class classification problem was discussed by Srihari [109]. He showed that the recognition rate of the combined system increases monotonically with n for $p > 0.5$, and decreases monotonically for $p < 0.5$, where p is the recognition rate of the individual classifiers. Majority vote rule also was used by Azurov et al. [17] in a two-class decision problem.

The applications of combined classifiers for recognition of handwritten characters and numerals is significantly increasing. Kimura and Shridhar [70] described a combination of a statistical algorithm and a structural method for handwritten numeral recognition. The focus of their work was to reduce the error rate by rejecting the cases for which both methods disagree. Suen et. al. [113] proposed using multiple experts for the recognition of handwritten characters. A method of combining multiple experts for handwritten numeral recognition was also discussed by Huang and Suen [61]. Several decision combination methods based on ranking was outlined by Ho [58]. She proposed a theory of multiple classifier system, which uses the ranking of the class set to represent the decision made by an individual classifier. The method was then tested for visual word recognition with the combined method giving a significant increase in the recognition rate. Powalka et al. [93] also used a combination of multiple classifiers for handwritten word recognition. Combination of multiple classifiers has been used by Franke and Oberlander [41] to detect the writing style in a form reader system.

6.2.1 Combination Methods

As shown in Fig. 6-1, a multiple classifier system consists of a set of feature extractors and classifiers and a decision combination function

$$Z = f(c_1, c_2, \ldots, c_n) \tag{6.1}$$

where Z is a vector of length l (number of classes), c_1 to c_n are the outputs of n individual classifiers, and f is the combination function.

If each individual classifier is considered as a black box that receives input x and outputs a label $C(x)$, then depending on the classifier type $C(x)$ may be in one of the following three levels [128]:

1. **The abstract level**: the classifier only outputs a unique label, i.e., $C(x) = i$ where i represents the class to which the input pattern belongs.

2. **The rank level**: the classifier outputs a queue in which all the class labels are assigned a rank. The label at the top of the queue is the first choice.

3. **The measurement level**: the classifier attributes to each class a measurement value. This value represents the degree to which the input x belongs to each class.

Moving from the abstract level to the measurement level, the amount of information at the output of the classifier increases; the output of a measurement level classifier contains the highest information. An abstract level classifier can be easily built using any two other types of classifiers; for example, the class which has the maximum measurement value in a measurement level classifier can also be output as a unique output label.

Depending on the type of the individual feature extraction unit and its corresponding classifier, a combination decision unit can be selected from one of the following methods:

- Committee-like expert combination (voting).

- Conditional mixture of experts.
- Stacked generalization.
- Boosting.

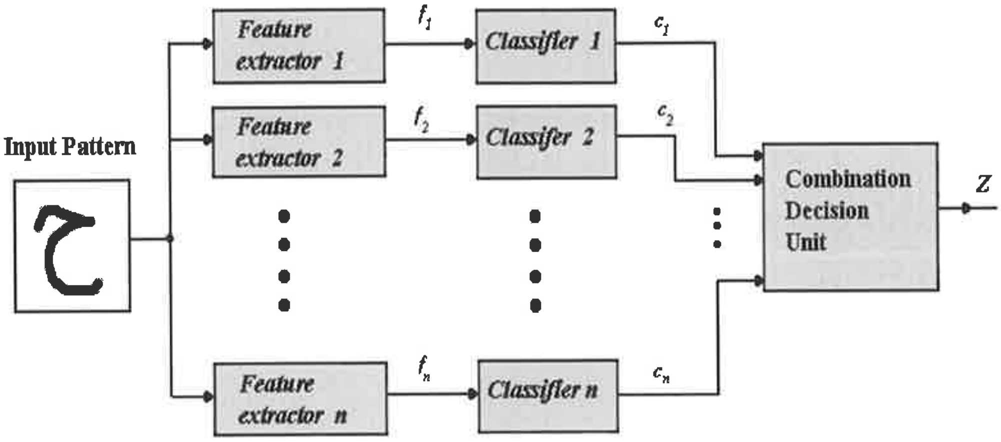


Figure 6-1: Block diagram of a typical system for combination of multiple classifiers.

Committee-like Combiners

In a committee-like classifier combiner [55], outputs of the individual classifiers are unconditionally combined. The output is a linear, weighted sum of the output of the individual classifiers. This is the simplest way of combining multiple classifiers with the output of the combined system expressed as

$$Z = \sum_{i=1}^{n} w_i \, c_i \tag{6.2}$$

where w_i is a weighting factor for the output of the ith classifier.

Conditional Mixture of Experts

In a conditional mixture of experts a gating scheme is used (Fig. 6-2). The gating network places a weight g_i on the output of each subsystem.

$$Z = f(c_1, c_2, \ldots, c_n, X) = \sum_{i=1}^{n} g_i \ c_i \tag{6.3}$$

where $g_i = G(X)$ is calculated by a measure of the input pattern X. The gating measure is calculated by detecting certain features from the input pattern.

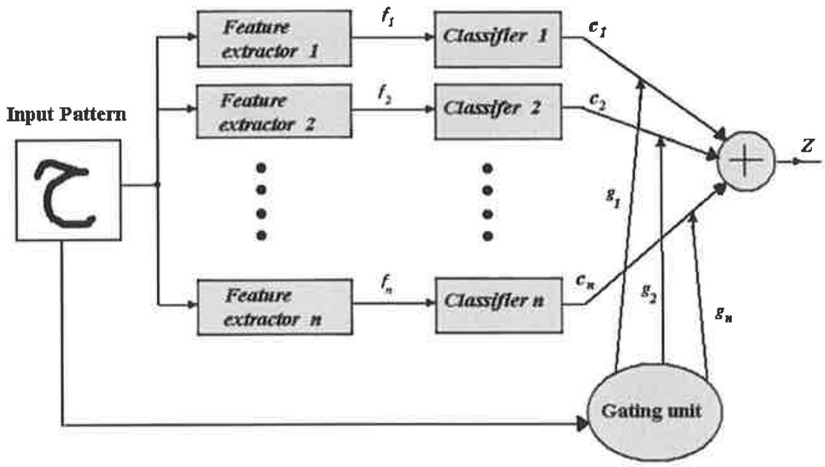


Figure 6-2: Block diagram of a typical conditional mixture of experts system.

Stacked Generalization

In stacked generalization, which was proposed by Wolpert [127], outputs of individual experts are treated as new features. As shown in Fig. 6-3, the combiner unit is a classifier itself and performs a pattern classification on these new features which are called level two features. The combiner classifier is trained to learn the correct output class using level two features from individual classes.

Boosting

Drucker et al. [34] introduced the boosting technique for constructing a classifier which makes small error rates from classifiers which are doing just slightly better than 50 percent recognition. The training process in this type of combiner is serial and after training the first classifier, the second one is trained with the data on which the first one failed, and the third one is trained with the data on which the first two

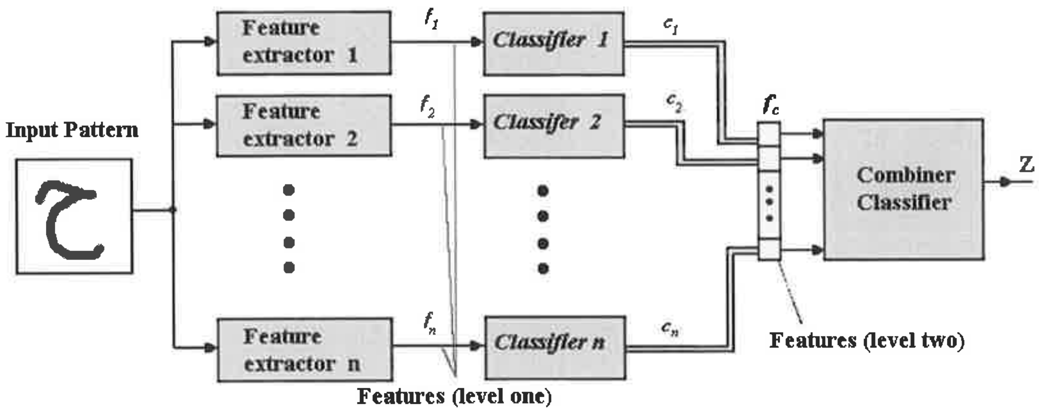


Figure 6-3: Block diagram of a typical stacked generalization combiner system.

classifiers disagree. This ensures that the classifiers complement each other. As shown in Fig. 6-4, the third classifier is consulted only when the first two classifiers disagree. As different training sets are used, the boosting method requires large training sets and it is more computationally expensive than the other previous three methods. In this combiner the final output r is defined as:

$$r = \begin{cases} c_1 & \text{if } c_1 = c_2 \\ c_3 & \text{otherwise} \end{cases} \tag{6.4}$$

6.3 Proposed Systems

Depending on the level of the outputs of the subsystems, there are different methods of combining the outputs. In this thesis, we study three different types of classifier combiner systems: *Weighted Voting Combiner (WVC)*, *Linear Committee Combiner (LCC)*, and the *Multi-label Classifier Combiner (MCC)*. In the first system, each classifier outputs a unique label as the output. The output label of each classifier is considered as a vote for the corresponding label. The votes are then linearly combined by weights which are calculated from a priori information obtained from the training set. In the second system (LCC), the classifiers present their outputs at the measurement level, which are then combined using the same technique as WVC. In the third method (MCC), each classifier outputs a queue in which the labels are

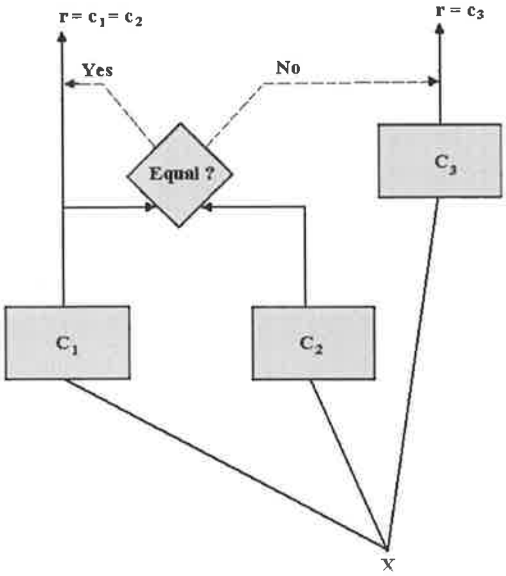


Figure 6-4: Block diagram of a combiner based on the boosting method (from [9]).

ordered according to their ranks. The rank level outputs are then combined in a decision making unit.

6.3.1 Weighted Voting Combiner

If the kth output of the ith classifier is denoted by c_i^k, then for an abstract level we have

$$c_i^k = \begin{cases} 1, & \text{if } \arg(\max(C_i(x))) = k \\ 0, & \text{otherwise.} \end{cases} \tag{6.5}$$

where $C_i(x)$ is the output vector of the ith classifier. A simple and common combining rule used is majority voting. For an equal voting weight, the voting function is defined as

$$V^k(x) = \sum_{i=1}^{n} c_i^k \tag{6.6}$$

where n is number of classifiers to be combined. For a majority rule voting combiner, the output class label of the combined system is given

$$Output\ Class = \arg(\max_k(V^k(x))). \quad (6.7)$$

Each classifier has a different recognition rate for different classes. This implies that using equal weights for each class, when voting, may produce suboptimal results. In other words, to achieve a better performance with the voting combiner, different weights should be assigned to the votes of the individual classifiers.

$$V^k(x) = \sum_{i=1}^{n} w_i^k c_i^k \quad (6.8)$$

where w_i^k is the weight of the ith classifier for class k. In our system, we use the recognition rates r_i^k of the individual classifiers for each class as the weight w_i^k. The voting function is then defined as

$$V^k(x) = \sum_{i=1}^{n} r_i^k c_i^k \quad (6.9)$$

where r_i^k is the recognition rate of the ith classifier for the kth class on the training set. In other words, recognition rates of the individual classifiers are used as a *priori* knowledge of the classifiers as experts. This ensures that each classifier has greater voting power for the classes in which it has a better recognition rate.

Class Recognition Rates We now present recognition rates of different systems for each class. As discussed earlier, we need this for our weighted voting combiner. Table 6.1 shows the class recognition rates of different classification systems on their test sets. In this table, **PM** to **LC** are single classifier systems studied in the previous chapter. To calculate the necessary weights, however, we use the training set.

Observations

- Even for a good classification system like **SF**, there are classes in which the recognition rate is not satisfactory. The reason is that there are confusions

Table 6.1: Inter-class recognition rates of the proposed systems on their test sets.

	Proposed Systems								
Char.	**PM**	**PR**	**RT**	**NM**	**SF**	**MI**	**CC**	**LP**	**LC**
ALEF	97%	100%	90%	69%	83%	48%	69%	43%	76%
BA	90%	97%	69%	66%	90%	66%	28%	62%	93%
HEH	72%	97%	55%	41%	76%	45%	45%	72%	93%
DAL	38%	79%	62%	90%	76%	69%	83%	79%	93%
RA	14%	83%	52%	86%	93%	66%	69%	85%	93%
SEEN	38%	45%	14%	59%	34%	34%	48%	34%	59%
SAD	24%	76%	55%	28%	72%	21%	45%	31%	62%
TTA	41%	76%	52%	55%	100%	55%	59%	45%	97%
AIN	55%	59%	45%	83%	90%	38%	86%	38%	76%
FA	79%	79%	41%	59%	97%	79%	69%	72%	93%
KAF	41%	76%	59%	38%	76%	79%	72%	45%	62%
LAM	59%	79%	48%	66%	97%	69%	79%	72%	93%
MEEM	62%	83%	28%	90%	97%	48%	100%	48%	100%
NOON	28%	59%	52%	83%	79%	66%	62%	72%	17%
WAW	24%	55%	31%	62%	100%	55%	45%	38%	76%
HA	34%	72%	62%	41%	83%	24%	48%	30%	93%
YA	48%	69%	52%	28%	66%	24%	62%	55%	90%

between two or more classes, e.g. character "SEEN" with character "SAD".

- For some classification systems which have an over all poor performance, there are particular classes for which they outperform the best classification system (for example see the performances of **PM** and **SF** systems on character "ALEF").

Experimental Results To show the performances of the combined systems, we built different combined systems by choosing different groups of the single classification methods introduced in the previous chapter. Table 6.2 shows the recognition rates of five combined systems. In the first systems (**WVC_1**), all the proposed systems discussed in the previous chapter are included while in the other combined systems only classifiers which have good performances are included.

Observations

- By using more classifiers for voting, the method becomes more computational expensive; i.e., each test pattern should be applied to all the classifiers, and

Table 6.2: Recognition rates of different systems combined by using weighted voting method (no rejection).

System Id.	Combined Systems	Recognition Rate
WVC-1	All classifiers	90%
WVC-2	SF, LC, PR,CC, NM	90%
WVC-3	SF, LC, PR,CC	88%
WVC-4	SF, LC, PR	87%
WVC-5	SF, LC	82%

then their outputs are combined.

- The more the number of good systems, the better the recognition rates; they should, however, make different mistakes. As indicated in Table 6.2, moving from **WVC_2** system which uses all the good systems to **WVC_5** which only uses the two best systems, the recognition rate decreases.

- In **WVC_5**, which only combines the two best classifiers, the recognition rate is less than the recognition rate of a single classifier **SF** system (83%). This means that the decision made by voting between two systems may result in a lower performance. This reduction in performance comes from those input patterns on which the two classifiers disagree.

Rejection Method For each classifier an extra output is added for rejection which is treated as a class label. The individual classifiers reject a pattern by a measure of closeness between the first two output labels as discussed in the previous chapter. The combined system have also an output label for rejection, and the same voting rule as Eq. (6.8) is applied. If the rejection output of the ith classifier is denoted by $c_{r,i}$ where

$$c_{r,i} = \begin{cases} 1, & \text{if the pattern is rejected by the classifier} \\ 0, & \text{otherwise.} \end{cases} \tag{6.10}$$

then the vote for the final rejection of the combined system is calculated as

$$V_r(x) = \sum_{i=1}^{n} w_i^r c_{r,i} \tag{6.11}$$

Table 6.3: Performance of the systems combined by weighted voting and by adding rejection mechanism.

System Id.	Recognition	Rejection	Reliability	No Rejection
WVC-1	96%	28%	0.94	90%
WVC-2	95%	26%	0.93	90%
WVC-3	94%	25%	0.92	88%
WVC-4	91%	23%	0.88	87%
WVC-5	88%	23%	0.84	82%

where w_i^r is a weight factor for the rejection vote of the ith classifier $w_i^r = 1 - \frac{E_{with\_rejection}}{E_{no\_rejection}}$ where $E_{no\_rejection}$ and $E_{with\_rejection}$ represent the error rates of the ith classifier before and after applying the rejection mechanism, respectively. In fact w_i^r is a measure of the correct rejection rate of the individual classifiers.

An input pattern is rejected by the combined system if the weighted vote of the rejection output is greater than the votes for all the class labels, i.e., $V_r(x) > V^k(x)$ for all k. Table 6.3 shows the results of rejection and recognition rates of the classifier systems in Table 6.2 after adding the rejection mechanism.

As the results show, by rejecting the confusing inputs both the recognition rate and reliability of all systems are increased. Increasing the number of combined classifiers slightly increases the recognition rate; however, a larger portion of the patterns are rejected. In practice, two issues should be considered when selecting classifiers to be combined.

1. A classifier is selected if there is no other similar classifier. Two classifiers are similar if they make the same mistakes on the training set.

2. A classifier is selected if it has a reasonable performance in terms of the recognition rate of each class. As shown in Table 6.2, adding **PR**, which is a classifier with a reasonable recognition rate, to **WVC-5** significantly increases the performance of the resulting system (**WVC-4**), while adding the poor performance classifiers like **RT** and **PM** to **WVC-2** does not change the recognition rate (see recognition rates of **WVC-1** and **WVC-2** before rejection). The reason is that the weights of poor classifiers are too small and they do not have big contribution in the final vote.

Table 6.4: Recognition rates of differnt systems combined by using linear committee combiner.

System Id.	Systems	Recognition Rate
LCC-1	All classifiers	91%
LCC-2	SF, LC, PR,CC, NM	90%
LCC-3	SF, LC, PR,CC	89%
LCC-4	SF, LC, PR	88%
LCC-5	SF, LC	84%

6.3.2 Linear Committee Combiner

As shown in Eq. (6.12), the majority voting rule can also be extended to measurement level classifiers in which d_i^k is normalized to the range $[0, 1]$. Voting combiner is a special case of the so called *linear committee classifiers* [60].

$$V^k(x) = \sum_{i=1}^{n} w_i^k d_i^k \tag{6.12}$$

where d_i^k is the kth output of the ith classifier. The output class label is calculated by the same rule as Eq. (6.7).

Experimental Results

Table 6.4 shows the correct recognition rates of the systems combined by the linear committee method by choosing different groups of classifiers. As shown in this table, the performance of this system is slightly better than the voting system. The reason is that there is more information in a measurement level output than an abstract level output.

Rejection Method

The rejection mechanism for the linear committee combiner is the same as the method introduced in the previous chapter for a single classifier. The rejection is based on the closeness of the two top most outputs of the combined classifiers. A pattern is rejected if the first two outputs of the combined system are closer than a threshold. If the first two outputs of the combined system are denoted by $V^{k1}(x)$

and $V^{k2}(x)$, respectively, then an input pattern is rejected if

$$V^{k1}(x) - V^{k2}(x) < closeness\_threshold \tag{6.13}$$

where the *closeness_threshold* is determined by a trial and error to give the best reliability of the system.

We run three tests on the combined system for three values of *closeness_threshold*. Tables 6.5, 6.6, and 6.7 show the performance of the different systems. Note that the bigger the threshold is, the more patterns are rejected.

Table 6.5: Recognition and rejection rates of the systems combined by using linear committee combiner (closeness threshold = 0.05).

System Id.	Recognition	Rejection	Reliability
LCC-0.05_R1	92%	4.5%	0.92
LCC-0.05_R2	90%	3%	0.9
LCC-0.05_R3	89%	3%	0.89
LCC-0.05_R4	88%	2%	0.88
LCC-0.05_R5	84%	3%	0.84

Table 6.6: Recognition and rejection rates of the systems combined by using linear committee combiner (closeness threshold = 0.1).

System Id.	Recognition	Rejection	Reliability
LCC-0.1_R1	94%	9%	0.93
LCC-0.1_R2	92%	8.7%	0.91
LCC-0.1_R3	91.6%	8%	0.91
LCC-0.1_R4	90%	5.7%	0.89
LCC-0.1_R5	87%	7%	0.86

Table 6.7: Recognition and rejection rates of the systems combined by using linear committee combiner (closeness threshold = 0.2).

System Id.	Recognition	Rejection	Reliability
LCC-0.2_R1	96.4%	21%	0.95
LCC-0.2_R2	94%	17%	0.93
LCC-0.2_R3	93.5%	16%	0.92
LCC-0.2_R4	92.4%	12%	0.91
LCC-0.2_R5	89%	14%	0.87

6.3.3 Multi-Label Classifier Combiner

In this section we present a method of combining different classifiers with rank level outputs. Because in this system the output of each classifier is a queue of k labels ordered according to their ranks, the system is called the *Multi-Label classifier Combiner*. As shown in Fig. 6-5, individual classification systems use different feature extraction techniques. The corresponding classifier of the ith system produces a queue of k_i ranked labels. The combiner unit then uses the queues to assign a class label to the output.

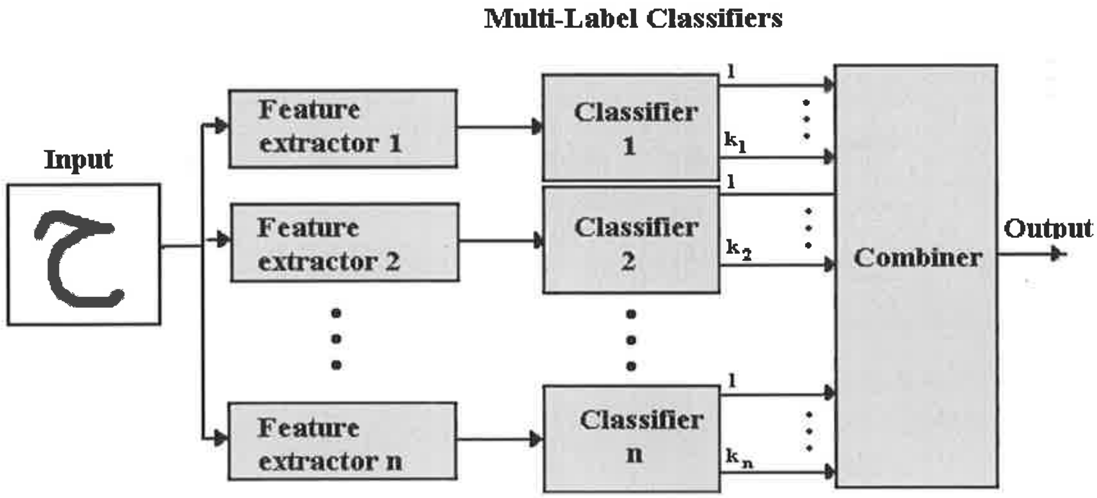


Figure 6-5: Block diagram of a multi-label classifier combiner.

In a single output label classifier, the output of the classifier is always a unique class choice which can be either a correct or a wrong decision. In some pattern recognition problems the classifiers are probably unable to uniquely identify the correct class but are able to output the correct class included in a set of output labels. In these applications using rank level classifier is recommended. A ranking of the classes carries much more information than a unique class choice [58]. The ranking method is generally applicable to all types of classifiers. The objective of the combiner unit is to determine the correct class from a set of labels at the output queues of individual classifiers.

Table 6.8: Rates of including the correct class label in a multi-label classifier.

	No. of Labels (k_i)								
System	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
PR	76%	86%	92%	94%	94%	94%	95%	96%	97%
NM	63%	80%	87%	91%	94%	96%	97%	98%	99%
SF	83%	95%	98%	98%	99%	99.6%	99.6%	99.6%	99.6%
CC	63%	80%	87%	91%	94%	96%	97%	98%	98%
LC	80%	92%	95%	96%	96%	96%	97%	97%	98%

Multi-label Classifier Selection

The first step in designing the combiner system is the selection of the classifiers to be combined. The criterion for selecting classifiers is based on their performances in classifying the input patterns into a queue which includes the correct class. If we consider a queue of length $k_i = M$ (M is the number of classes), all types of classifiers including the chance classifier, which selects the output by chance, will have a 100% recognition rate for including the correct class at the output! This implies that there should be a criterion for selecting a proper queue length (k_i) for each classifier. In our developed combiner system we use the recognition rates of the multi-label classifiers of the training set.

To measure the performance of the ith multi-label output classifier we define the probability of including the correct class label in an output queue with length k_i as

$$P_{k_i} = P(l_x \in \{q_j \ ; \ j = 1 : k_i\} \mid x) \tag{6.14}$$

where l_x is the correct label that should be assigned to the input x, and $\{q_j \ ; \ j = 1 : k_i\}$ is the output queue of the classifier with a length of k_i. With this definition, the rates of including the correct class label in the output queue of length k_i for the classifier systems studied in the previous chapter are as shown in Table 6.8.

Observations

- Increasing the length of the output queue will increase the probability of the correct class being included in the output queue.

- A high recognition rate of a single label classifier does not necessarily mean that this classifier has a better performance as a multi-label output classifier. For example system **NM** has a recognition rate of 63% as a single classifier which is less than 76% for **PR**, but **NM** reaches a recognition rate of 96% using a queue of length 6, while **PR** has only 94% correct recognition rate.

We assign a target recognition rate r_t; then, for each classifier we find the minimum queue length in which the system has a recognition rate equal to or exceeds r_t.

$$k_i = \min\{l_q \mid \ r(l_q) >= r_t\} \tag{6.15}$$

where l_q is the queue length, and $r(l_q)$ is the correct recognition rate of the system for the queue with length l_q. For example if we choose 96% as the target recognition rate and consider the rates in Table 6.8 as of the training set, then the length of the queues for **PR** to **LC** are $8, 6, 3, 6$, and 4 respectively.

Combination Method

The second step in designing the combiner system is to locate the correct class label from a set of queues each containing different numbers of class labels. The output queue of the ith classifier is denoted by Q_i, where:

$$Q_i = \{q_j; \ \ j = 1 : k_i\} \tag{6.16}$$

The output queues of individual classifiers are then combined to produce another queue. The output combined queue Q_c is defined as the intersection of all the queues of individual classifiers:

$$Q_c = \cap_i Q_i \tag{6.17}$$

Depending on the number of elements in the final queue, one of the three following cases may happen:

1. Q_c contains only one class label: Because with a high probability we were assured that the correct class is included in all the queues, this class label is most probably the correct class label.

2. Q_c contains more than one class label: this happens when there are very similar classes. In this case, we use Borda count method [21] to rerank the class labels in the final queue. Borda count method, which is a generalized form of majority vote, is defined for a class label as the sum of the number of class labels ranked below it by each individual voter (classifier):

$$B_c = \sum_{j=1}^{n} B_j(c) \tag{6.18}$$

where n is the number of classifiers, B_c is the Borda count for class label c, and $B_j(c)$ is the number of class labels ranked below c by the jth classifier. The bigger is the Borda count, the higher will be the rank of the class in the final queue. After calculating the Borda count for all labels in the final queue, we select the label with the highest Borda count as the correct class label.

3. Q_c is an empty set, i.e., $Q_c = \phi$: this means that the correct class label is not included in one or more classifiers. In this case we built the final queue by finding the labels which are only included in $n-1$ output queues, and repeat the first two above mentioned steps to find the correct class label. If the final queue is again an empty set we repeat the procedure for the labels which are only included in $n-2$ queues, and if the resulting queue is empty again, we reject the pattern.

Experimental Results

We used the systems in Table 6.8 as individual classifiers. The target recognition rate is adjusted to 96%. With this assumption a proper queue length (k_i) for each classifier is determined. The best result achieved for the combined system is 90% with a rejection rate of 10% which gives a reliability of 0.89.

As explained in the previous section, a pattern is rejected only when none of the labels is included in at least one of any group of $n-2$ queues. We ran another test in which we reject the pattern if it is not included in all the queues, i.e., $Q_c = \phi$. With this assumption, the recognition rate of the combined system jumps up to 94%

while the rejection rate is also increased up to 24% which gives a reliability of 0.92.

6.4 Conclusions

In this chapter we studied different techniques for combining the outputs of multiple classifiers. Each individual classifier may have an output in one of the abstract, rank, and measurement levels. Based on the various output levels, three systems of combiners were selected to test the performances of combined classifiers.

In the first system we used a weighted voting system for combining the outputs of the individual classifiers. In this combiner, each classifier outputs at the abstract level, which means it only gives a single class label as the output. The simplest way of voting is to assign an equal vote for these unique labels. However, since different classifiers have quite different performances on each class label, it is not appropriate to assign an equal vote for all class labels. By using the recognition rate of the classifier for each class label as the voting weight, a 90% recognition rate for a system without rejection was achieved. By rejecting about 28% of the patterns, the recognition rate increased up to 96%.

A system that combines the outputs of classifiers at the measurement level was then built by using different classifiers. Our experimental results of using a linear committee combiner showed slightly better performance than a voting combiner. In this system a recognition rate of 96.4% was achieved by rejecting 21% of the confusing patterns, which gave a reliability of 0.95.

A method of combining classifiers that have a rank level output was also introduced. A recognition rate of 94% with a rejection rate of 24% was achieved by using this combiner.

We showed that in all of the above mentioned combined systems, the recognition rates and the reliability of the combined systems outperform the single classification schemes. All the three types of combiners we studied could achieve a high reliability of more than 0.92; **LCC-0.2_R1** showed the best performance with a 0.95 reliability.

Chapter 7

Recognition of Handwritten Numerals

7.1 Introduction

In the previous chapter we used Persian and Arabic isolated handwritten characters. Numeral recognition, however, have more applications in real life. As we discussed in chapter 3, a lot more research is still needed to be done for Persian and Arabic handwritten character and numeral recognition. Due to the smaller number of patterns and smaller range of variation in writing styles, using the digits is a better start for designing and testing any new algorithm for handwritten character recognition.

This chapter is devoted to recognition of handwritten Persian and Arabic numeral symbols {۰،۱،۲، ... ،۹}. We introduce a line segment model to represent all the characteristics of these Persian and Arabic digits. Feature extraction is then based on a heuristic approach of detecting different line segments based on this model. The extracted features are also used to demonstrate the similarity between digits and variation in writing styles. Combination of multiple classifiers is also examined for increasing the overall recognition rate of the system.

Elastic matching technique has already been used for recognition of handwritten Arabic numerals $\{0, 1, 2, ..., 9\}$ [99]. In this chapter we will also study the performance of the elastic matching algorithm for recognition of handwritten Persian and Arabic digits. The elastic matching technique is presented by a brief review of the background theory followed by our proposed system for extracting the stroke sequences from off-line data. Finally, the results of a study of multiple expert combination is presented by introducing two combined system methodologies. The chapter is completed by some concluding remarks.

7.2 Persian and Arabic Numerals

Persian and Arabic have the same digit sets; however, there are few differences. For example, digits 4 and 6 have slightly different patterns in these languages. Figure 7-1 shows a sample of handwritten Persian numerals. As shown in this figure, images of the numerals consist of several line segments. For instance, digits $1, 2, 3, 4$, and 9 all have a vertical line segment in their patterns.

There is a range of variations on handwriting styles for numerals. Some sources of variations are:

- Vertical line segments are usually replaced by slanted or curved lines.
- Some digits may have two different shapes in Persian and Arabic, e.g. digits 4 and 6.
- Digit 0 may be either written as a small dot or like a small circle.

These characteristics introduce a large range of style variation in Persian and Arabic handwritten numerals.

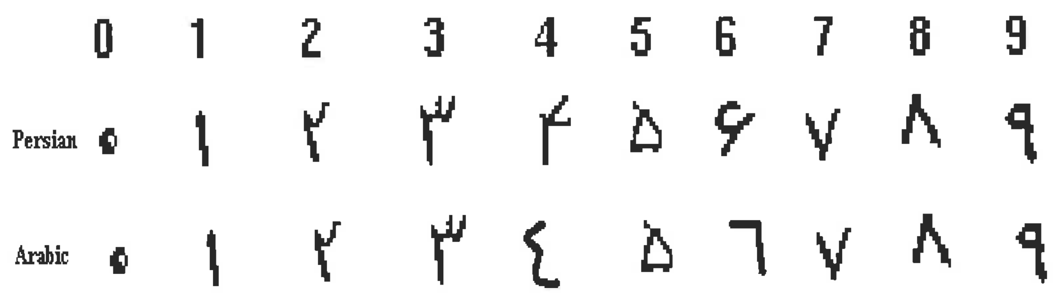


Figure 7-1: A handwritten sample of Persian and Arabic numerals.

7.3 Line Segment Model

In the previous two chapters we found that for Persian and Arabic, features that explore the structure of the character show a better recognition rate than the features based on the pixel distribution. In this section, we introduce a new structural feature especially designed for Persian and Arabic numerals. It is based on a heuristic approach of detecting different line segments in the image. First we introduce a line segment model for the digit patterns, and then we use a line crossing counting method for detecting the existence of the line segments in the image.

Unlike the Arabic numerals $\{0, 1, 2, .., 9\}$ which can be displayed by combinations of 7 line segments, the Persian or Arabic digits {۰, ۱, ۲, ··· , ۹} need more line segments. A model to represent these digits is shown in Fig. 7-2. This model consists of 11 horizontal and vertical line segments. Each digit can be represented by a subset of line segments from this model.. We should note that this model is not a practical way of displaying digits, but it is good enough, as we will see, for classifying them.

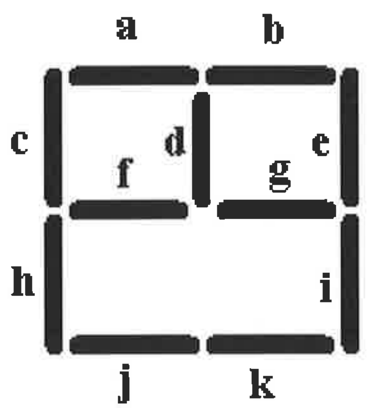


Figure 7-2: Line segment model for Persian and Arabic numerals.

7.3.1 Feature Extraction

The construction of the feature vector is based on the existence of line segments in the image. For each horizontal and vertical line segment a to k of the model of Fig. (7-2), we assign a binary vector V_a to V_k, respectively. The element of each vector represents the existence, "one", or absence, "zero", of a line segment in the corresponding row or column. Each vector is obtained from the region of the character matrix where the corresponding line segment may lie. For instance, the vector V_a is calculated at the upper left region of the character matrix.

To calculate the elements of the vectors V_a to V_k, we use the line crossing method in the vertical and horizontal directions. Each element of the vector that corresponds to a vertical line segment, e.g. V_c, V_d and V_e, is assigned a 1 if there exists a crossing between a horizontal scan line and the body of the character at the corresponding row, otherwise it is assigned a 0. Likewise each element of the vectors corresponding to horizontal lines, e.g. V_a, V_f and V_j, is assigned a 1 if there exists a crossing between a vertical scan line and the character body at the corresponding column. The binary vectors V_a to V_k are calculated by only one vertical and one horizontal scan through the character matrix. Figure 7-3 shows the pattern of digit 3 and its line segment representation.

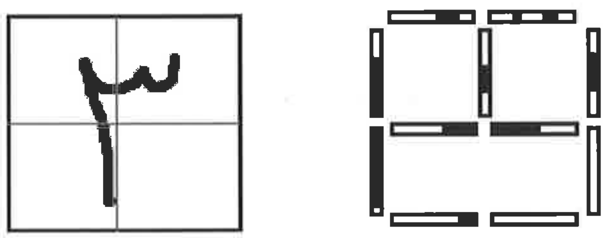


Figure 7-3: Image of digit 3 and its corresponding line segment model representation.

The feature vector f, is then built up of 10 elements by combining the vectors V_a to V_k as follows:

$$f = \begin{bmatrix} Sum(V_h \wedge \bar{V_i}) \\ Sum(V_i \wedge \bar{V_h}) \\ Sum((V_c \wedge V_e \wedge \bar{V_d}) \vee (V_c \wedge V_d \wedge \bar{V_e}) \vee (V_d \wedge V_e \wedge \bar{V_c})) \\ Sum(V_h \wedge V_i) \\ Sum(V_c \wedge V_d \wedge V_e) \\ Sum((V_a \wedge \bar{V_f}) \vee (V_f \wedge \bar{V_a})) + Sum((V_b \wedge \bar{V_g}) \vee (V_g \wedge \bar{V_b})) \\ Sum(V_j) + Sum(V_k) \\ Sum(V_a \wedge V_f) \\ Sum(V_b \wedge V_g) \\ Sum(V_a \wedge V_f \wedge V_j) + Sum(V_b \wedge V_g \wedge V_k) \end{bmatrix} \tag{7.1}$$

where $\vee$ and $\wedge$ are the logical OR and AND functions, respectively, and $sum(x)$ is a function that returns the summation of all elements of the vector x. Before applying this feature vector to the classifier, it is normalized into the range $[0, 1]$. The equations of the feature vector are carefully derived so as f represents most of the distinctive features of Persian and Arabic digits; each element of this vector shows the presence of one or group of line segments in a digit shape. Figure 7-4 shows the feature vector of a sample of digit 3. As shown in Fig. 7-3, this digit is characterized by a combination of a vertical line in the lower left part of the character (consisting of line segment h in Fig. 7-2) plus three smaller vertical lines in the upper region (line segments c, d and e) and a horizontal line in the middle part of the character matrix (line segments f and g). These characteristics are represented in Fig. 7-4 by large values of the elements $1, 3, 5, 6,$ and 7 of the feature vector ($f[1]$, $f[3]$, $f[5]$, $f[6]$ and $f[7]$).

Because we use the line crossing method, small changes in the curvature of the line segment will not affect the corresponding variable for that line segment. This means that this method is suitable for small changes in writing style. Also because the feature vector is normalized, the method is scale invariant.

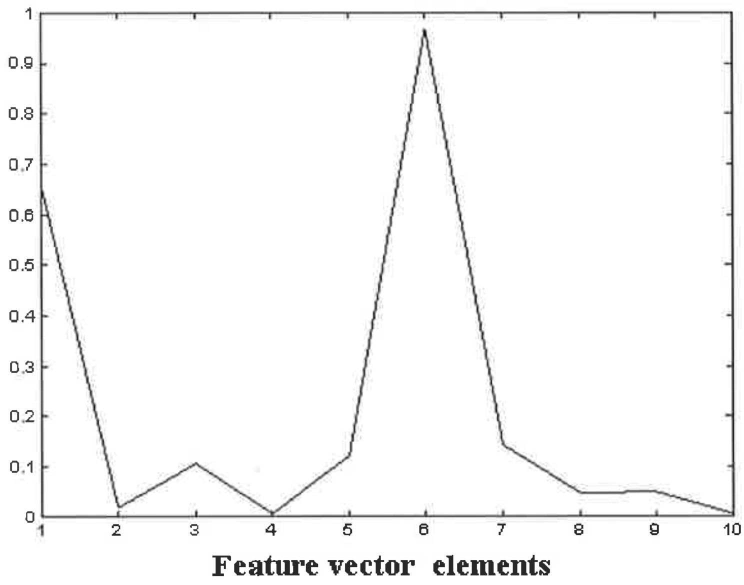


Figure 7-4: A sample feature vector of digit 3 calculated by the proposed feature extraction technique.

7.3.2 Similarity and Variability Analysis

As we discussed earlier in this chapter, there are possible similarities between the patterns of some Persian and Arabic digits, e.g. 0 and 5, which have very similar shapes but differ only in their sizes, or the digits 2 and 3, which differ only by a small stroke (Fig. 7-1). To show the similarity between digits, we use the average correlation coefficient between their feature vectors. Table 7.1 shows the correlation coefficients between the feature vectors of digits averaged over handwritten samples from 48 writers; pairs of digits with high correlation are highlighted in the table.

Observations:

- Pairs of very similar digits, (0,5), (2,3), (7, 8), and (4, 6), have large correlations.
- Despite the difference in shape of digits 7 and 8, they have very similar feature vectors with an average correlation coefficient of 0.98. This similarity is due to the feature extraction method; the line crossing method cannot detect the slope of a line segment.

Table 7.1: Average correlation coefficients between the feature vectors of digits.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	**0.87**	0.22	0.31	0.29	0.49	**0.82**	0.59	0.58	0.59	0.59
1	0.22	**0.78**	0.75	0.73	0.66	0.10	0.67	0.41	0.36	0.41
2	0.31	0.75	**0.96**	**0.94**	0.76	0.20	0.75	0.63	0.56	0.34
3	0.29	0.73	**0.94**	**0.95**	0.75	0.14	0.73	0.63	0.55	0.25
4	0.49	0.66	0.76	0.75	**0.84**	0.43	**0.78**	0.50	0.45	0.37
5	**0.82**	0.10	0.20	0.14	0.43	**0.92**	0.51	0.39	0.43	0.61
6	0.59	0.67	0.75	0.73	**0.78**	0.51	**0.84**	0.63	0.60	0.55
7	0.58	0.41	0.63	0.63	0.50	0.39	0.63	**0.99**	**0.98**	0.34
8	0.59	0.36	0.56	0.55	0.45	0.43	0.60	**0.98**	**0.99**	0.39
9	0.59	0.41	0.34	0.25	0.37	0.61	0.55	0.34	0.39	**0.88**

- The only difference between the digits 4 and 6 is that in 6 the line segment in the lower part of the digit is sloped while in 4 there are a vertical straight line and a small horizontal line (see Fig. (7-1)).

- Another interesting fact is that the average self correlation coefficients (diagonal elements in the table) are not exactly 1; this is caused by variations in handwriting styles. The digit 1 has the largest variation (with a correlation coefficient of only 0.78) and digits 7 and 8 have the smallest variation (with correlation coefficients of 0.99).

7.3.3 Recognition and Classification

In this section, we present the results of recognition of handwritten digits by the proposed feature extraction technique. The handwritten samples of 48 different writers are first digitized, and then the binary image of each digit is put in a 48×48 matrix. The preprocessing stage consists of thinning the binary image, and centering the character to solve the problem of translation. Handwritten samples are divided into two randomly selected disjoint sets: the training set containing samples from 10 writers and the test set containing samples from the other 38 writers.

We have tested the system by using different classifiers including a multi-layer Perceptron (MLP), which consists of ten input units, a hidden layer with 20 units,

Table 7.2: Recognition rates of single classifiers for handwritten digits.

System Id.	**Classifier**	**Recognition**
LS-1	MLP	80%
LS-2	EMD	80%
LS-3	PNN	77%
LS-4	KNN(1)	80%
LS-5	KNN(3)	79%
LS-6	KNN(5)	77%

Table 7.3: Confusion matrix of a single MLP classifier for handwritten numerals.

		Digits									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
	0	**36**	0	0	0	0	2	0	0	0	0
	1	0	**34**	1	0	1	0	0	1	0	1
D	**2**	0	0	**31**	1	0	0	**5**	0	0	0
i	**3**	1	0	**4**	**31**	1	0	0	1	0	0
g	**4**	0	1	0	0	**29**	0	**8**	0	0	0
i	**5**	**4**	0	0	0	0	**32**	0	0	2	0
t	**6**	2	0	2	3	1	0	**28**	1	0	1
s	**7**	0	0	0	0	0	0	0	**25**	**13**	0
	8	0	0	1	0	0	0	0	**12**	**25**	0
	9	0	1	0	0	0	1	1	0	0	**35**

and ten output units. Table 7.2 shows the recognition results of using different single classifiers without rejection.

Most of the misclassifications cases are caused by similar digits. Table 7.3 shows the confusion matrix of the MLP classifier. In particular the digits 7 and 8 have the highest misclassifications. However, this problem can be resolved, as we will see later, by combining multiple classifiers.

7.4 Elastic Matching

Perhaps the most widely known method which uses elastic deformation properties is *snakes* [68], [103]. A snake is a deformable spline (smooth curve segment) that is superimposed onto an image and deformed to match the image contours. Because smooth contours are sought, snakes are not well suited for contours that are not smooth, such as Persian and Arabic characters. In this section we apply the elastic matching technique for handwritten recognition of Persian and Arabic digits. The elastic matching technique has been used for the Arabic numeral set $\{0, 1, 2, ..., 9\}$ [99], and here we evaluate its performance for Persian and Arabic handwritten digits.

7.4.1 Background

Elastic matching (dynamic time warping) has been applied to speech recognition problems over two decades ago [65, 122]. Pioneered by Tappert [115], the technique was successfully applied to the recognition of handwritten characters. Elastic matching, however, was more often used in writer-dependent on-line recognition systems. The literature often reports a higher recognition rates for on-line handwritten data than for off-line data. There is apparently more information inherent in the data in on-line systems, which is collected as a table of points, than in scanned data of off-line systems. Finding corresponding points between image pairs is a fundamental problem when using elastic matching in off-line character recognition. By its very nature, elastic matching is well suited for a single writer on-line system; nevertheless, some researchers have reported its application to off-line data. For example, Scattolin [99] used elastic matching for off-line recognition of handwritten numerals, $\{0, 1, 2, ..., 9\}$.

The elastic matching algorithm was derived from the *dynamic programming* technique used for string matching [72]. When comparing two string sequences, three operations are allowed, namely *insertion*, *deletion*, and *substitution*. Each of these operations has an associated cost which is considered when calculating the distance between two sequences. Elastic matching is also used as a distance measure. A frequently used formulation of elastic distance is the one introduced by Tappert in [115]. The distance between an unknown sequence and a given model k is expressed as

$$D(i,j;k) = d(i,j;k) + \begin{cases} \min \begin{Bmatrix} D(i-1,j;k) \\ D(i-1,j-1;k) \\ D(i-1,j-2;k) \end{Bmatrix} & \text{if } \quad j > 2 \\ \min \begin{Bmatrix} D(i-1,j;k) \\ D(i-1,j-2;k) \end{Bmatrix} & \text{if } \quad j = 2 \\ \min \{D(i-1,j;k)\} & \text{if } \quad j = 1 \end{cases} \tag{7.2}$$

where $D(i, j; k)$ is the cumulative distance to point i in the input pattern (unknown) and point j in the kth template (prototype k), and $d(i, j; k)$ is a distance between points i and j which is usually a combination of the Euclidean distance and the difference between the elevation angles (ϕ_i, ϕ_j):

$$d(i,j;k) = (x_i - x_j)^2 + (y_i - y_j)^2 + c\left|\phi_i - \phi_j\right| \tag{7.3}$$

where c is a weighting constant and is empirically determined to give the maximum recognition rate. The elastic distance in Eq. (7.2) is normalized before it is used in a recognition task

$$D = \frac{D(n,m;k)}{n}, \tag{7.4}$$

where n and m are the number of points in the unknown character and the kth model respectively.

The warping function w maps the index of the points of the unknown character to the index of the prototype (see Fig. 7-5). The boundary conditions $w(1) = 1$ and $w(N) = M_k$, where M_k *is* the number of points in the kth prototype, ensure that the first and last points are matched. Elasticity is provided by the continuity condition $w(i+1) - w(i) = 0, 1, 2$, operating within the scope of the prototype. Thus, as shown in Fig. 7-5, successive points in the unknown character, e.g. points 4 and 5, may be mapped either to a single point, or two different points whose indices may differ by one or two (skipping one point). This is analogous to the insertion, deletion, and substitution concepts explained for string comparison at the beginning of this section.

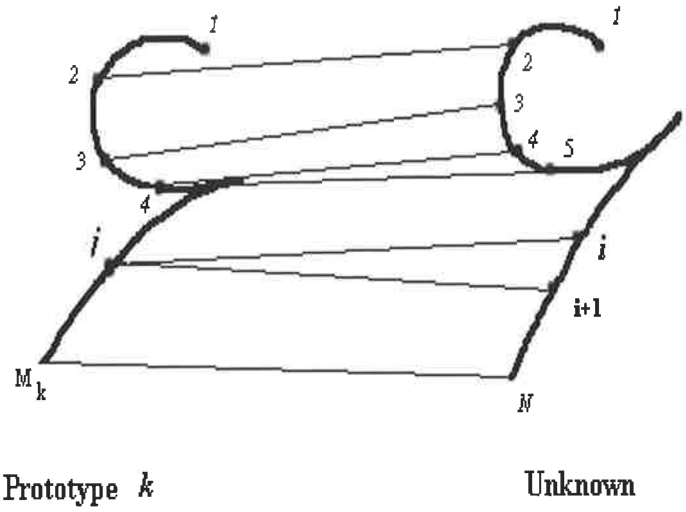


Figure 7-5: Elastic distance between a prototype and an unknown pattern.

One of the advantages of the elastic matching technique is that there is no need for complex feature extraction. The recursive nature of the elastic distance, however, makes it a very time consuming task. The time required to calculate the distance between an unknown character and a prototype depends on the length and number of the strings to be compared.

7.4.2 Proposed System

As mentioned before, elastic matching is well suited to on-line systems because in off-line systems the strokes' time order is not known; thus, it is more difficult to use elastic matching in off-line character recognition than on-line systems. In order to make a sequence of points, we need an algorithm to extract the dynamic information from the image. The proposed system of elastic matching recognition is shown in Fig. 7-6. Before describing the algorithm, we need to define some terms.

Definition 4 *An end point is a black pixel of the image which has only one neighboring black pixel.*

Definition 5 *A junction point is a black pixel with more than two neighboring black pixels.*

Definition 6 *A primitive stroke is a point sequence which has two end points on both ends.*

Definition 7 *A singular point is either a junction point or an end point.*

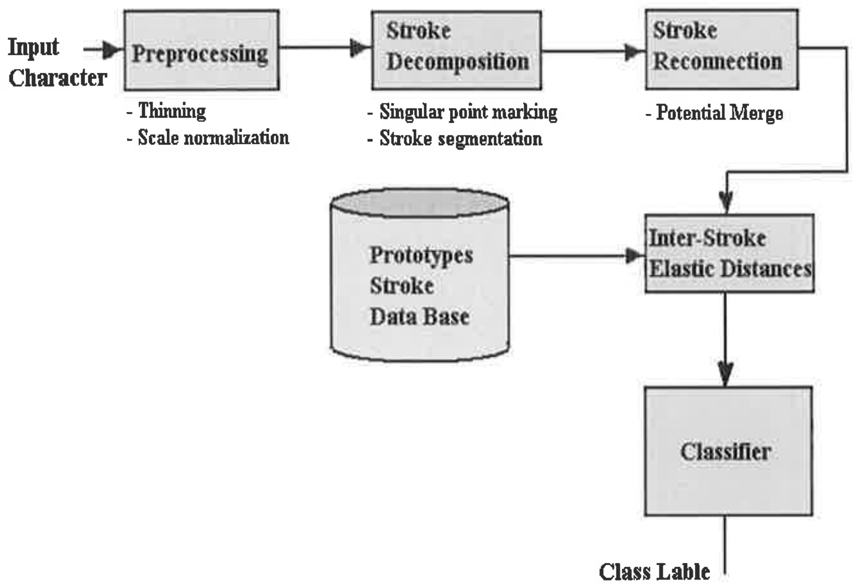


Figure 7-6: Proposed Elastic Matching system for recognition of handwritten characters.

Stroke Decomposition

After preprocessing, which consists of thinning and scale normalization, the character body is decomposed into primitive strokes. The decomposition algorithm is divided into two parts: singular point marking and segmenting the character body into primitive strokes. the character body is broken into different parts at the junction points, then the resulting strokes are examined to see whether or not they are primitive strokes. The process of segmentation and marking singular points continues until the character is completely divided into its primitive strokes. Finally each primitive stroke is traversed from one end point to the other, and is converted to a string of x-y coordinates.

Stroke Reconnection

The main issue when reconnecting the primitive strokes to build larger ones is the *principle of good continuation* [108]. This implies that, at a junction point, the best connection is the one which produces the least change in a continuous sequence [32]. Another issue in stroke reconnection is to minimize the angle difference between the two strokes meeting at a junction point [49, 69]. This means that two strokes are considered to have *good continuity* if the difference in the elevation angle is less than 45°.

The above algorithm for reconnecting the strokes does not connect all the primitive strokes, some may be left unconnected. Any of these primitive strokes may have one of the following relations to the previously connected stroke:

- one of its end points is very close to a point in the large stroke. It means that the distance between the endpoint and one of the larger stroke points is less than a threshold. In this case, the large stroke is broken at this point and the primitive stroke will be inserted in between as shown in Fig. 7-7. In this figure, the points of the smaller strokes are traversed and inserted twice, once in the forward order and once in the reverse order.

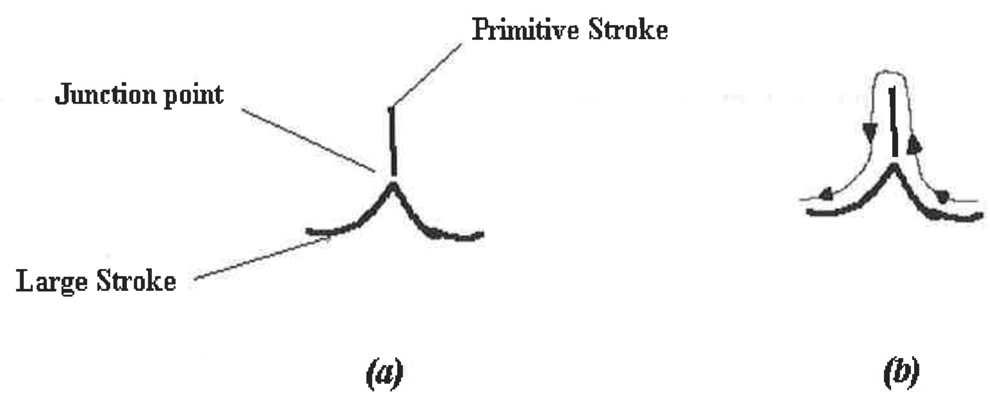


Figure 7-7: Reconnecting a primitive stroke to a larger stroke: (a) before reconnection, (b) writing order of the resulting stroke after reconnection.

- two of its end points are very close to two points of the larger stroke, thus creating a loop. In this case, points of the loop are inserted in between the larger stroke (see Fig. 7-8).

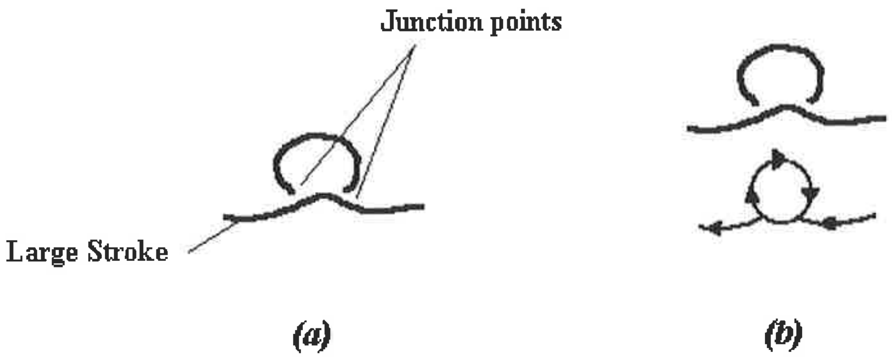


Figure 7-8: Inserting a loop into a larger stroke.

- none of the end points are close to any point of the larger stroke. This means that the strokes are two separate strokes and should not be connected.

The process of reconnecting the strokes is continued until there are no more possible connections.

Inter-Stroke Elastic Distance

As we mentioned earlier, the recursive elastic distance between two sequences in Eq. 7.2 is very computationally expensive. To overcome this problem we used the following iterative distance function introduced by Scattolin [99]:

$$DI(i,j;k) = DI(i-1,j;k) + \begin{cases} \min\left\{\begin{array}{l} d(i,j;k) \\ d(i,j+1;k) \\ d(i,j+2;k) \end{array}\right\} & \text{if} \quad j < m-1 \\ \min\left\{\begin{array}{l} d(i,j;k) \\ d(i,j+1;k) \end{array}\right\} & \text{if} \quad j = m-1 \\ \min\{d(i,j;k)\} & \text{if} \quad j = m \end{cases} \tag{7.5}$$

where $DI(1,1;k) = d(1,1)$. For a model k with m points, the above equation is evaluated iteratively from $DI(1,1;k)$ to $DI(n,m;k)$ which is the distance of the unknown pattern to the prototype k. One should note that the distance functions in Eq. 7.2 and Eq. 7.5 are not identical.

As we mentioned before, occasionally a character may have two or more separated strokes. To compare an unknown character U, with i strokes each with n_l points, to a model M, with j strokes each with m_j points, we first find the inter-stroke elastic distances and then use the following equation as the total distance between the two patterns

$$D_{U,M} = \sum_{l=1}^{i} \min_j \left(\frac{DI(n_l, m_j; S_k^j)}{n_l}\right) \tag{7.6}$$

where S_k^j represents the jth stroke of the model k, and $\frac{DI(n_l,m_j;Sj,k)}{n_l}$ is the normalized elastic distance between the ith stroke of the unknown pattern and jth stroke of the model.

7.4.3 Experimental Results

As we discussed earlier, the main disadvantage of the elastic distance is its computational expense, which becomes worse when increasing the number of prototypes. In the proposed system, only three handwritten samples are used for each digit. Another way is to use a clustering technique prior to the elastic matching.

Using the elastic matching distance features for a nearest neighbor classifier

(KNN, with $k = 1$), the recognition rate was 70%. The results show that the elastic matching technique also fails to distinguish between similar digits like (2, 3), (6, 9), and (0, 5). This is due to the similarity between the stroke sequences. The elastic distances between these similar digits are small, and hence they are easily confused by the classifier. For instance, as shown in Fig. 7-9, digits 2 and 3 differ only by a very small vertical stroke. Digits 0 and 5 also have similar stroke sequences, hence a very small elastic distance when they are normalized to the same scale.

Experimental results showed that the elastic matching fails to distinguish between similar patterns; however, it discriminates well between dissimilar digits and it still can be used in conjunction with other methods. In the next section we present the results of using multiple classifiers to resolve the problem of similarity between digits.

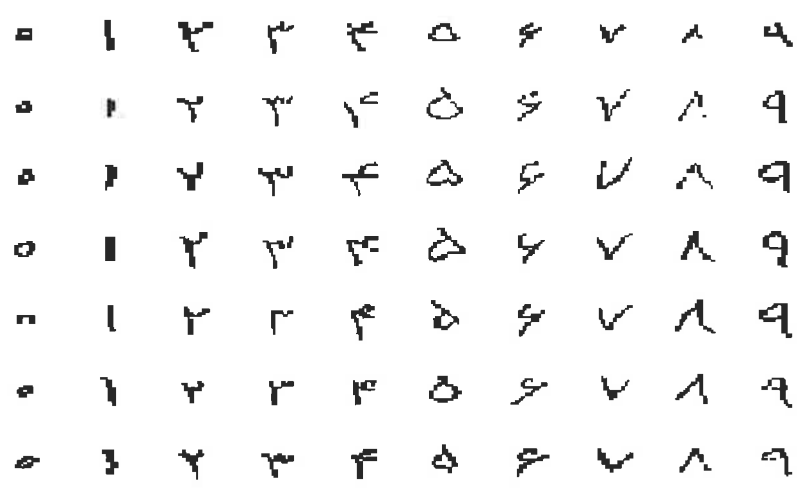


Figure 7-9: Preprocessed handwritten samples of Persian and Arabic numerals.

7.5 Multiple classifiers

The problem of similarity between digits, which becomes worse when they are distorted, leads us to use the idea of combination of classifiers. We present two methods of combination of multiple experts. In the first method the elastic matching technique is combined with the output of **LS-1** classifier system introduced in table 7.2.

In the second, we present a heuristic technique of multiple classifier combination which is especially designed to overcome the similarity problem.

The first system which combines the outputs of the elastic matching classifier and **LS-1** system, shows a better performance compared with the systems in table 7.2. The method of combination is similar to linear committee combiners explained in the previous chapter. Samples of 10 writers are used as the training set and the rest of the samples are used as the test set. A recognition rate of 89% was achieved by combination of the two classifiers. Though the elastic matching showed a poor performance as a single classifier, it improves the recognition rate of the **LS-1** system when they are combined. This means that these two classifiers complement each other; for similar patterns, the recognition is mainly done by the **LS-1** system and for other digits the elastic distance boosts the discrimination power of the combined system.

The main disadvantage of the combined system is its computational complexity. In the next section we will show that sometimes using a heuristic method that is based on specific characteristics of the patterns gives better results than an exhaustive algorithm like the elastic matching.

7.5.1 Gating Mixture of the Experts

We now present some trivial combination algorithms that enhance the performance of a classifier. In this method, the recognition rate of a classifier is improved by adding some simple classifiers that are especially designed to enhance the weaknesses of the main classifier. As shown in Fig. 7-10, the core of the combined system is called the *main classifier*. The input to the main classifier are the features extracted by the line segment model, and the classifier outputs a unique class label. Depending on this label, the decision making unit, which is simply a gating unit, decides whether to use the output label of the main classifier as the final output or use instead the output of one of the other smaller classifiers. The small classifiers are especially designed to distinguish between pairs of similar digits, and always outperform the main classifier for recognizing the confused digits.

As shown in table 7.3, there are digits which the main classifier confuses. Among the confused digits, there are three pairs which cause most of the confusions: $(0, 5)$, $(7, 8)$, and $(4, 6)$. For these three pairs of confused digits, three simple recognition systems are designed.

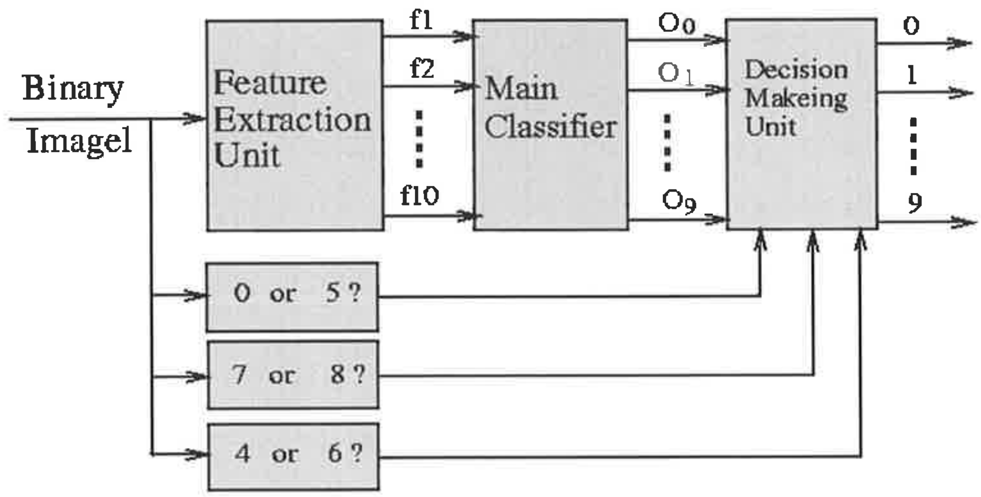


Figure 7-10: Block diagram of the combined system for handwritten numeral recognition.

To resolve the similarity problem between 0 and 5, we use the height and width of the digit. the height and width of the character is then applied to a single neuron trained with the Perceptron learning rule. To resolve the ambiguity between the feature vectors of 7 and 8, we find the average signed vertical distances of the pixels residing in the four-column wide central region (shaded region in Fig. 7-11). For the pixels above the horizontal axis the distance is considered negative, and for pixels below the horizontal axis it is considered positive. If the average is negative, the digit is an 8, otherwise it is a 7.

Digits 4 and 6 differ in the slope of their strokes in the lower part of the character matrix (see Fig. 7-12). To distinguish between the digits 4 and 6, two parameters are calculated and applied to a single neuron. The first one is the number of pixels in the lower part of the digit with a finite slope, and the second one is the number of pixels with an infinite slope in the same region.

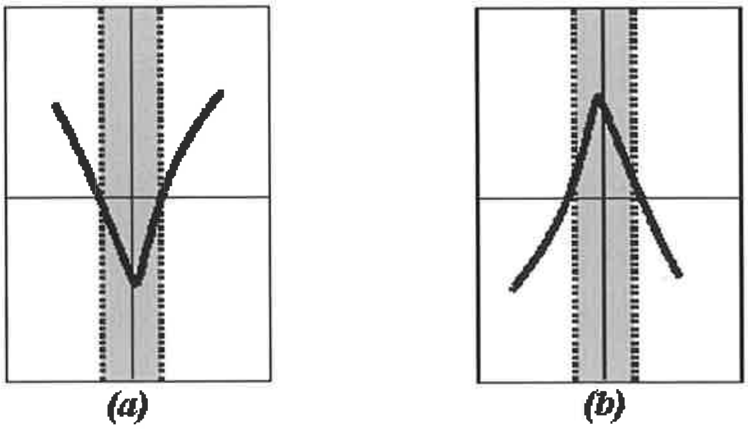


Figure 7-11: Distinguishing between a) digit 7, and b) digit 8 by detecting their pattern near the centre.

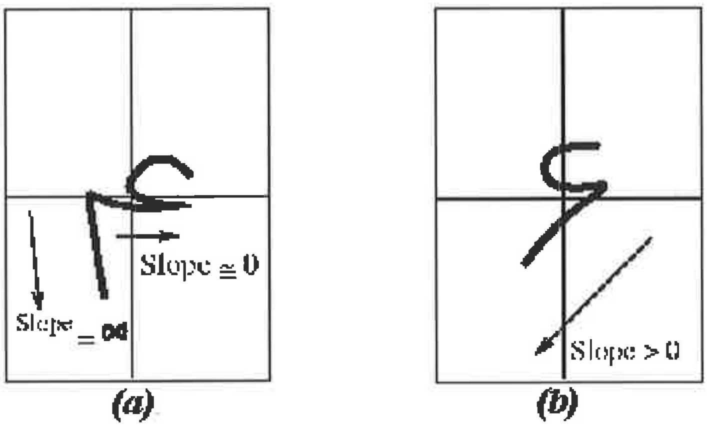


Figure 7-12: Distinguishing between a) digit 4, and b) digit 6 by detecting the slope at the lower part of the pattern.

Table 7.4: Recognition rates of different classifiers used as the main classifier of the gating mixture of experts.

System Id.	Main Classifier	Recognition
C_1	LS-1	91%
C-2	LS-2	86%
C-3	LS-3	85%
C-4	LS-5	88%
C-5	LS-6	87%

The decision making unit of the system combines the outputs of the main classifier with the outputs of the three simple classifiers. Depending on the output of the main classifier, the decision-making unit decides whether the output of the main classifier or the output of one of the simple classifiers should be used: if the output of the main classifier is one of the digits $0, 4, 5, 6, 7$ or 8, then the output of the corresponding small classifier is used instead.

The main classifier in Fig. 7-10 can be any of the recognition systems introduced in table 7.2. Table 7.4 presents the recognition rates of the combined systems. As shown in this table for all systems the performance is significantly improved by the combination of multiple classifiers.

As the results show, using trivial techniques for improving the weaknesses of the main classifier could work even better than a computationally expensive method like elastic matching, that showed a recognition rate of 89% when combined with **LS-1** system.

7.5.2 Rejecting the Patterns

As shown in the previous chapter, adding a rejection unit increases both the recognition rate and the reliability of the system. Again the *closeness* of the two top most outputs of the classifiers are used as a criterion for rejection. Table 7.5 shows the results for different combined systems. In all systems, the reliability and recognition rates are increased by adding the rejection unit.

Table 7.5: Recognition rate, rejection rate, and the reliability of the combined systems for handwritten numeral recognition.

System Id.	Main Classifier	Recognition	Rejection	Reliability
C_1	LS-1	94%	7%	0.94
C-2	LS-2	93%	9%	0.92
C-3	LS-3	88%	15%	0.86
C-4	LS-5	93%	9%	0.92
C-5	LS-6	91%	10%	0.90

7.6 Conclusions

In this chapter we studied the characteristics and recognition techniques of Persian and Arabic numerals. We introduced a new structural feature extraction technique especially designed for Persian and Arabic numerals. The method, which explores the structure of the characters, is based on a heuristic approach of detecting different line segments in the image. In the so called line segment model, each digit is represented by a combination of 11 vertical and horizontal line segments. This representation may not be a practical way of displaying digits, but it showed enough discrimination power for classification of the digits. The features are then extracted from the line segment model of the digit, based on the existence of certain line segments in its image. We tested the recognition power of the features by employing different single classifier systems, and the best recognition rate achieved was 80%. The study of the confusion matrices of the recognition systems revealed that most of the misclassification cases were caused by confusion between the digits $(0, 5)$, $(7, 8)$, and $(4, 6)$.

We also evaluated the elastic matching technique for recognition of handwritten digits. The elastic matching classification system showed a performance of 70% correct recognition. The experimental results also showed that the elastic matching technique failed to distinguish between similar digits like (2, 3), (6, 9), and (0, 5). Another problem with the elastic matching method is its computational cost.

Finally we presented two recognition systems based on the combination of multiple classifiers. The first system combined the outputs of the elastic matching classifier and the classification method based on the line segment model. A recognition rate of

89% was achieved, which is better than using only the elastic matching. The main pitfall of this combined system was, however, its computational complexity.

In the second system, a trivial gating scheme was introduced. In this method, the recognition rate of the main classifier was improved by adding some simple classifiers that were especially designed to enhance the weaknesses of the main classifier. By using this combination, the recognition rate increased up to 91%. The recognition rate and the reliability were then improved further by adding a rejection unit to the system. The best unit gave a performance of 94% correct recognition rate with 7% rejection rate, which gives a reliability of 0.94. For applications in which the wrong classifications of the digit patterns has a high cost, rejecting more ambiguous pattern will increase the reliability of the system.

Chapter 8

Conclusions

8.1 Summary

In this dissertation, we studied the problems of handwritten recognition of Persian and Arabic characters, and tried some possible solutions to overcome these problems. The stem of our research and investigation steps can be summarized as follows:

- We studied Persian and Arabic character sets, fonts, and handwritten styles in chapter 2. Potential difficulties of a handwritten recognition systems for these languages were highlighted.
- In chapter 3, we reviewed Persian and Arabic character recognition.
- Chapter 4 was devoted to analysis of Persian and Arabic handwritten characters. Two main issues, namely similarity of the patterns and handwriting variability, were addressed from a pattern deformation point of view. A distortion model was presented in this chapter. We also studied the effects of each distortion type on the patterns of individual characters.
- Different feature extraction algorithms and classifier design methods were investigated in chapter 5. We evaluated the performances of some of the techniques for both printed and handwritten characters, and compared with the performances of human experts. Some new feature extraction techniques were also proposed and tested.

- Combination of multiple classifiers was studied as a practical solution to the weaknesses of single classification schemes in chapter 6. Different strategies of combining different types of classifiers were proposed and studied and their performances were evaluated.

- Finally in chapter 7, handwritten recognition of Persian and Arabic numerals was studied. A new feature extraction technique and a classifier combination method were proposed.

8.2 Results and Conclusions

The main results and conclusions of this dissertation are:

- By presenting the difficulties of a handwritten recognizer for Persian and Arabic character sets, we showed why even by using the best feature extraction technique and the best classification methods it is almost impossible to resolve the problem of similarity between characters.

- A review on the literature revealed that here is still much to do to achieve a reliable system for Persian and Arabic character recognition. We then concluded that the lack of communication between the research groups, poor financial support, and the lack of standard data sets are big constraints for implementing commercial systems, as compared to the number of implementations of character recognition systems in other languages. However, the research on Persian and Arabic character recognition is also becoming more intensive than before and commercial systems are becoming available.

- A model was presented to describe the various distortions by geometric transformations. We used a mixed mode of distortion model and then used it to study the effect of each deformation on the individual characters. Using this model, possible sources of distortion of the handwritten characters were examined.

- By using the developed distortion model, we showed both theoretically and also by examples that various sources of distortions have different effects on individual characters. This implies that different normalization procedures are needed for individual patterns. We also used the distortion model to demonstrate the problem of similarity between the characters when distorted.

- Several feature extraction methods were proposed and compared with other methods reported in the literature. The results on the feature extraction methods showed different performances with the printed and handwritten samples. This means that a good recognition rate on the printed characters does not necessarily imply a good performance with handwritten patterns. Even some invariant feature extraction techniques showed a poor recognition rate on handwritten characters.

- Experimental results showed that among the different feature families those which are based on pixel distribution proved to be less successful than features based on the structure of the patterns.

- It is often an impossible task to compare the recognition systems reported in the literature. The results are usually obtained by adjusting different control parameters such as writer constraints, environments, and fine tuning of the system. By evaluating some of these systems we showed that they do not have high recognition rates by using our collected handwritten samples.

- The performance of human experts on the collected preprocessed data showed a similar reliability for all the subjects. The best reliability result for the human expert on the collected samples was 0.86. The interesting result is that the best proposed recognition system made almost the same mistakes as human experts; they showed a poor performance in distinguishing between similar patterns.

- The best recognition rates obtained by using a single classifier scheme were 83% without rejection and 88% with an 11% rejection rate of the ambiguous characters.

- We evaluated three different systems for combining multiple classifiers: weighted voting, linear committee combiner, and a multi-label combiner. In all cases the experimental results showed that the combined system always outperforms all of the individual classifiers. By rejecting ambiguous patterns, both the recognition rate and the reliability improved. Using *a prior* information on the performance of the individual classifiers for each class label increased the total recognition rate.

- Because the individual classifiers with measurement level outputs include more information at their outputs, they showed a better performance when they were combined. The best recognition results achieved by the weighted voting combiner, linear committee combiner, and multi-label combiner were 94%, 96%, and 94% with rejection rates of 28%, 21%, and 24%, respectively.

- A new feature extraction was developed for recognition of unconstrained handwritten Persian and Arabic numerals. The best recognition rate achieved for a single classifier system was 80%, while using a combined system increased the recognition rate up to 91%. The study of the confusion matrices of the recognition systems revealed that most of the misclassifications were caused by similar digits. The recognition rate was increased up to 94% by rejecting 7% of the patterns.

- We also studied the performance of the elastic matching method for handwritten digit recognition. As a single classification system, the elastic matching classification system showed a poor performance of 70%, while combining it with the classification method based on the line segment model increased the recognition rate up to 89%. Experimental results showed that the elastic distance between similar patterns, such as digits 2 and 3, is very small, hence the classifier fails to distinguish between similar characters. The main pitfall of this combined system was, however, its computational complexity.

8.3 Possible Research Directions

In this thesis we have studied some of the problems for handwritten Persian and Arabic character recognition. However, there are several directions in which this line of research might be continued. Still there is much work to do for any of the recognition systems we proposed if they are to be of any practical use. Some possible directions in the field are:

1. **Data Collection:** As we discussed in this dissertation, the researchers in the field of Persian and Arabic handwritten character recognition still suffer from the lack of a standard data base. Such a data set could be used for comparison between all algorithms developed for the recognition of printed and handwritten Persian and Arabic documents. The process of data collection might include form design, collecting unconstrained data from various sources, grouping the collected samples according to their qualities, grouping the samples according to the type of the text, e.g. printed, handwritten, etc., and comparing different algorithms reported in the literature.

2. **Applications of the Deformation Model:** We only used the distortion model to study the effect of each geometrical distortion on the individual characters; however, there are other possible applications for this model. For example, this model may be used for evaluation of quality of handwritings in calligraphy. The method can be used for calculating the distortion parameters of a piece of text compared to its original template. Depending on the importance of each geometrical distortion, a function that combines the effects of individual distortions may be defined. The output of the function can then be used for evaluation of the quality of written text.

3. **Improving the Performances of the subsystems:** The recognition rates can be improved further by employing other feature extraction techniques, and by improving the preprocessing unit. The performance also can be improved by investigating the methods of classifier selection in a combination of multiple classifiers. Those classifiers that complement each other produce a better

performance when they are combined.

4. **Text Recognition**: We only tested the system for isolated character recognition; the system should be tested for the characters that are produced by a segmentation process. Recognition of the characters and the segmentation process can also be combined. In the first step, the locations in a subword that can be a good candidate for dividing the subword into characters should be determined, then for these candidate points the recognition is performed. If the character recognizer fails to classify the segmented character, the next candidate location is used for segmentation.

5. **Using Contextual Information:** Even with the best feature extraction technique and the best classifier, we concluded that more evidence is needed for text recognition. As shown in Fig. 8-1, the character recognition unit instead of one output label may produce a ranked queue of labels. The word recognition unit which uses contextual information then decides which label is the correct character. This study includes the analysis of Persian and Arabic words, analysis of the subwords and possible combination of characters in a subword, and calculating the character sequence probabilities in a subword. In an application with a limited number of words, e.g. programming in a computer language, a dictionary of possible words can also be used as contextual information.

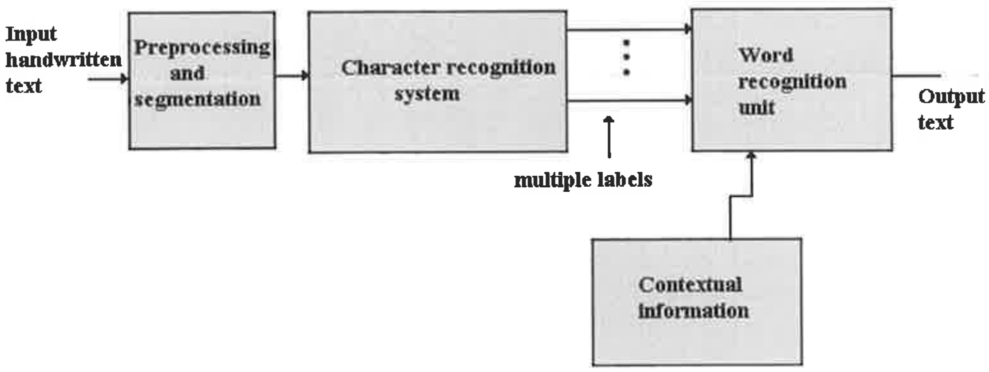


Figure 8-1: Block diagram of a handwritten text recognition system.

6. **Implementation issues:** any of the recognition systems we proposed still needs more work if it is to be of any practical use. Some of the techniques we developed are still computationally expensive, and there may be further improvements in their algorithms. For example, the combination of multiple classifier gives a better recognition result than a single classifier scheme but it increases the computation complexity. Because the outputs of different classifiers can be independently calculated, one possible solution is using a parallel implementation.

Bibliography

[1] I. S. I. Abuhaiba and P. Ahmed. Restoration of temporal information in off-line Arabic handwriting. *Pattern Recognition*, 26(7):1009–1028, July 1993.

[2] I.S.I. Abuhaiba, S. Datta, and M.J.J. Holt. Fuzzy state machines to recognize totally unconstructed. *Image and Vision Computing*, 13(10):755–770, 1995.

[3] I.S.I. Abuhaiba, S.A. Mahmoud, and R.J. Green. Recognition of handwritten cursive Aabic characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):664–672, 1994.

[4] B. Al-Badr and R.M. Haralick. Symbol recognition without prior segmentatin. In *Proceedings of the SPIE - Document Recognition*, pages 303 – 314, 1994.

[5] B. Al-Badr and R.M. Haralick. Segmentation-free word recognition with application to Arabic. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 355–359, 1995.

[6] B. Al-Badr and S. A. Mahmoud. Survey and bibliography of Arabic optical text recognition. *Signal Processing*, 41(1):49–77, 1995.

[7] H.B. Al-Sadoun and A. Amin. A new structural technique for recognizing printed Aabic text. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(1):101–126, 1995.

[8] H. Al-Yousefi and S.S. Udupa. Recognition of Arabic characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:853–857, 1992.

[9] Fevzi Alimoglu. Combining multiple classifiers for pen-based handwritten digit recognition. Master's thesis, Bogazici University, 1996.

[10] H. Almuallim and S. Yamaguchi. A method of recognition of Arabic cursive handwriting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:715–722, 1987.

[11] S. A. Alshebeili, A. A. F. Nabawi, and S. A. Mahmoud. Arabic character recognition using 1-d slices of the characters. *Signal Processing*, 56:59–75, 1997.

[12] A. Amin. State of the art on character recognition. In *Arabic Language Meeting Paris*, IBM, Paris, 22-24 January 1985.

[13] A. Amin. Machine recognition of multi font printed Arabic texts. In *Proceedings of the International Conference on Pattern Recognition*, pages 392–395, 1986.

[14] A. Amin and H.B. Al-Sadoun. A new segmentation technique of Arabic text. In *Proceedings of the International Conference on Pattern Recognition*, pages 441–445, 1992.

[15] A. Amin and J.F. Mari. Machine recognition and correction of printed Arabic text. *IEEE Transactions on Systems, Man and Cybernetics*, 19:1300–1306, 1989.

[16] A. Amin and G. Masini. Machine Recognition of cursive Arabic words. *Application of Digital Image Processing IV*, SPIE-359:286–292, 1982.

[17] V.D. Mazurov ans A.I. Krivonogov and V.L. Kazantsev. Solving of optimization and identification problems by the committee methods. *Pattern Recognition*, 20(4):371–378, 1987.

[18] K. Badi and M. Shimura. Machine recognition of arabic cursive scripts. *Trans. Institute of Electronics Communication Engineers*, E65:107–114, 1982.

[19] H.S. Baird, H. Bunke, and K. Yamamoto. *Structured Document Image Analysis*. Springer, 1992.

[20] B. Baluch and S. Shahidi. Visual word recognition in beginning readers of persian. *Perceptual and Motor Skills*, 72:1327–1331, 1991.

[21] D. Black. *The Theory of Committees and Elections.* Cambridge University Press, London, 1963.

[22] J. C. Bliss. A relatively high-resolution reading aid for the blind. *IEEE Trans. on Man, Mach. System*, 10:1–9, 1969.

[23] B.M.F. Bushofa and M. Spann. Segmentation of arabic characters using contour information. In *????*, pages ??–??, 1997.

[24] J. Cao, M. Shridhar, F. Kimura, and M. Ahmadi. Statistical and neural classification of handwritten numerals: A comparative study. In *Proceedings of the International Conference on Pattern Recognition*, pages 643–646, 1992.

[25] R.G. Casey and E. Lecolinet. Strategies in character segmentation: A survey. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1028–1033, 1995.

[26] S. Chen and F. Shih. A fuzzy model for unsupervised character classification. *Information Systems*, 2:143–165, 1995.

[27] F.H. Cheng and W.H. Hsu. Research on Chinese OCR in Taiwan. *International Journal of Pattern Recognition and Artificial Intelligence*, 5:139–164, 1991.

[28] Z. Chi, J. Wu, and H. Yan. Handwritten numeral recognition using self-organizing maps and fuzzy rules. *Pattern Recognition*, 28(1):59–66, 1995.

[29] F. Chin and F. Wu. A microprocessor-based optical character recognition check reader. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 982–985, 1995.

[30] J.P. Crettez. A set of handwriting families : style recognition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 489–494, August 1995.

[31] M. Dehghan and K. Faez. Farsi handwritten character recognition with moment invariants. In *Proceedings of 13th international conference on digital signal porcessing (DSP97)*, 1997.

[32] D. Doermann and A. Rosenfeld. The interpretation and recognition of interfering strokes. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 41–50, 1993.

[33] J-P. Drouhard, R. Sabourin, and M. Godbout. A comparative study of the k nearest neighbour, threshold and neural network classifiers for handwritten signature verification using an enhanced directional PDF. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 807–810, 1995.

[34] H.R. Drucker, R. Schapire, and P. Simard. Improving performance in neural networks using a boosting algorithm. In S.J. Hanson, J. Cowan, and Giles, editors, *Advances in Neural Information Processing Systems*. Morgan-Kaufmann, 1993.

[35] S.S. El-Dabi, R. Ramsis, and A. Kamel. Arabic character recognition system: a statistical approach for recognizing cursive typewritten text. *Pattern Recognition*, 23:485–495, 1990.

[36] T.S. El-Sheikh and R.M. Guindi. Computer recognition of Arabic cursive scripts. *Pattern Recognition*, 21:293–302, 1988.

[37] A.J. Elms and J. Illingworth. Combination of hmms for the representation of printed characters in noisy document images. *Image and Vision Computing*, 13(5):385–392, 1995.

[38] M. Fahimi and R. Sani. Persian handwritten recognition. In *Proceedings of the first annual CSI Computer Conference (CSICC95)*, pages 103–110, Tehran,Iran, December 1995. In Persian.

[39] M. Fathi and A. Broumand-Nia. Recognition of persian neumerals and isolated characters under non-uniform light condition. In *Proceedings of the first annual CSI Computer Conference (CSICC95)*, pages 96–102, 1995. In Persian.

[40] Habibollah Fazaeli. Katibeh-ye fazaeli. In *Atlas-e Khat (The Atlas of calligraphy)*, 1970.

[41] J. Franke and M. Oberlander. Writing style detection by statistical combination of classifiers in form reader application. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 581–584, 1993.

[42] H. Freeman. On the encoding of arbitrary geometric configurations. *IEEE Trans. Electron. Computers*, EC-10:260–268, 1968.

[43] K. Fukushima and T. Imagawa. Handwritten alphanumeric character recognition by the neocognitron. *IEEE transactions on Neural Networks*, 2(3):355–365, 1991.

[44] K.W. Gan and K.T. Lua. Chinese character classification using an adaptive resonance network. *Pattern Recognition*, 25:877–882, 1992.

[45] S. Garcia-Salicetti, B. Dorizzi, P. Gallinari, A. Mellouk, and D. Fanchon. A hidden Markov model extension of a neural predictive system for on-line character recognition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 50–53, 1995.

[46] A.V. Genkin and V.S. Pereverzen-Orlov. Automatic reading of handwritten characters: A review. *Pattern Recognition Theory and Applications*, 5(3):417–466, 1995.

[47] H. Goraine and M. Usher. Recognition of typewritten arabic characters in different fonts. In *Proc. IEE Colloquium on Character Recognition and Applications*, pages 9/1–5, 1989.

[48] V.K. Govindan and A.P. Shivaprasad. Character recognition – a review. *Pattern Recognition*, 23:671–683, 1990.

[49] V. Govindaraju and S.N. Srihari. Separating handwritten text from interfering strokes. In *From Pixels to Features III*, pages 17–28. North Holland, 1992.

[50] U. Grenander. *Hands : A pattern theoric study of biological shaps.* Springer-Verlag, 1991.

[51] W. Guerfali and R. Plamondon. Normalization and restoring on-line handwriting. *Pattern Recognition*, 26(3):419–431, 1993.

[52] D. Guillevic and C.Y. Suen. Cursive script recognition applied to the processing of bank cheques. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 11–14, 1995.

[53] I. Guyon. Applications of neural networks to character recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 5:353–382, 1991.

[54] Thien M. Ha and Horst Bunke. Design, implementation, and testing of perturbation method for handwritten neumeral recognition. Technical report, Institut fur Informatik und Angewandte Mathematik, Neubruckstr. 10, CH-3012 Bern, Switzerland, 1996.

[55] L.K. Hansen and P. Salamon. Neural network ensemblers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

[56] K.M. Hassibi. Machine-printed arabic ocr using neural networks. In *Proc. 4th International Conference and Exhibition on Multilangual Computing (Arabic and Roman Script)*, pages 2.3.1–2.3.12, University of Cambridge, London, UK, April 1994.

[57] Y. He, M.Y. Chen, and A. Kundu. Off-line handwritten word recognition using hmm with adaptive length viterbi algorithm. In *Proceedings of the International Conference on Pattern Recognition*, pages 460–462, 1994.

[58] T.K. Ho. *A Theory of Multiple Classifier Systems and Its Application to Visual Word Recognition.* PhD thesis, University of New York at Buffalo, 1992.

[59] M.K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. Inform. Theory*, IT-8:179–187, 1962.

[60] Y.H. Hu. Pattern classification with multiple classifiers. In *Tutorials of the International Symposium on Multi-Technology Information Processing*, pages 2–1:2–21, 1996.

[61] T.S. Huang and C.Y. Suen. Combination of multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):90–96, 1995.

[62] Y.S. Huang, K. Liu, and C.Y. Suen. The combination of multiple classifiers by a neural network approach. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(3):579, 1995.

[63] Y.S. Huang and C.Y. Suen. Combination of multiple classifiers with measurement values. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 598–601, 1993.

[64] S. Impedovo, L. Ottaviano, and S. Occhinegro. Optical character recognition – a survey. *International Journal of Pattern Recognition and Artificial Intelligence*, 5:1–24, 1991.

[65] F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-23:67–72, 1975.

[66] L.D. Jackel, H.P. Graf, W. Hubbard, J.S. Denker, and D. Henderson. An application of Neural Net chips : Handwritten digit recognition. In *IEEE International conference on Neural Networks*, pages 107 – 115, San Diego, 1988.

[67] K.M. Jambi. Arabic character recognition: Many approaches and one decade. *Arabian J. Engrg. Sci.*, 16(4):499–509, 1991.

[68] M. Kass, A. Witkin, and D. Terzopoulos. Snakes : Active contour models. In *Proceedings of First International Conference on Computer Vision*, pages 259–269, 1987.

[69] T. Kasvand and N. Otsu. Recognition of line shapes based on thinning, segmentation with good connectivity algorithms, and regularization. In *Proceedings of the International Conference on Pattern Recognition*, pages 497–500, 1984.

[70] F. Kimura and M. Shridhar. Handwritten numeral recognition based on multiple algorithms. *Pattern Recognition*, 24:969–983, 1991.

[71] G.V. Kondraske and A. Shennib. Character pattern recognition for a telecommunication aid for the deaf. *IEEE trans. Biomed. Eng*, 33:366–370, 1986.

[72] J. B. Kruskal. An overview of sequence comparison : Time warps, string edits, and macromolecules. *Society for Industrial and Applied Mathematics SIAM Review*, 25(2):201–237, 1983.

[73] A. Krzyzak, W. Dai, and C.Y. Suen. Unconstrained handwritten character classification using modified backpropagation model. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 155–166, 1990.

[74] Y. LeCun, O. Matan, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, and H.S. Baird. Handwritten zip code recognition with multilayer networks. In *Proceedings of the International Conference on Pattern Recognition*, pages 35–40, 1990.

[75] F.J. Maarse, R.G.J. Meulenbroek, H.L Teulings, and A.J.W.M. Thomassen. Computational measures for ballisticity in handwriting. In *Proceedings of the Third International Symposium on Handwriting and Computer Applications*, pages 16–18, 1987.

[76] P.R. Madhvapathy. Pattern recognition using simple measures of projections. Master's thesis, Colorado State University, Fort Collins, 1986.

[77] S. A. Mahmoud, I.S.I Abuhaiba, and R.J. Green. Skeletonization of arabic characters using cluster based skeletonization algorithm (cbsa). *Pattern Recognition*, 24(5):453–464, 1991.

[78] A. Malaviya and L. Peters. Extracting meaningful handwriting features with fuzzy aggregation method. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 841–844, 1995.

[79] J. Mantas. A fuzzy decision operator in recognizing Greek hand-drawn characters. In *Proceedings of the International Conference on Pattern Recognition*, pages 753–755, 1986.

[80] J. Mantas. An overview of character recognition methodologies. *Pattern Recognition*, 19:425–430, 1986.

[81] V. Märgner. SARAT – A system for the recognition of Arabic printed text. In *Proceedings of the International Conference on Pattern Recognition*, pages 561–564, 1992.

[82] T. Matsui, T. Noumi, I. Yamashita, T. Wakahara, and M. Yoshimuro. State of the art of handwritten numeral recognition in japan - the results of the first ipts character recognition competition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 391–396, 1993.

[83] J. L. McClelland and D. E. Rumelhart. An interactive activation model of context effects in letter perception : Part1. an account of basic findings. *Psychological Review*, 88(5):375–407, September 1981.

[84] S. Mori, C.Y. Suen, and K. Yamamoto. Historical review of OCR research and development. *Proceedings of the IEEE*, 80:1029–1058, 1992.

[85] N.J. Naccache and R. Shinghal. Spta: A proposed algorithm for thinning binary patterns. *IEEE Transactions on Systems, Man and Cybernetics*, 14(3):409–418, 1984.

[86] G. Nagy. At the frontiers of OCR. *Proceedings of the IEEE*, 80:1093–1100, 1992.

[87] G. Nagy and N. Tuong. Normalisation techniques for hand-printed numerals. *Communications of the Association for Computing Machinery*, 13(8):475–481, Aug. 1970.

[88] A. M. Nazif. A system for the recognition of printed arabic characters. Master's thesis, Faculty of Engineering, Cairo University, Cairo, Egypt, 1975.

[89] N.J. Nilsson. *Learning Machinrs.* McGraw-Hill, 1965.

[90] B. Parhami and M. Taraghi. Automatic recognition of printed Farsi texts. *Pattern Recognition*, 14(1 - 6):395 – 403, 1981.

[91] M. Parizeau and R. Plamondon. A fuzzy-syntactic approach to allograph modeling for cursive script recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):702–712, 1995.

[92] B. Plessis, A. Sicsu, L. Heutte, E. Menu, E. Lecolinet, O. Debon, and J.V. Moreau. A multi-classifier combination strategy for the recognition of handwritten cursive words. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 642–645, 1993.

[93] R.K. Powalka, N. Sherkat, and R.J. Whitrow. Recognition characterization for combining handwriting recognition results at word level. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 68–73, 1995.

[94] Robert K. Powalka. *An Algorithm toolbox for On-line Cursive Script Recognition.* PhD thesis, The Nottingham Trent University, May 1995.

[95] T.H. Reiss. *Recognizing Planar Objects Using Invariant Image Features.* Lecture Notes in Computer Science. Springer-Verlag, 1993.

[96] M. Revow, C. K.I. Williams, and G. E. Hinton. Using mixtures of deformable models to capture variations in hand printed digits. In *Proceedings of the*

International Workshop on Frontiers in Handwriting Recognition, pages 142–152, 1993.

[97] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 332:533–536, 1986.

[98] M. Safabakhsh and M.A. Shaygan. 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, Dec. 1995.

[99] P. Scattolin. Recognition of handwritten numerals using elastic matching. Master's thesis, Concordia University, Canada, 1995.

[100] L. Schomaker, G. Abbink, and G. Selen. Writer and writing-style classification in recognition of online handwriting. In *Proceedings of the European Workshop on Handwriting Analysis and Recognition : A European Prespective*, volume Digest Number 1994/123, London, July 1994. The Institution of Electrical Engineers.

[101] L. Schomaker and R. Plamondon. The relation between pen force and pen point kinematics in handwriting. *Biological Cybernetics*, 63:227–289, 1990.

[102] E.L. Schwartz. Spatial mapping in the primate sensory projection: Analytic structure and relevance to perception. *Biol. Cybernetics*, 25:181–194, 1977.

[103] A.W. Senior and F. Fallside. Using constrained snakes for feature spotting in off-line cursive script. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 305–310, 1993.

[104] A. Shoukry. Arabic character recognition state of the art. In *Proc. 11th National Computer Conf.*, pages 382–390, Dhahran, Saudi Arabia, 1989.

[105] F. Smieja. Rejection of incorrect answers from a neural net classifier. Technical report, German National Research Centre for Computer Science (GMD), Germany, 1993. Report number:1993/2.

[106] D.F. Specht. Probabilistic neural networks. *Neural Networks*, 3(1):109–118, 1990.

[107] J.V. Spiegel, G. Kreider, C. Claeys, I. Debusschere, G. Sandini, P. Dario, F. Fantini, P. Bellutti, and G. Soncini. A foveated retina-like sensor using ccd technology. In C. Mead and M. Ismail, editors, *Analog VLSI and Neural Network Implementations.* SeKluwer Publ., 1989.

[108] K.T. Spoehr and S.W. Lehmkuhle. *Visual Information Processing.* W.H. Freeman and Company, 1982.

[109] S.N. Srihari. Reliability analysis of majority vote systems. *Information Sciences*, 26:243–256, 1982.

[110] S.N. Srihari. Recognition of handwritten and machine-printed text for postal address interpretation [patrec 1112]. *Pattern Recognition Letters*, 14(4):291–302, April 1993.

[111] C. Y. Suen. Distinctive features in automatic recognition of handprinted characters. *Signal Processing*, 4:193–207, 1982.

[112] C.Y. Suen, R. Legault, C. Nadal, M. Cheriet, and L. Lam. Building a new generation of handwriting recognition systems. *Pattern Recognition Letters*, 14(4):303–316, April 1993.

[113] C.Y. Suen, C. Nadal, T.A. Mai, R. Legault, and L. Lam. Recognition of totally unconstrained handwritten numerals based on the concept of multiple experts. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 131–144, 1990.

[114] Y.Y. Tang, H.D. Cheng, and C.Y. Suen. Transformation-ring-projection (trp) algorithm and its vlsi implementation. *International Journal of Pattern Recognition and Artificial Intelligence*, 5(1, and 2):25–56, 1991.

[115] C. C. Tappert. Cursive script recognition by elastic matching. *I.B.M. Journal of Research and Development*, 26:765–771, 1982.

[116] C.C. Tappert. Adaptive on-line handwriting recognition. *IEEE Seventh Int. Conf. on Pattern Recognition*, 1984, August. 1004-1007.

[117] C.C. Tappert, C.Y. Suen, and T. Wakahara. The state of the art in on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(8):787–808, 1990.

[118] H.L. Teulings and F.J. Maarse. Digital recording and preprocessing of handwriting movements. *Human Movement Science*, 3:193–217, 1984.

[119] A.J.W.M. Thomassen, H.L. Teulings, H.L. Sckomaker, L.R.B. Morasso, and P.J. Kennedy. Towards the implementation of cursive-script understanding in an online handwriting-recognition system. In *ESPRIT'88 : Putting the technology to use (Part 1)*, pages 628–639, 1988.

[120] J.J. Hull T.K. Ho and S.N. Srihari. Combination of decisions by multiple classifiers. In *Structured Document Image Analysis*, pages 188–202. Springer-Verlag, 1992.

[121] A.M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12:97–136, 1980.

[122] V.M. Velichko and N.G. Zagoruyko. Automatic recognition of 200 words. *International Journal Man Machine Studies*, 2:223–234, 1970.

[123] J. R. Ward. One view of on-going problems in handwriting character recognition. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 101–108, 1990.

[124] J.R. Ward and T. Kuklinski. A model for variability effects in handprinting with implications for the design of handwriting character recognition systems. *IEEE Transactions on Systems, Man and Cybernetics*, 18:438–451, 1988.

[125] D. Whitteridge and P.M. Daniel. The representation of the visual field on the cerebral cortex in monkeys. *J. Physiol.*, 159:203–221, 1961.

[126] C. K. I. Williams, M. Revow, and G. E. Hinton. Instantiating deformable model with a neural net. *To appear in : Computer Vision and Image Understanding*, 1996.

[127] D.H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

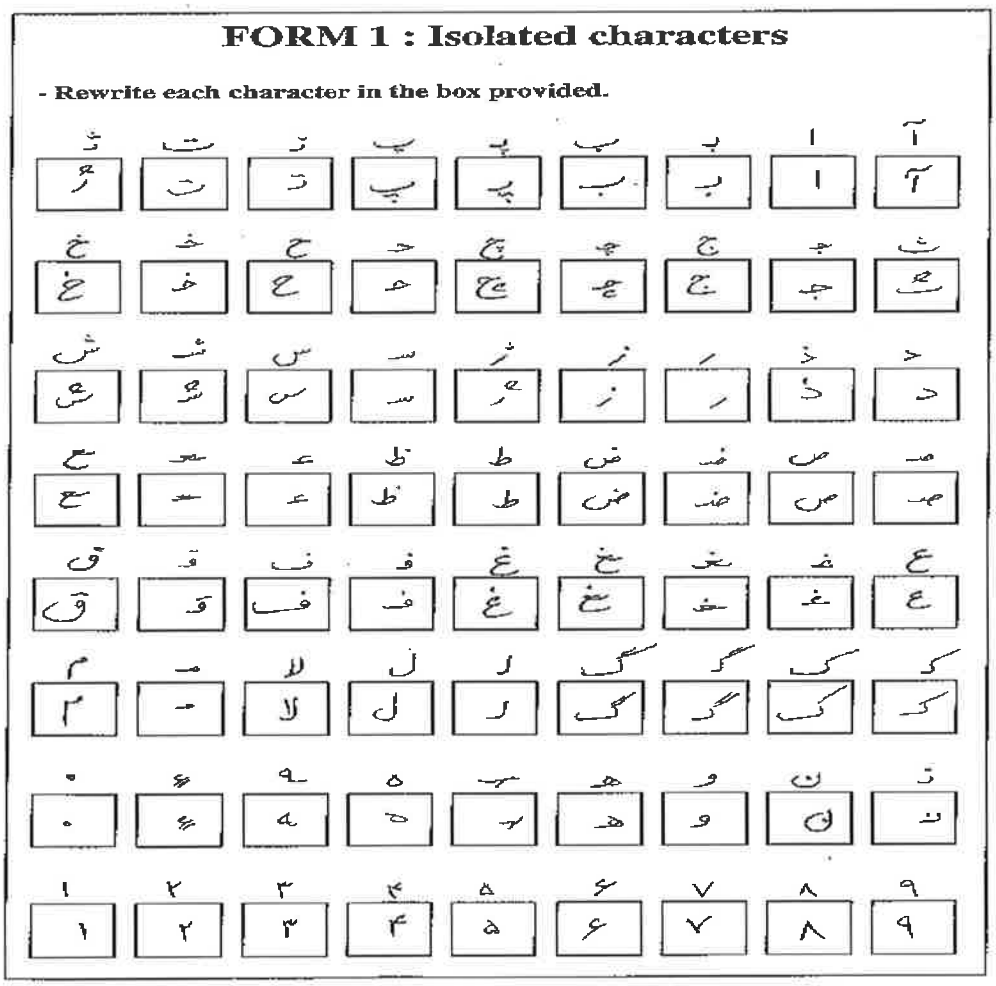
[128] L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 22:418–435, 1992.

[129] F. Yamaoka, Y. Lu, A. Shaout, and M. Shridhar. Fuzzy integration of classification results in a handwritten digit recognition system. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 255–264, 1994.

[130] M. Zhang, C.Y. Suen, and T.D. Bui. An optimal pairing scheme in associative memory classifier and its application in character recognition. In *Proceedings of the International Conference on Pattern Recognition*, pages 50–53, 1992.

Appendix A

Designed Forms


FORM 1 : Isolated characters

- Rewrite each character in the box provided.

Figure A-1: The form designed to collect Persian handwriting isolated characters.

FORM 2 : Sentences

یا رب این نوگل خندان که سپردی به منش | میسپارم بتو از چشم حسود چمنش

شعر حافظ همه بیت الغزل معرفتست | آفرین بر نفس دلکش و لطف سخنش

چو با حبیب نشینی و باده پیمایی | بیاد دار محبان باده پیما را

بخلق و لطف توان کرد صید اهل نظر | ببند و دام نگیرند مرغ دانا را

در آسمان نه عجب گر به گفته حافظ | سرود زهره برقص آورد مسیحا را

- Rewrite the above poems here in separate lines.

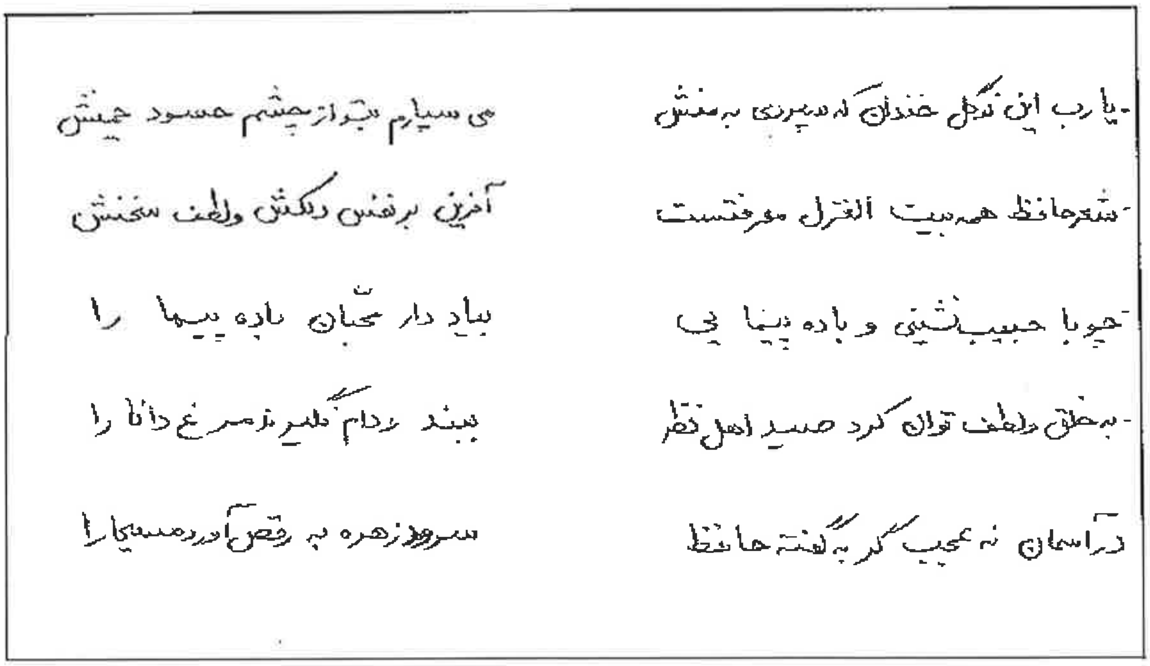
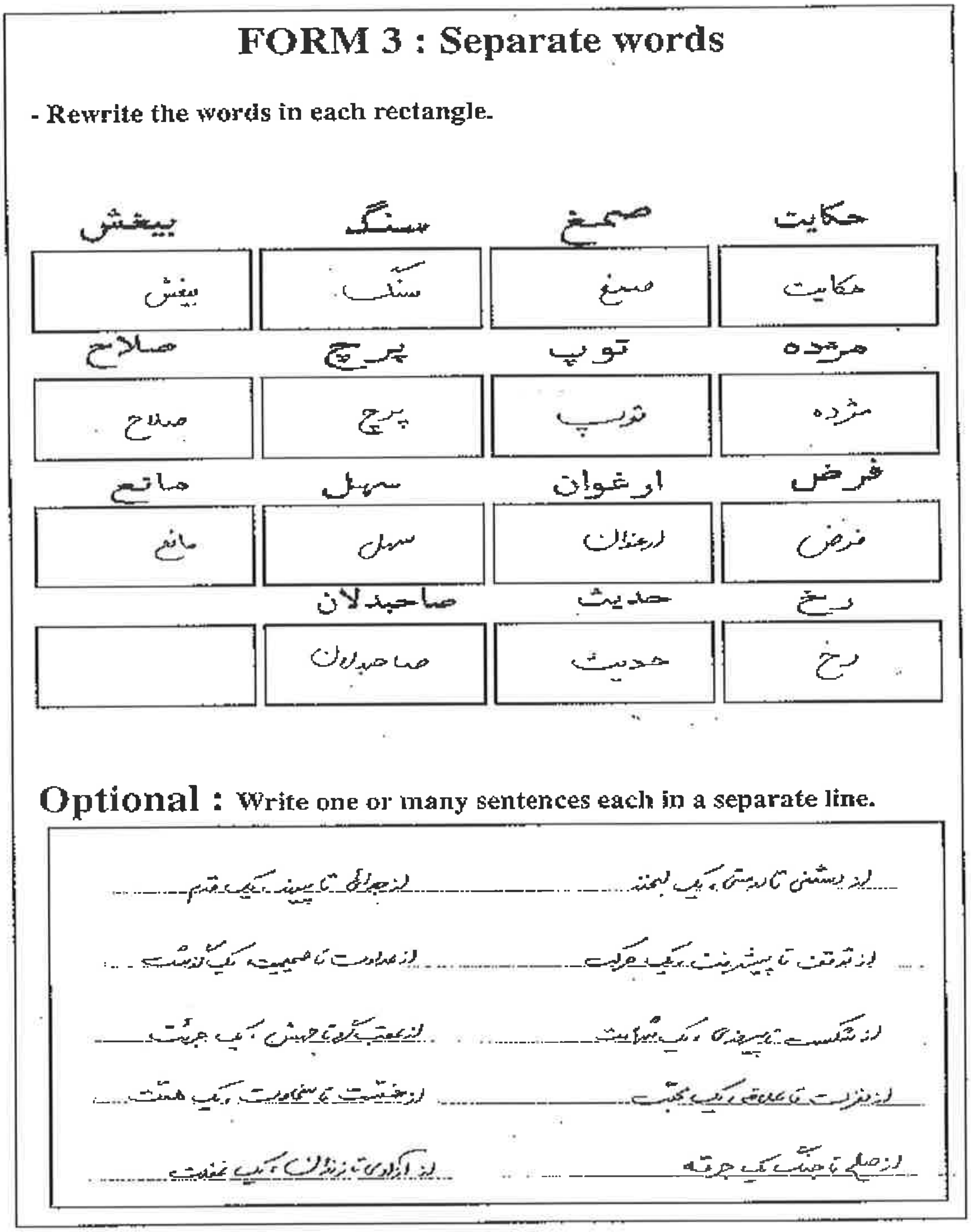


Figure A-2: The form designed to collect unconstrained Persian handwriting sentences.



FORM 3 : Separate words

- Rewrite the words in each rectangle.

حکایت	صمغ	سنگ	بیغش
مژده	توپ	پرچ	صلاح
فرض	ارغوان	سهل	مانع
رخ	حدیث	صاحبدلان	

Optional : Write one or many sentences each in a separate line.

Figure A-3: The form designed to collect unconstrained Persian handwriting words and sentences.

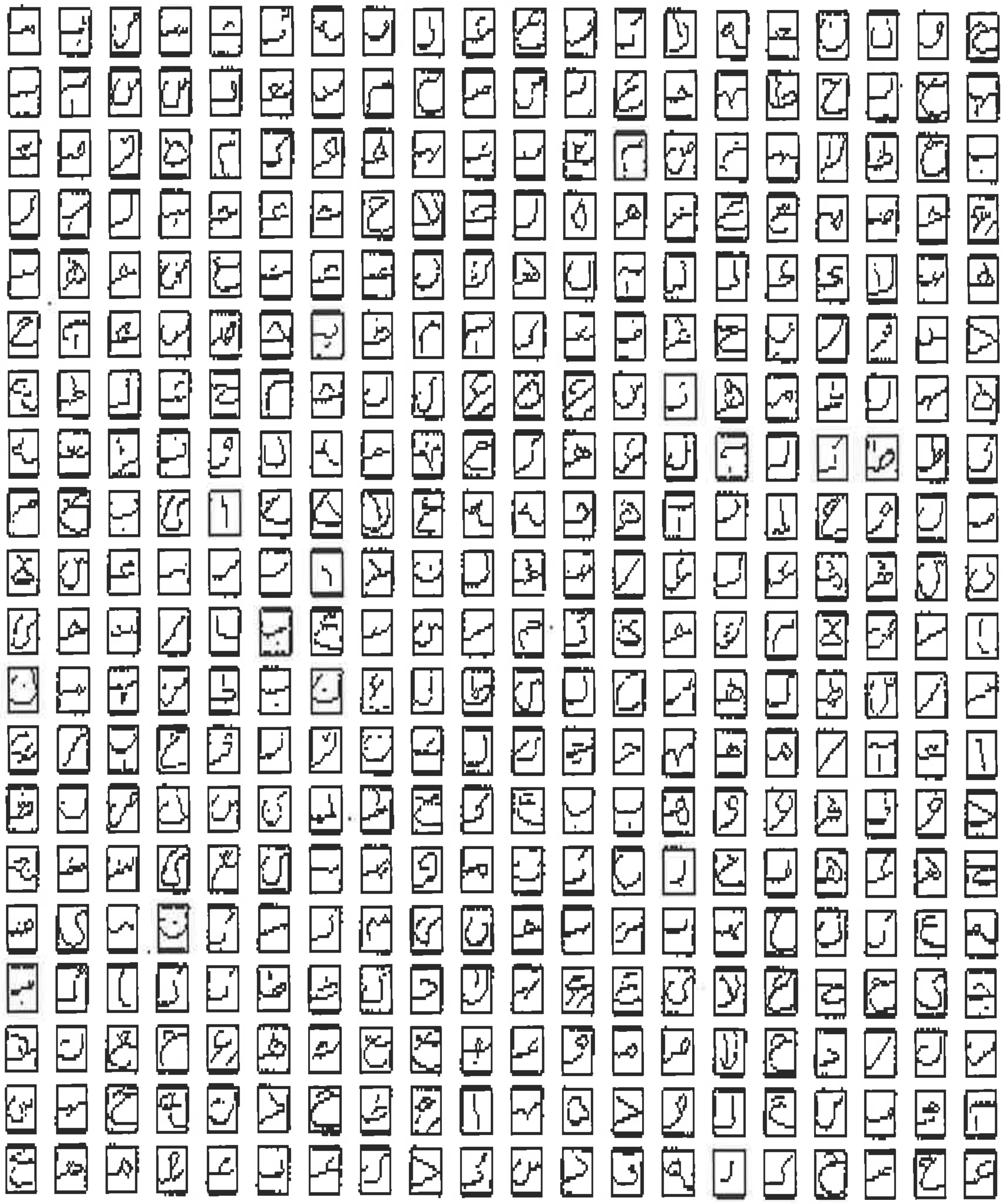


Figure A-4: The form designed for testing the recognition rate of human experts.

Appendix B

Distortion Characteristics

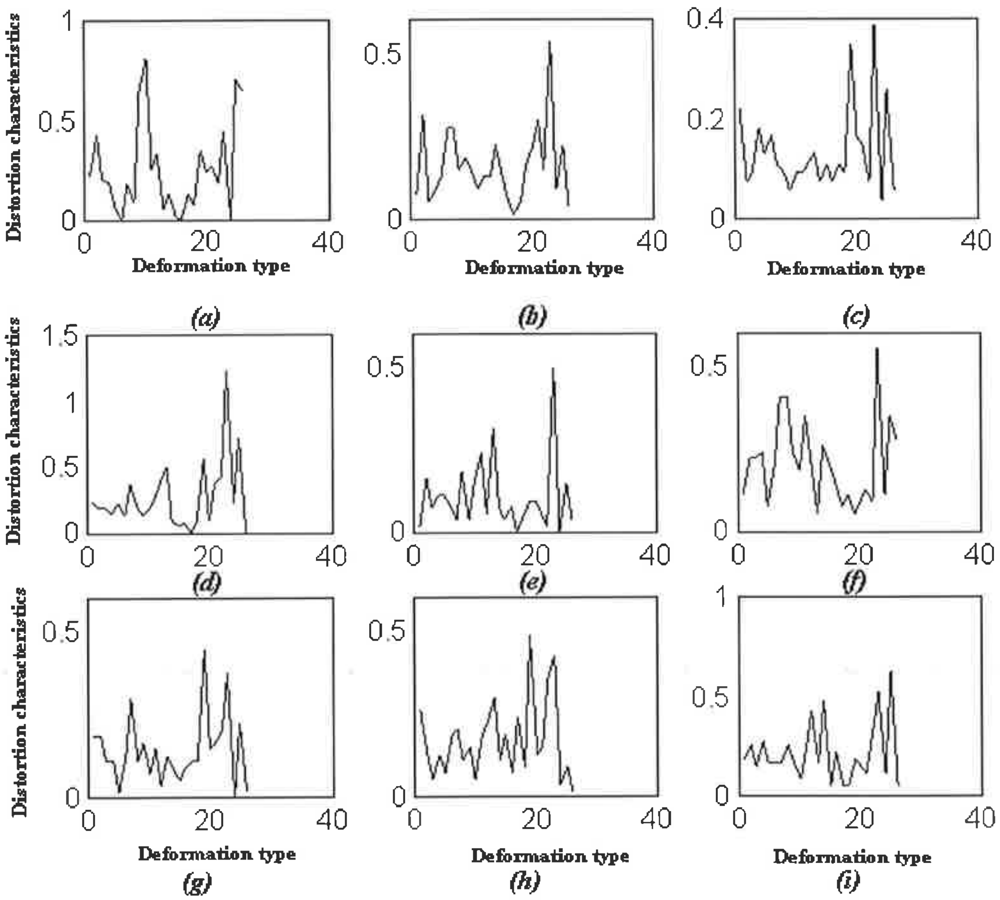


Figure B-1: Distortion characteristics of different Persian characters: a) "ALEF", b) "BA", c) "HEH", d) "(beginning) HEH", e) "DAL", f) "RA", g) "SEEN", h) "SAD", and i) "TTA".

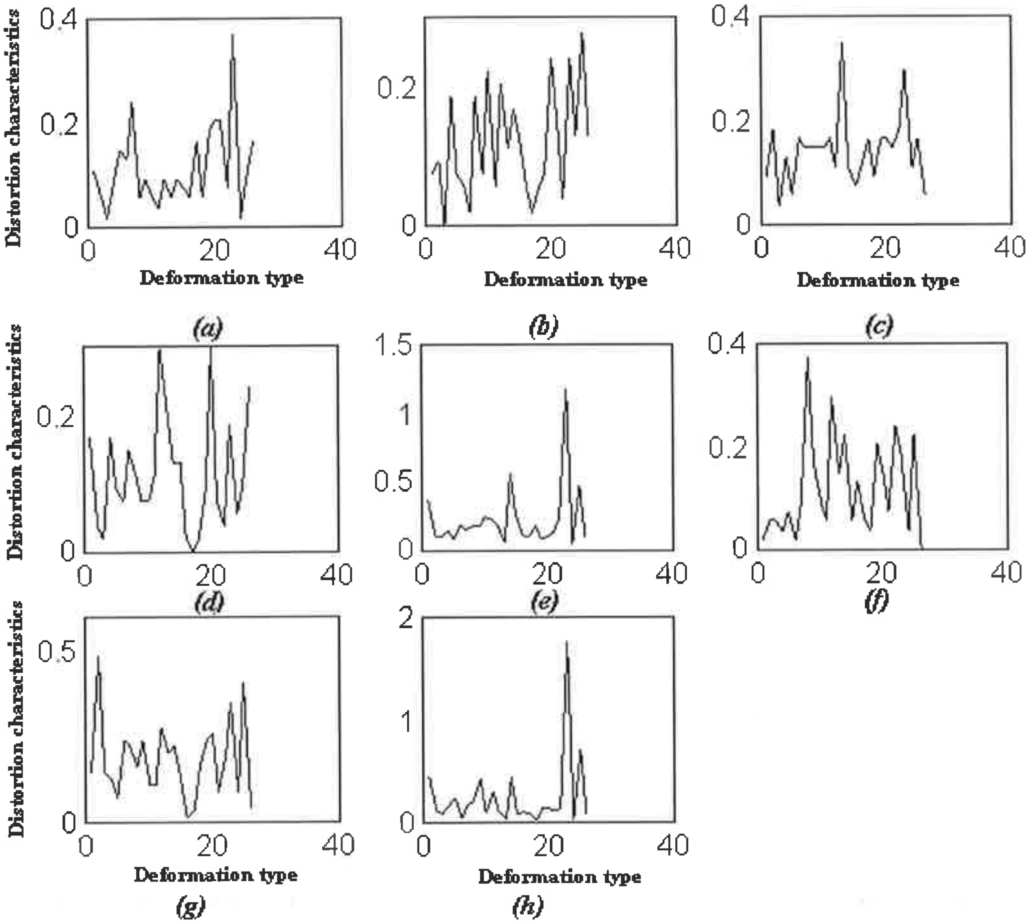


Figure B-2: Distortion characteristics of different Persian characters: a) "AIN", b) "FA", c) "KAF", d) "LAM", e) "MEEM", f) "NOON", g) "WAW", and h) "HA".