A DISCRETE HIDDEN MARKOV MODEL FOR THE RECOGNITION OF HANDWRITTEN FARSI WORDS

PUNTIS JIFROODIAN-HAGHIGHI

A THESIS

IN

THE DEPARTMENT

OF

COMPUTER SCIENCE and SOFTWARE ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE

CONCORDIA UNIVERSITY

MONTREAL, QUEBEC, CANADA

January 2010

Your file Votre référence
ISBN: 978-0-494-67130-6
Our file Notre référence
ISBN: 978-0-494-67130-6

ABSTRACT

A DISCRETE HIDDEN MARKOV MODEL FOR THE RECOGNITION OF HANDWRITTEN FARSI WORDS

PUNTIS JIFROODIAN-HAGHIGHI

Handwriting recognition systems (HRS) have been researched for more than 50 years. Designing a system to recognize specific words in a handwritten clean document is still a difficult task and the challenge is to achieve a high recognition rate. Previously, most of the research in the handwriting recognition domain was conducted on Chinese and Latin languages, while recently more people have shown an interest in the Indo-Iranian script recognition systems.

In this thesis, we present an automatic handwriting recognition system for Farsi words. The system was trained, validated and tested on the CENPARMI Farsi Dataset, which was gathered during this research. CENPARMI's Farsi Dataset is unique in terms of its huge number of images (432,357 combined grayscale and binary) , inclusion of all possible handwriting types (Dates, Words, Isolated Characters, Isolated Digits, Numeral Strings, Special Symbols, Documents), the variety of cursive styles, the number of writers (400) and the exclusive participation of Native Farsi speakers in the gathering of data.

The words were first preprocessed. Concavity and Distribution features were extracted and the codebook was calculated by the vector quantization method. A Discrete Hidden Markov Model was chosen as the classifier because of the cursive nature of the Farsi script. Finally, encouraging recognition rates of 98.76% and 96.02% have been obtained for the Training and Testing sets, respectively.

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to my Supervisor, Dr. Ching Y. Suen for his valuable guidance, great support and encouragement.

I am grateful to people whose studies have contributed to my thesis. Among them is Paulo Rodrigo Cavalin, who helped me a lot to learn the Hidden Markov Models in theory and in practice.

Special thanks should go to everyone else in the Center for Pattern Recognition and Machine Intelligence (CENPARMI) at Concordia University. Specially, Mr. Nicola Nobile, the research manager of the CENPARMI, for his technical assistance and for his help in creating the CENPARMI Farsi Dataset. Also my special thanks to Ms. Shira Katz for her editing and proofreading.

Last but not least, I am indebted to my parents who brought me up with endless love to believe in myself and to them I owe everything. Their care, understanding and encouragement enabled me to finish this thesis.

To

Farahdokht Alaei, My dear Mother

and

Mohammad Jifroodian Haghighi, My dear Father

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Introduction

In the twenty-first century, the abilities of robots and automated systems have been used to ease human affairs. Nowadays, customers want to have their demands fulfilled as soon as possible. Employers want to hire fewer employees. For instance, customers may want to cash a check at midnight while there is no bank clerk available. Therefore, the bank may need a signature recognition system to validate the check's signature and to cash the check for the customer. A post office needs an automatic recognition system which can separate the letters destined for different cities. During a war, military authorities may need the help of a system to read threatening letters being sent to them as soon as possible when they have no access to a translator.

These are a few examples of handwriting recognition, which is a branch of a practical science called Computer Vision. In other words, handwriting recognition is the automatic way of reading a human's handwriting.

Handwriting recognition is a combination of three fields of study: Image Processing, Pattern Recognition and Machine learning. The handwritten words, sentences or texts are considered as images for the recognition system and should be processed before they can be sent to the pattern recognition phase. In this phase, they are compared with the existing patterns which the system has already learned and the recognition system chooses the best match for the current image. In other words, the system recognizes the images.

In recent years, a lot of research has been conducted on the recognition of Latin and Chinese scripts. Unfortunately, fewer researchers have shown an interest in conducting research in Indo-Iranian languages and the reason has been due to less governmental support of such handwriting recognition projects.

Farsi (Persian) is a branch of the Indo-Iranian languages and is similar to all of them in terms of letters, words and grammar. Therefore, a Farsi word recognition system with some small changes can also be used to recognize the words in all the Indo-Iranian languages. This research presents a Farsi handwriting word recognition system to recognize a small scale dictionary of 73 Farsi words, which are most commonly used in the fields of Finance and Measurement. The proposed method is suitable for limited vocabulary applications such as reading and recognizing the postal codes and the names of the cities.

1.2 Literature Survey

A lot of research has been conducted on the recognition of Latin and Chinese scripts [60-70]. Recently more people have shown interest in Indo-Iranian Languages script recognition systems [44-59]. To be able to train, validate and test our Farsi Handwriting word recognition system, we needed a Dataset with a sufficient number of images. Some Farsi Datasets will be described in Chapter 2, Section 2.3.1 [5, 7-10]. The

Dataset which we have designed and collected is a multi-purpose Handwritten Dataset, which includes all possible script types of the Farsi language [19]. Our Dataset will be compared with the other existing Datasets in Chapter 2.

For a small size lexicon such as 73 words, the holistic method as described in [20] is used. Among all of the classification models Hidden Markov Models (HMM) were chosen because of the connected nature of the Farsi script. HMM systems stochastically model sequences of variable lengths and cope with nonlinear distortions along one direction [21].

Numerous research studies have been conducted on handwritten word recognition using HMM [11, 20, 40- 42, 46-49, 56]. Some of these studies are briefly described below. The following research has helped us to find the correct future research directions.

In M. Dehghan et al.'s research [11] a holistic Farsi word recognition system was proposed. This system used Discrete Hidden Markov Models (HMM) as the classifier and a Self-Organizing Feature Map (SOFM) to preserve the neighbourhood information for smoothing the observation probability distributions of trained HMMs. Some pre-processing algorithms were applied to the word images to normalize the words' stroke widths, centralize the baselines and track the words' chaincodes.

The feature extraction method used was based on the word image contour. Horizontal and vertical sliding windows were then applied over the word image. In each frame, a local histogram of the contour chain codes was calculated. One of the four possible slopes (0, 45, 90, 135 degrees) was assumed for the contour direction. Therefore, the histogram in each zone had four components and each frame was represented as a 20 elements feature vector. SOFM was used as a clustering algorithm in the discrete HMM. The Baum-Welch's algorithm was used as the HMM training algorithm.

A designed topology for the HMM used in this recognition system is shown in Figure 1-1.

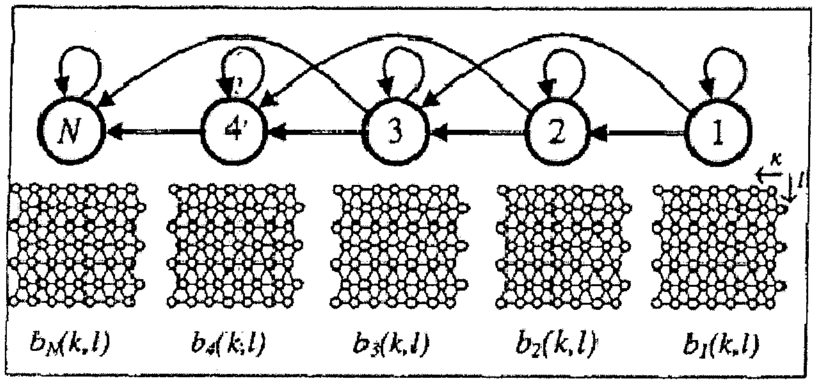


Figure 1-1: Right to left, HMM topology. Each state has transitions with lengths 1 and 2. N shows the maximum number of states in the topology [11].

Also, in this study, more than 17,000 images of 198 city names of Iran were chosen as the dataset. Sixty percent of the images were used as the training set and the rest were used as the testing set.

The recognition results, after smoothing the HMM with different smoothing parameters, are shown in Table 1-1.

Top-*n*	1	2	5	10	20
Rec. Rate($Sf=10^{-1}$)	62.96	74.13	84.40	89.67	94.56
Rec. Rate($Sf=10^{-2}$)	**65.05**	**76.09**	**86.08**	**90.83**	**95.00**
Rec. Rate($Sf=10^{-3}$)	62.68	74.04	85.69	91.35	94.89
Rec. Rate($Sf=10^{-4}$)	58.75	71.66	84.29	90.46	94.72

Table 1-1: Recognition results with different smoothing parameters [11].

In the best case with $SF = 10^{-4}$, 94.72% of the testing set' s images were found between the first 20 recognized words. With $SF = 10^{-1}$, the best recognition rate for the first correctly recognized word was achieved at 62.96 %.

In M. Pechwitz et al.'s research, an offline handwriting word recognition system was proposed for the Arabic language. The system used a one dimensional Semi-Continuous HMM (SCHMM) as the classifier and sliding windows to extract the features. The system was trained and tested on the IFN/ENIT database [39]. Basic preprocessing tasks such as image binarization, word segmentation and noise reduction were performed at the database development level. The window slided from right to left and the features were extracted and concatenated from three successive columns each time. Baseline dependant features were extracted. Two Baselines including the Upper and Lower Baselines were considered for each image. The Upper Baseline was fixed at the 40% of the distance between the baseline and the image's top. The length normalization of words was applied based on the number of characters in each word. The Karhunen-Loève transform was then applied to the feature vector to reduce its dimension. The SCHMM had 7 states per character and the following three transitions: A self-transition, a transition to the next state and the next two states. Finally the recognition result was increased to 89% [40].

In Y. Kessentini et al.'s research, a multi-stream HMM-based approach was proposed for offline multi-script handwriting word recognition. The purpose of the research was to design and implement a handwriting recognition system which could be used for the recognition of different scripts. The model was composed of some sub-units which could perform the recombination of input streams such as characters or words. Some anchor points were defined between the sub-units [41]. The word images were preprocessed to prepare them for the feature extraction. The preprocessing stage included Normalization, Contour Smoothing and Baseline Detection. To build the feature vector, the word image was divided into vertical overlapping sliding windows. Depending on the nature of the script, the sliding windows were shifted from right to left or left to right. The Main features used for the system were Contour-related and Density Features. The Density Features were calculated based on the density of the foreground pixels and the Contour Features were calculated based on the word image's upper and lower contours. For each sliding window, the upper and lower contour points were calculated and the corresponding Freeman directions are determined. Finally, the direction points were accumulated in the direction histogram and each point was classified as Lower Contour, Interior Contour, Upper Contour or No Point Found. To train the multi-stream Hidden Markov Model, two stages were considered: First, the estimation of the model's stream component parameters and second, the estimation of the proper stream exponents. The system recognized the words by concatenation of its characters. Twenty-six character models were considered for the Latin script and 159 character models were considered for the Arabic script, since the Arabic script has more variables. The recognition system was trained and tested on the IRONOFF and IFN/ENIT databases. The results were announced for different lexicon sizes which are shown in Table 1-2 [41].

Lexicon Size	10	100	196	500
Recognition Rate	95.6	84.2	79.6	76.2

Table 1-2: The announced results for four different lexicon sizes.

In S. Alma'adeed et al.'s research, a HMM recognition system was used for Arabic handwritten words. Their recognition system was able to recognize the dataset written by 100 different writers.

Their handwriting recognition system was divided into three sections: preprocessing, recognition and post-processing. In the preprocessing stage, the stroke width, the slope and the height of the letters were normalized. Only 55 classes were chosen instead of 120 classes to represent different letter shapes in Arabic, because the system could only recognize the 55 words which were defined in the lexicon. A Modified Viterbi's Algorithm was (MVA) used in the recognition phase [64].The database was segmented into Training and Testing sets. Two thirds of the data was used as the Training set and the rest as the Testing set. A codebook size of seventy was chosen empirically. Finally, a recognition rate of 45% was found on the Testing set.

1.3 Our Approach

To design a handwriting recognition system with a high performance, we have to train it with various handwriting styles. Therefore, as the first step, we gathered a Farsi Dataset, which could be used as a reference point to measure the performance of not only a handwriting recognition system, but also word spotting systems [19].

The next step was to design and implement the handwriting recognition system. The design cycle is shown in Figure 1-2.

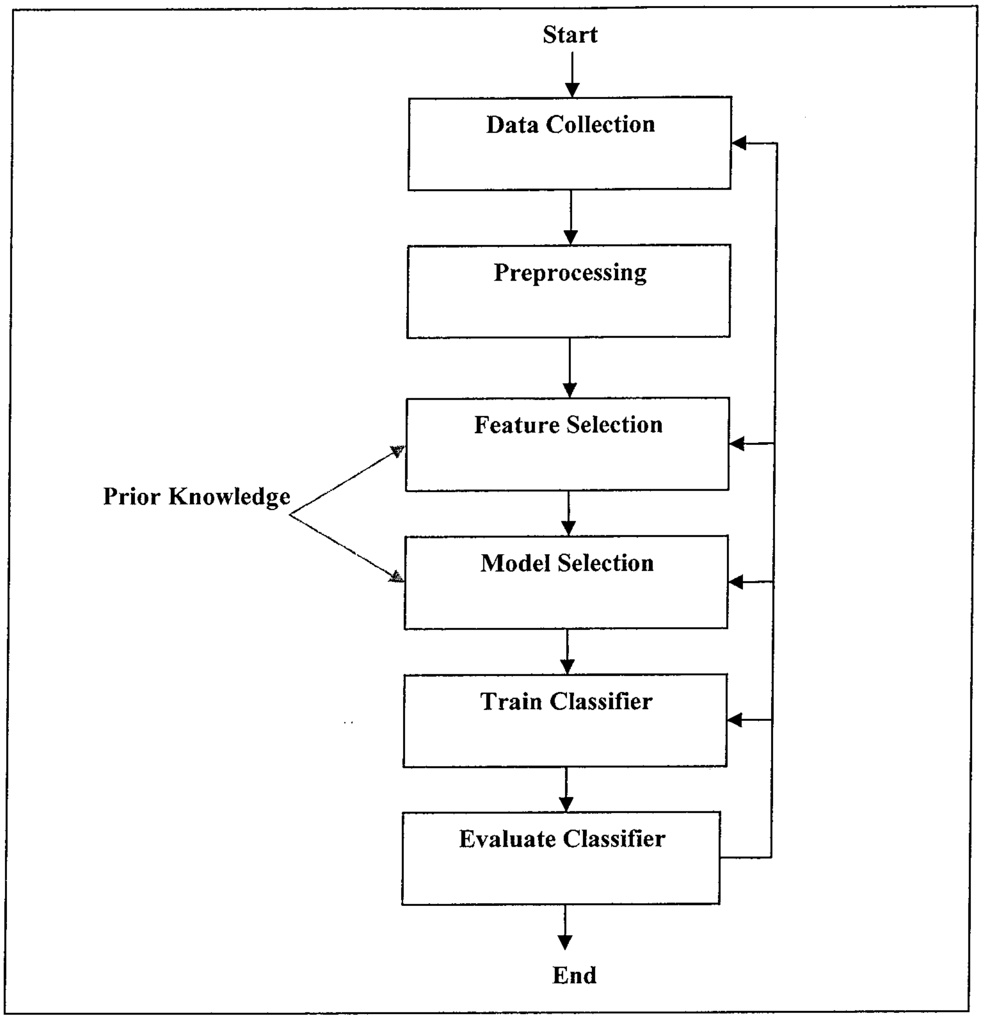


Figure 1-2 : Steps that should be completed to design a recognition system [71].

For implementation of the recognition system, we chose Matlab programming language, which gives access to an Image Processing Toolbox. This toolbox makes the design and implementation of the system easier and faster [13].

1.4 Our Goal

Our goal is to build a handwriting recognition system using a Hidden Markov Models (HMM) classifier. HMM is a complicated classifier and it is not as easy to work with as other classifiers such as Support Vector Machines (SVM). But since the best results in the research have been gained by using HMM in handwriting recognition, especially the ones using the holistic method to recognize the words, it was chosen for our research.

This system is designed to be part of a higher level recognition system which can find the lexicon's words in our Dataset and pass it through our recognition system. We ran several experiments while changing the HMM parameters and testing different features. Finally, encouraging recognition rates of 98.76% and 96.02% have been obtained for the Training and Testing sets, respectively. Our designed system is reliable, robust to noise and is reasonably fast.

1.5 Organization of the Thesis

Chapters 2 and 3 present an overview of the Farsi language and the Farsi Dataset, the Recognition system and a description of how we gathered the data and preprocessed it to make it ready for our recognition system.

Chapters 4, 5 and 6 describe the recognition procedure and the steps which were fulfilled to reach the recognition result, while Chapter 7 shows the results and error analysis. Conclusions and directions of possible future work are discussed in Chapter 8.

Chapter 2: CENPARMI Farsi Dataset

The CENPARMI Farsi Dataset is structurally organized in the same way as the other CENPARMI Indo-Iranian Datasets. The main purpose of this thesis has been to design a Dataset which does not only facilitate the development and evaluation of the Farsi recognition systems but can also be used to compare the performance of different recognition systems. In this chapter, the Indo-Iranian languages and Farsi Dataset are described in detail.

2.1 Indo-Iranian Languages

Indo-Iranian languages are a branch of the Indo-European languages. This language family is widely spoken in central and southern Asia, in Iran, Afghanistan, Iraq, Pakistan, Turkey, India and Bangladesh. This language family is divided into two language sub-families, known as the INDIC sub-family and the IRANIAN sub-families.
Table 2-1 shows IRANIAN and INDIC sub-families of Indo-Iranian languages and some instances of their member languages [2].

INDIC	IRANIAN		
Sanskrit	Avestan	Old Persian (Farsi)	Scythian
Prakrit	Pashto	Persian (Farsi)	
Pali		Kurdish	
Gujarati		Ossetic	
Marathi		Baluchi	
Hindustani			
Hindi			
Urdu			
Benagli			
Bihari			
Sindhi			
Bhili			
Rajasthani			
Panjabi			
Pahari			

Table 2-1: IRANIAN and INDIC sub-families of Indo-Iranian languages and some instances of their member languages.

Indo-Iranian languages are written from right to left in a cursive way. In Table 2-2, a comparison is shown between the isolated characters in Farsi and five other Indo-Iranian languages.

As shown in Table 2-2, Pashto with its 46 letters has the largest number of letters among these Indo-Iranian languages. Other languages shown in the table can be considered as a subset of Pashto's letters, except for a few letters that are found in some languages and can not be found in Pashto. Those letters are circled with a red color in Table 2-2 (Circled letters: 5, 31, 40, 42, and 48).

	Phonetics	Pashto	Urdu	Farsi	Dari	Arabic
	aa	ا	ا	ا آ	ا آ	ا ء
	b	ب	ب	ب	ب	ب
	p	پ	پ	پ	پ	
	t	ت	ت	ت	ت	ت
	tt	ټ	ٹ			
	s	ث	ث	ث	ث	ث
	j	ج	ج	ج	ج	ج
	ch	چ	چ	چ	چ	
	dz	ځ				
	sch	څ				
	h	ح	ح	ح	ح	ح
	kh	خ	خ	خ	خ	خ
	d	د	د	د	د	د
	dd	ډ	ڈ			

Table 2-2: Comparison between Farsi isolated letters and five other Indo-Iranian languages (Part a).

	Phonetics	Pashto	Urdu	Farsi	Dari	Arabic
	aa	ا	ا	ا آ	ا آ	ا ء
	b	ب	ب	ب	ب	ب
	p	پ	پ	پ	پ	
	t	ت	ت	ت	ت	ت
	tt	ټ	ٹ			
	s	ث	ث	ث	ث	ث
	j	ج	ج	ج	ج	ج
	ch	چ	چ	چ	چ	
	dz	ځ				
	sch	څ				
	h	ح	ح	ح	ح	ح
	kh	خ	خ	خ	خ	خ
	d	د	د	د	د	د
	dd	ډ	ڈ			

Table 2-2: Comparison between Farsi isolated letters and five other Indo-Iranian languages (Part b).

	Phonetics	Pashto	Urdu	Farsi	Dari	Arabic
	gh	غ	غ	غ	غ	غ
	f	ف	ف	ف	ف	ف
	v					
	gh	ق	ق	ق	ق	ق
	k	ك	ک	ک	ک	ک
	g	ګ	گ	گ	گ	
	L	ل	ل	ل	ل	ل
	m	م	م	م	م	م
	n	ن	ن	ن	ن	ن
	nn	ڼ				
	h	ه	ه	ه	ه	ه
			ھ			
	w	و	و	و	و	و
	v					

Table 2-2: Comparison between Farsi isolated letters and five other Indo-Iranian languages (Part c).

	Phonetics	Pashto	Urdu	Farsi	Dari	Arabic
	ay	ی	ی	ی	ی	ی
	i	ي	ے			
	e	ې				
	əy	ۍ				
	ə	ئ	ء			
	ye					

Table 2-2: Comparison between Farsi isolated letters and five other Indo-Iranian languages (Part d).

2.2 Overview of the Farsi Language

In this section, we will describe more about the Farsi Language and the characteristics of its scripts.

2.2.1 Farsi Language

Farsi (Persian) is widely spoken in Iran, Afghanistan, Tajikistan, Uzbekistan, Bahrain and the surrounding areas, as shown in Figure 2-1.

Farsi has been a medium for literary and scientific contributions to the Islamic world. For five centuries, prior to British colonization, Farsi was widely used as a second language in the south western region of the Asian continent. It took prominence as the language of culture and education in several Muslim courts in southern Asia and became the "official language" under the Mughal emperors [1].

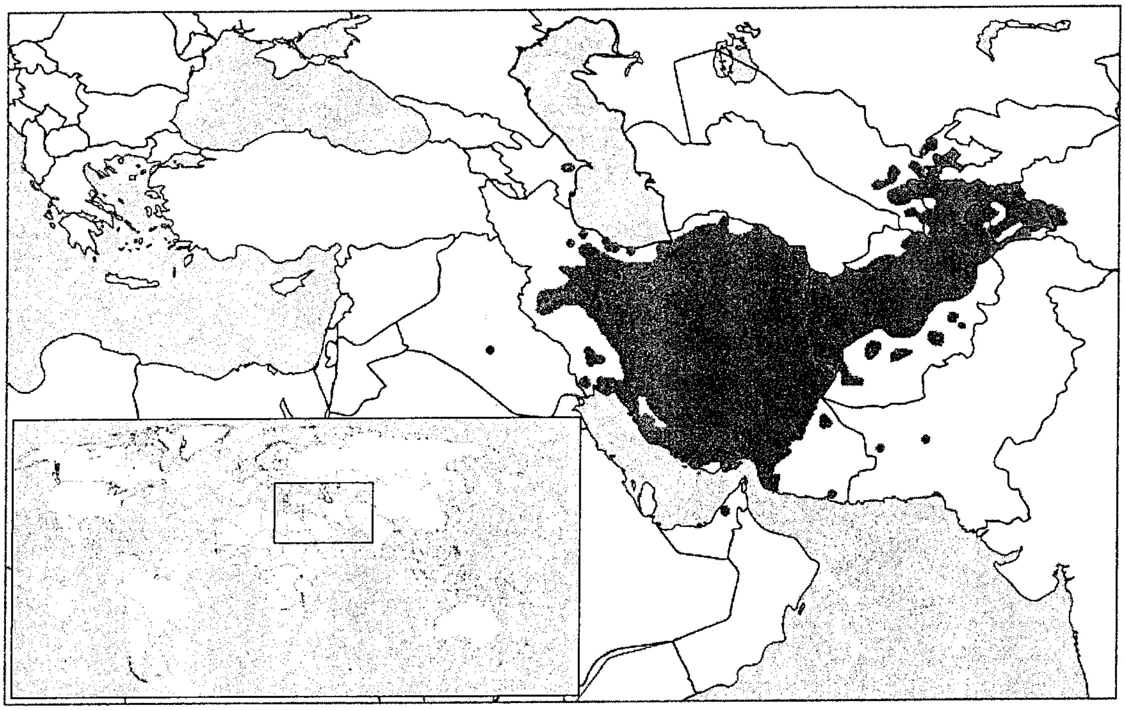


Figure 2-1: Areas shown in red are Farsi-speaking areas in Asia.

2.2.2 Farsi Isolated Characters: Main Characteristics

The Farsi language has 32 letters, which are written from right to left in a cursive way. This script does not have upper or lower case letters but there are some diacritics which differentiate similar words in terms of pronunciation and meaning. As shown in Table 2-2 (Section 2-1), some Farsi letters have one, two or three dots. Depending on the location of the letter within the word, every Farsi letter can have a maximum of four different shapes (Isolated (Free), Initial, Middle (Medial) or Final shape). Table 2-3: shows the Farsi characters with their four possible shapes [21].

Character	Isolated	Initial	Middle	Final	Transliteration
Alef	آ	آ	ـا	ـا	a
Beh	ب	بـ	ـبـ	ـب	b
Peh	پ	پـ	ـپـ	ـپ	p
The	ت	تـ	ـتـ	ـت	t
Theh	ث	ثـ	ـثـ	ـث	Th
Jeem	ج	جـ	ـجـ	ـج	j
Cheh	چ	چـ	ـچـ	ـچ	ch
Heh	ح	حـ	ـحـ	ـح	h
Kheh	خ	خـ	ـخـ	ـخ	kh
Dal	د	د	ـد	ـد	d
Thal	ذ	ذ	ـذ	ـذ	th
Reh	ر	ر	ـر	ـر	r
Zeh	ز	ز	ـز	ـز	z
Zheh	ژ	ژ	ـژ	ـژ	zh
Seen	س	سـ	ـسـ	ـس	s
Sheen	ش	شـ	ـشـ	ـش	sh
Sad	ص	صـ	ـصـ	ـص	s
Zad	ض	ضـ	ـضـ	ـض	th
Tah	ط	طـ	ـطـ	ـط	t
Zah	ظ	ظـ	ـظـ	ـظ	z
Ain	ع	عـ	ـعـ	ـع	a
Ghain	غ	غـ	ـغـ	ـغ	gh
Feh	ف	فـ	ـفـ	ـف	f
Ghaf	ق	قـ	ـقـ	ـق	gh
Kaf	ک	کـ	ـکـ	ـک	k
Gaf	گ	گـ	ـگـ	ـگ	g
Lam	ل	لـ	ـلـ	ـل	l
Meem	م	مـ	ـمـ	ـم	m
Noon	ن	نـ	ـنـ	ـن	n
Waw	و	و	ـو	ـو	v
Heh	ه	هـ	ـهـ	ـه	h
Yeh	ی	یـ	ـیـ	ـی	y

Table 2-3: Farsi characters with their four possible shapes [21].

Figure 2-2 shows the anatomy of the Farsi letters. [3] These details can be used to design the structural features in Handwriting Recognition.

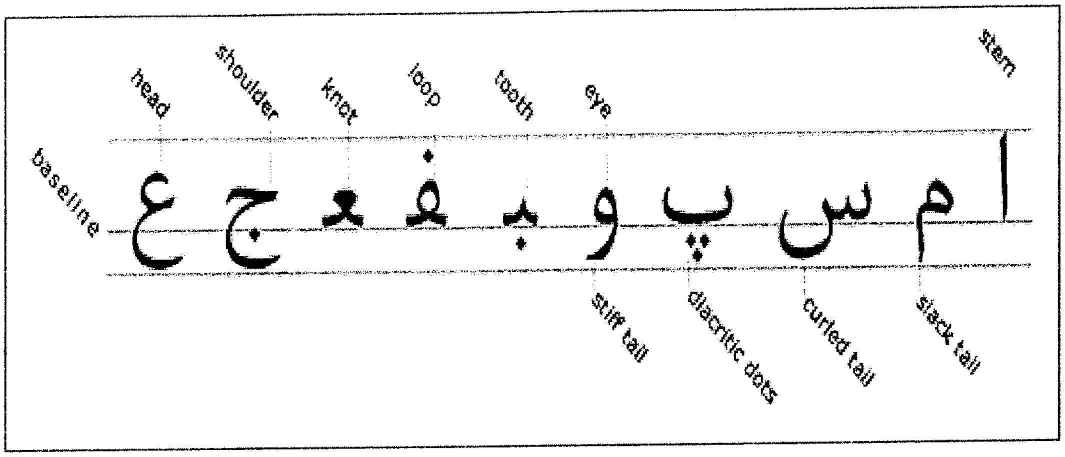


Figure 2-2: Anatomy of Farsi letters [3].

In Farsi, the same words can be written in different shapes. Some variants are depicted in Figure 2-3 and Figure 2-4. These variants have been written by different writers and have been collected in our Dataset.

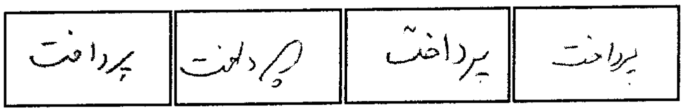


Figure 2-3: Four variants of the word "Payment" in Farsi.

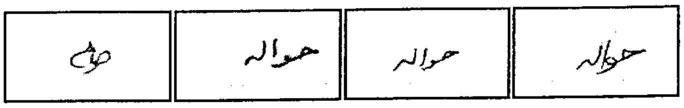


Figure 2-4: Four variants of the word "Money Order" which is equivalent to حواله (Havaleh) in Farsi.

Concentrating on these shape differences can help us to design a more accurate recognition system and to achieve a better recognition result.

2.2.3 Farsi Isolated Digits: Main Characteristics

The decimal numbering system originated in India (Devanagari ०१२३...)and was subsequently adopted in Arabic languages with a different appearance (Arabic-Indic ٠١٢٣...). The Europeans adopted decimal numbers from the Arabic languages, although once again the forms of the digits changed greatly (European numerals 0, 1, 2, 3, etc).

In addition, there are two main variants of the Arabic-Indic digits. Those used in Iran, Pakistan and India (in Farsi, Dari and Urdu languages) are called Eastern Arabic-Indic numerals and those used in other parts of the Arab world are called Arabic-Indic numerals [4].

Table 2-4 shows a comparison between Arabic-Indic and Eastern Arabic-Indic numerals.

European Numerals	Arabic-Indic NUmerals	Eastern Arabic-Indic Numerals
0	٠	۰
1	١	۱
2'	٢	۲
3	٣	۳
4	٤	۴
5	٥	۵
6	٦	۶
7	٧	۷
8	٨	۸
9	٩	۹

Table 2-4: : A Comparison between Arabic-Indic and Eastern Arabic-Indic Numerals.

Arabic-Indic digits may be written differently for the numbers 0, 2, 4,5,6,7, based on the language. For instance, the numbers 0, 4, 5 and 6 are written differently in Farsi from Arabic. Also, for handwritten numerals, in some Arabic countries an Arabic "three" is written like a Farsi "two". In Table 2-5, numerals are compared in six Indo-Iranian languages. As shown, they may be different in the cases of 0, 4, 5, and 6.

European Numerals	Urdu	Pashtu	Dari	Farsi	Arabic
0	۰	۰	۰	۰	٠
1	۱	۱	۱	۱	١
2	۲	۲	۲	۲	٢
3	۳	۳	۳	۳	٣
4	۴	۴	۴	۴	٤
5	۵	۵	۵	۵	٥
6	۶	٦	۶	۶	٦
7	۷	۷	۷	۷	٧
8	۸	۸	۸	۸	٨
9	۹	۹	۹	۹	٩

Table 2-5: Comparison between the digits in six Indo-Iranian languages.

2.3 Farsi Dataset

The Centre for Pattern Recognition and Machine Intelligence (CENPARMI) has developed a Farsi Dataset that can be used as a reference point to measure the performance of not only handwriting recognition systems, but also word spotting systems [19]. CENPARMI's Farsi Dataset is unique in terms of its huge number of images (432,357 combined grayscale and binary) , inclusion of all possible handwriting types (Freestyle Dates, Words, Isolated Characters, Isolated Digits, Numeral Strings, Special Symbols, Documents), the variety of cursive styles, the number of writers (400) and the exclusive participation of native Farsi speakers in the gathering of data. CENPARMI's Farsi Dataset also follows the same structural format of the other CENPARMI's other Indo-Iranian Languages Dataset. For this thesis the selection of Farsi words was based on the highest frequency of their appearances in the selected financial documents. This Dataset also includes Farsi Texts (Documents) and the selected texts include the Dataset's words, which can be used in word spotting systems.

The Dataset in this thesis is divided into Grouped and Ungrouped subsets, which will give the user the flexibility of whether or not to use CENPARMI's pre-divided (Grouped) Dataset (60% of the images are used as the Training set, 20% as the Validation set and the rest as the Testing set) .

2.3.1 Related Work

In order to get a good recognition result, experiments should be conducted on a structured, high quality Dataset. There have been a few developments in recent years towards this goal, using the Farsi language.

Because of the fact that the Farsi language is a branch of the Indo-Iranian languages and is a cursive language, some researchers have been misled into using Arabic or other Datasets to evaluate the performance of their Farsi handwriting recognizers. However, according to the comparisons we have made in Table 2-2 ,

Table 2-4 and Table 2-5, the Farsi script has significant differences with other Indo-Iranian languages. Therefore, the most accurate recognition result can be gained only when we use the proper Dataset for the language. Compared with the Arabic language, Farsi has four more letters in its alphabet (پ،چ،ژ،گ).

According to a research study we conducted at CENPARMI, the handwritten Farsi digit ۲ is considered as digit 3 in Arabic, while it is always considered as digit 2 in Farsi [6].

To the best of our knowledge, no multipurpose Farsi Dataset that can cover all script types of that language has been published to date. A standard dataset for recognition of handwritten digits, numerical strings, legal amounts, letters and dates in the Farsi Language has been published by CENPARMI [7] .This Dataset includes 18,000 images of Farsi digits and 11,900 images of Farsi characters from 171 writers.
In 2006, an isolated character and digit Dataset for the Farsi language was presented by the Pattern Recognition and Image Processing Laboratory of the Amirkabir University of Technology in Iran. It consists of 52,380 Grayscale images of characters and 17,740 numerals [5].

In 2007, a Dataset of Farsi handwritten digits was presented by the Tarbiat Modarres University of Iran [8]. This Dataset consists of 102,352 binary images of Farsi digits with 200 dpi resolution.

In 2008, IFN FARSI Dataset of city names was presented by Iran's Semnan University, the Braunschweig Technical University of Germany and the Amirkabir University of Iran. This Dataset consists of 7,271 binary images of 1,080 Iranian Province/City names collected from 600 writers [9]. IAUT/PHCN is another Dataset which was presented in 2008 by the Islamic Azad University of Iran. It contains 32,400 binary images of the Iranian city names, collected from 380 writers [10]. Table 2-6 compares CENPARMI Farsi Dataset with the mentioned Farsi Datasets.

Dataset Presented By	Year of Presentation	Number of Images	Type of Script						
			Date	Digit	Numeral String	Character	Word	Special Symbols	Text
CENPARMI[7]	2006	C: 11,900 D: 18,000	X	X	X	X			
Amirkabir University[5]	2006	C : 52,380 D: 17,740		X		X			
Tarbiat Modarres University[8]	2007	D: 102,352		X					
Semnan , Braunschweig and Amirkabir Universities[9]	2008	W: 7,271					X		
Islamic Azad University[10]	2008	W: 32,400					X		
CENPARMI[18]	2009	432,357	X	X	X	X	X	X	X

Table 2-6: A comparison between CENPARMI's Farsi Dataset and existing Farsi Datasets. C, D and W stand for number of Characters, number of Digits and number of Words, respectively.

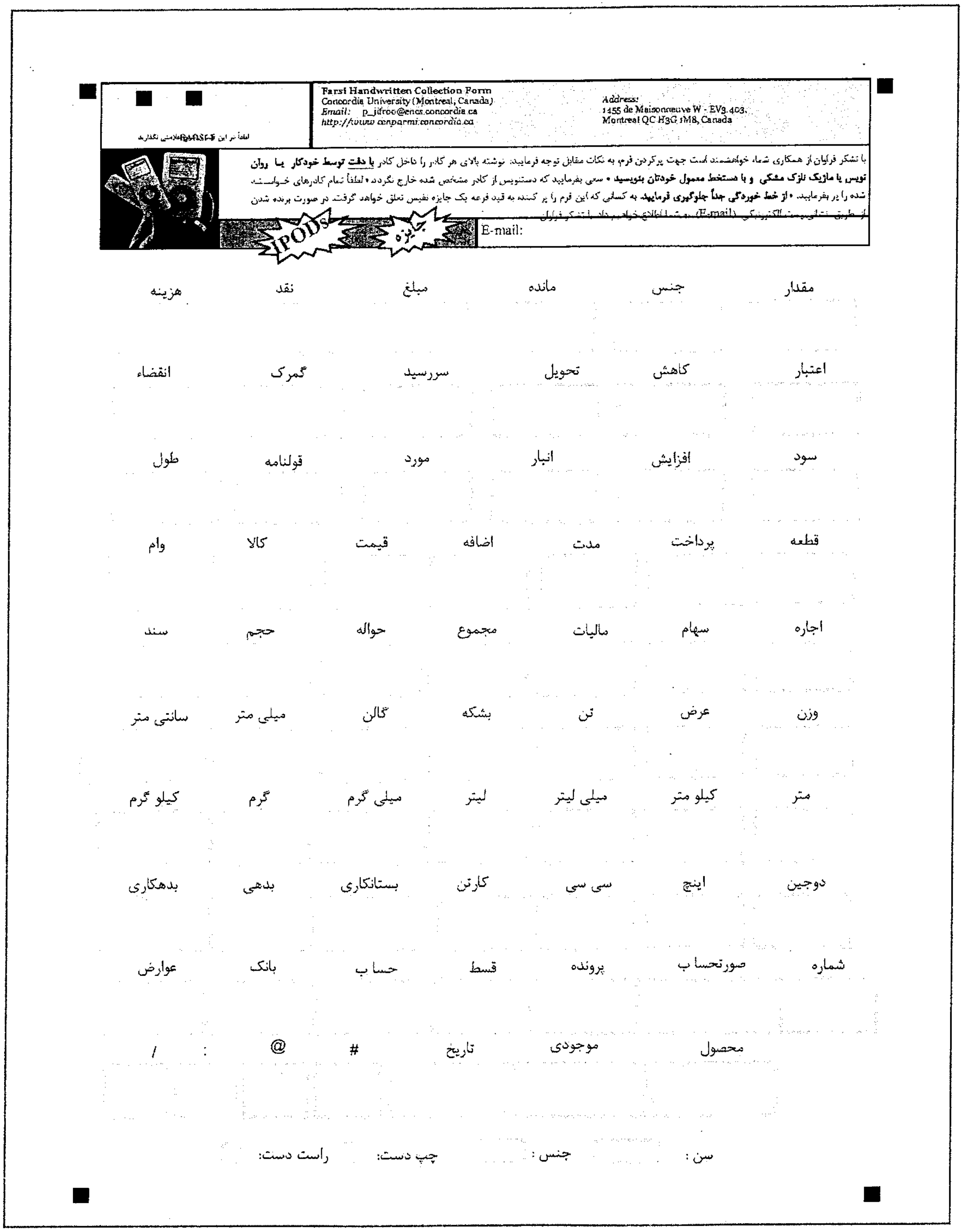
2.3.2 Problems Faced in Data Collection

Although Montreal is a multicultural city and help was obtained from Iranian students and Concordia University's professors to gather the Farsi forms, it was not easy to find Iranian people, especially the ones who would be interested in filling out the forms. Therefore, we decided to send some forms to other cities such as Vancouver, Toronto and even Tehran, to be sure that the forms would be filled out by native Farsi speakers.

2.3.3 Data-Collection Process

To collect the data we designed a two-pages collection form. The first page includes the Words and Special Symbols while the second page includes the Date, Isolated Digits, Numeral Strings and Isolated Characters. Figure 2-5 and Figure 2-7 show the two pages of the empty designed form while Figure 2-6 and Figure 2-8 represent the same form when both pages are filled out.

At the top of each form, some information has been gathered from each writer, such as gender, handwriting orientation and range of age. This information can be studied in the psychology of handwriting.



لطفاً در این قسمت هیچ علامتی نگذارید

Farsi Handwritten Collection Form
Concordia University (Montreal, Canada)
Email: p_jifroo@encs.concordia.ca
http://www.cenparmi.concordia.ca

Address:
1455 de Maisonneuve W - EV3.403,
Montreal QC H3G 1M8, Canada

با تشکر فراوان از همکاری شما، خواهشمند است جهت پرکردن فرم به نکات مقابل توجه فرمایید. نوشته بالای هر کادر را داخل کادر **با دقت توسط خودکار یا روان نویس یا ماژیک نازک مشکی و با دستخط معمول خودتان بنویسید** • سعی بفرمایید که دستنویس از کادر مشخص شده خارج نگردد • لطفاً تمام کادرهای خواسته شده را پر بفرمایید. • **از خط خوردگی جداً جلوگیری فرمایید.** به کسانی که این فرم را پر کننده به قید قرعه یک جایزه نفیس تعلق خواهد گرفت. در صورت برنده شدن از طریق نامه الکترونیکی (E-mail) به شما اطلاع خواهیم داد. با تشکر فراوان

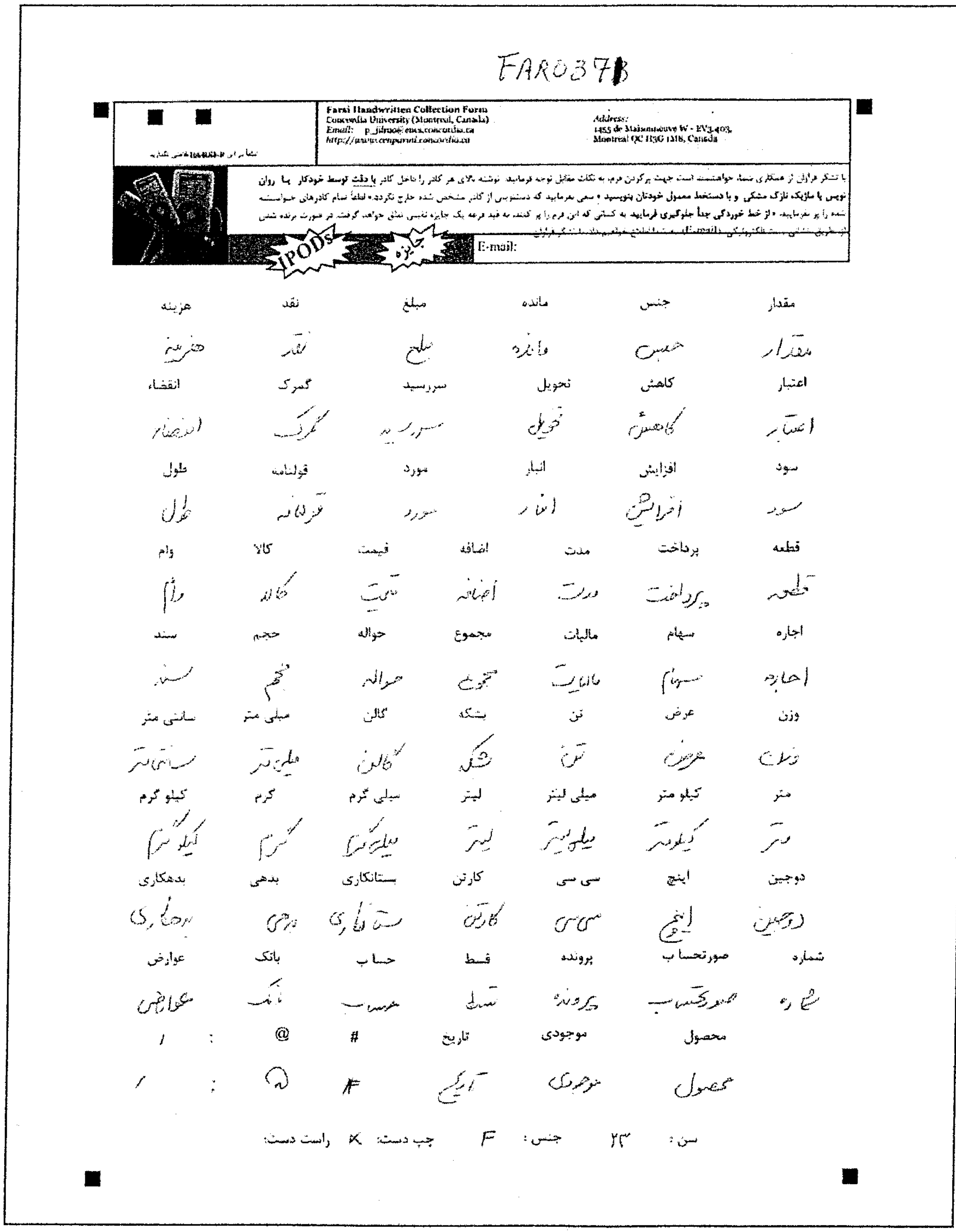
IPODS جایزه

E-mail:

مقدار	جنس	مانده	مبلغ	نقد	هزینه	
اعتبار	کاهش	تحویل	سررسید	گمرک	انقضاء	
سود	افزایش	انبار	مورد	قولنامه	طول	
قطعه	پرداخت	مدت	اضافه	قیمت	کالا	وام
اجاره	سهام	مالیات	مجموع	حواله	حجم	سند
وزن	عرض	تن	بشکه	گالن	میلی متر	سانتی متر
متر	کیلو متر	میلی لیتر	لیتر	میلی گرم	گرم	کیلو گرم
دوجین	اینچ	سی سی	کارتن	بستانکاری	بدهی	بدهکاری
شماره	صورتحسا ب	پرونده	قسط	حسا ب	بانک	عوارض
محصول	موجودی	تاریخ	#	@	:	/

سن : جنس : چپ دست: راست دست:

Figure 2-5: Farsi collection form, Page one, Words and Special Symbols.



FAR037B

Farsi Handwritten Collection Form
Concordia University (Montreal, Canada)
Email: p_jifroo@encs.concordia.ca
http://www.cenparmi.concordia.ca

Address:
1455 de Maisonneuve W - EV3.403,
Montreal QC H3G 1M8, Canada

شماره این فرم [illegible]

با تشکر فراوان از همکاری شما، خواهشمند است جهت پرکردن فرم به نکات مقابل توجه فرمایید. نوشته بالای هر کادر را داخل کادر **با دقت توسط خودکار یا روان نویس یا ماژیک نازک مشکی و با دستخط معمول خودتان بنویسید** ، سعی بفرمایید که دستنویس از کادر مشخص شده خارج نگردد، لطفاً تمام کادرهای خواسته شده را پر بفرمایید ، **از خط خوردگی جداً جلوگیری فرمایید**. به کسانی که این فرم را پر کنند به قید قرعه یک جایزه نفیس نقدی خواهد گرفت. در صورت برنده شدن [illegible]

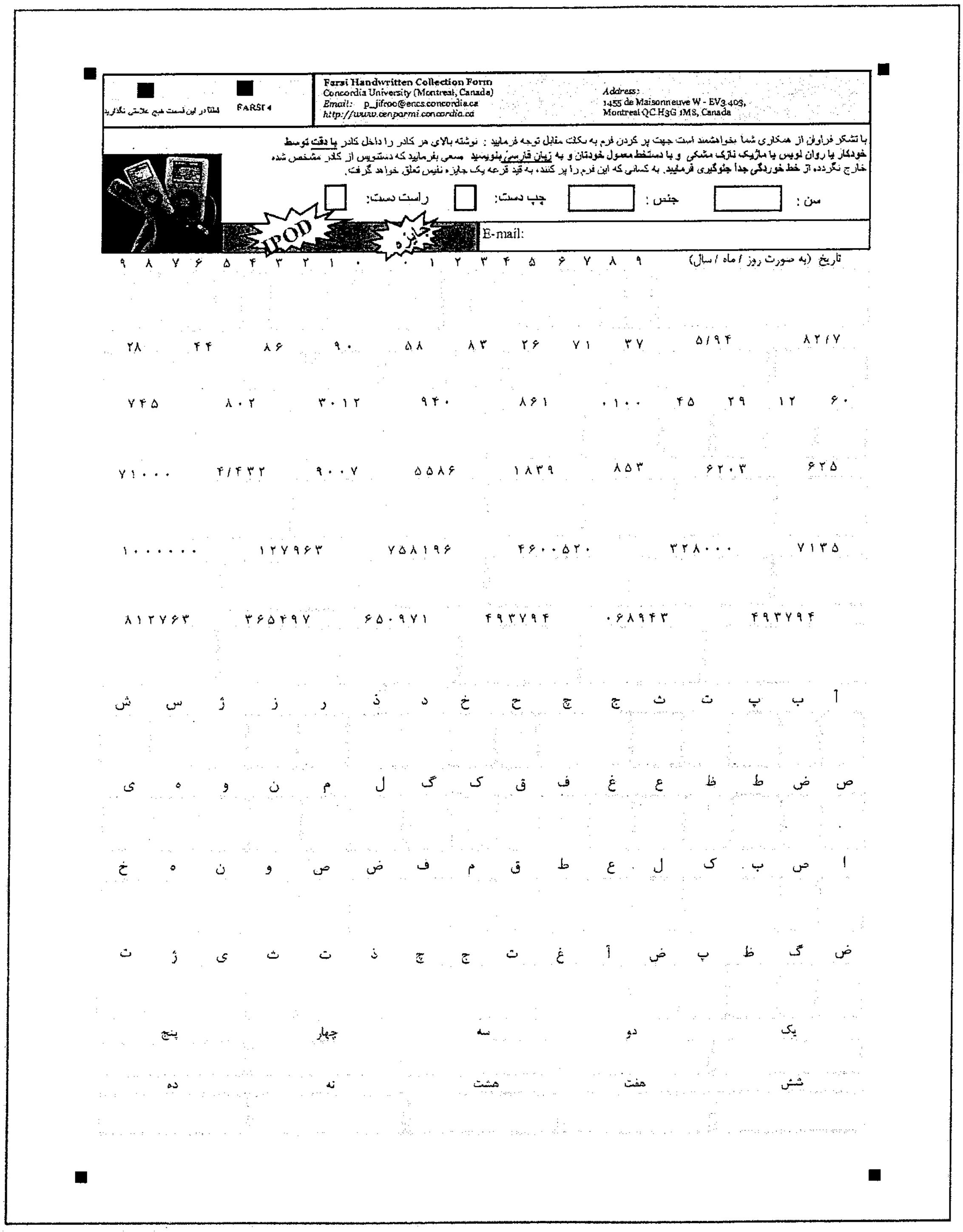
IPODs جایزه

E-mail:

مقدار	جنس	مانده	مبلغ	نقد	هزینه
مقدار	جنس	مانده	مبلغ	نقد	هزینه
اعتبار	کاهش	تحویل	سررسید	گمرک	انقضاء
اعتبار	کاهش	تحویل	سررسید	گمرک	انقضاء
سود	افزایش	انبار	مورد	قولنامه	طول
سود	افزایش	انبار	مورد	قولنامه	طول
قطعه	پرداخت	مدت	اضافه	قیمت	کالا	وام
قطعه	پرداخت	مدت	اضافه	قیمت	کالا	وام
اجاره	سهام	مالیات	مجموع	حواله	حجم	سند
اجاره	سهام	مالیات	مجموع	حواله	حجم	سند
وزن	عرض	تن	بشکه	گالن	میلی متر	سانتی متر
وزن	عرض	تن	بشکه	گالن	میلی متر	سانتی متر
متر	کیلو متر	میلی لیتر	لیتر	سیلی گرم	گرم	کیلو گرم
متر	کیلومتر	میلی لیتر	لیتر	میلی گرم	گرم	کیلو گرم
دوجین	اینچ	سی سی	کارتن	بستانکاری	بدهی	بدهکاری
دوجین	اینچ	سی سی	کارتن	بستانکاری	بدهی	بدهکاری
شماره	صورتحسا ب	پرونده	قسط	حسا ب	بانک	عوارض
شماره	صورتحساب	پرونده	قسط	حساب	بانک	عوارض
محصول	موجودی	تاریخ	#	@	:	/
محصول	موجودی	تاریخ	#	@	:	/

سن : ۲۳ جنس : F چپ دست: K راست دست:

Figure 2-6: Farsi Collection form, page one, collected and labeled as Far 0371.



لطفا در این قسمت هیچ علامتی نگذارید FARSI 4

Farsi Handwritten Collection Form
Concordia University (Montreal, Canada)
Email: p_jifroo@encs.concordia.ca
http://www.cenparmi.concordia.ca

Address:
1455 de Maisonneuve W - EV3.403,
Montreal QC H3G 1M8, Canada

با تشکر فراوان از همکاری شما خواهشمند است جهت پر کردن فرم به نکات مقابل توجه فرمایید : نوشته بالای هر کادر را داخل کادر **با دقت** توسط خودکار یا روان نویس یا ماژیک نازک مشکی و با دستخط معمول خودتان و به زبان فارسی بنویسید سعی بفرمایید که دستنویس از کادر مشخص شده خارج نگردده از خط خوردگی جداً جلوگیری فرمایید. به کسانی که این فرم را پر کنند، به قید قرعه یک جایزه نفیس تعلق خواهد گرفت.

سن : ☐ جنس : ☐ چپ دست: ☐ راست دست: ☐

E-mail:

تاریخ (به صورت روز / ماه / سال) ۹ ۸ ۷ ۶ ۵ ۴ ۳ ۲ ۱ ۰ ۰ ۱ ۲ ۳ ۴ ۵ ۶ ۷ ۸ ۹

۸۲/۷ ۵/۹۴ ۳۷ ۷۱ ۲۶ ۸۳ ۵۸ ۹۰ ۸۶ ۴۴ ۲۸

۶۰ ۱۲ ۲۹ ۴۵ ۰۱۰۰ ۸۶۱ ۹۴۰ ۳۰۱۲ ۸۰۲ ۷۴۵

۶۲۵ ۶۲۰۳ ۸۵۳ ۱۸۳۹ ۵۵۸۶ ۹۰۰۷ ۴/۴۳۲ ۷۱۰۰۰

۷۱۳۵ ۳۲۸۰۰۰ ۴۶۰۰۵۲۰ ۷۵۸۱۹۶ ۱۲۷۹۶۳ ۱۰۰۰۰۰۰

۴۹۳۷۹۴ ۰۶۸۹۴۳ ۴۹۳۷۹۴ ۶۵۰۹۷۱ ۳۶۵۴۹۷ ۸۱۲۷۶۳

آ ب پ ت ث ج چ ح خ د ذ ر ز ژ س ش

ص ض ط ظ ع غ ف ق ک گ ل م ن و ه ی

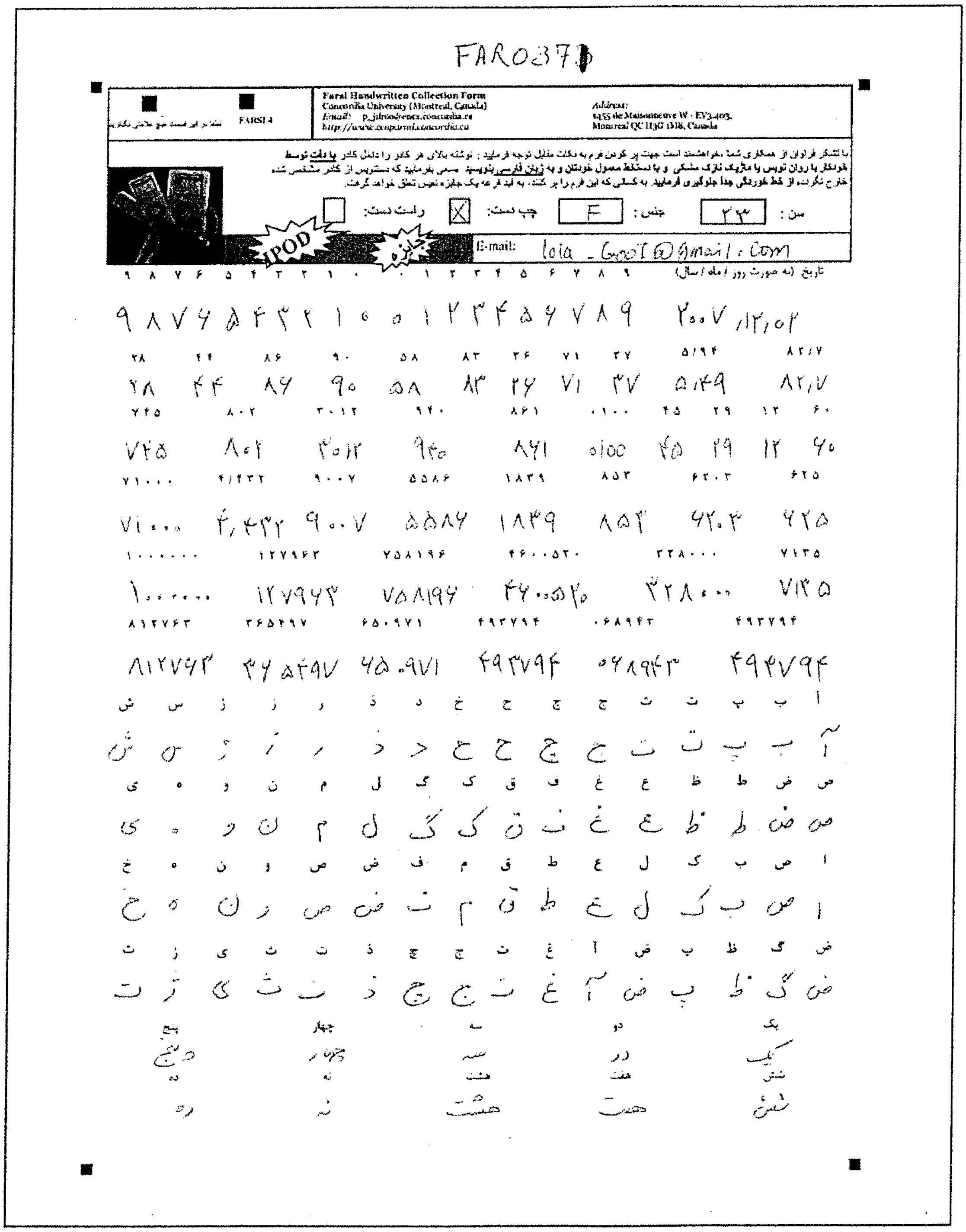
ا ص ب ک ل ع ط ق م ف ض ص و ن ه خ

ض گ ظ پ ض آ غ ت ج چ ذ ت ث ی ژ ت

یک دو سه چهار پنج

شش هفت هشت نه ده

Figure 2-7: Farsi collection form, Page two, Date, Isolated Digits, Numeral Strings, Isolated Characters (letters) and some Words.


FAR0371

لطفا در این قسمت هیچ علامتی نگذارید FARSI 4

Farsi Handwritten Collection Form
Concordia University (Montreal, Canada)
Email: p_jifroodi@encs.concordia.ca
http://www.cenparmi.concordia.ca

Address:
1455 de Maisonneuve W - EV3.403,
Montreal QC H3G 1M8, Canada

با تشکر فراوان از همکاری شما، خواهشمند است جهت پر کردن فرم به نکات مقابل توجه فرمایید : نوشته بالای هر کادر را داخل کادر با دقت توسط خودکار یا روان نویس یا ماژیک نازک مشکی و با دستخط معمول خودتان و به زبان فارسی بنویسید. سعی بفرمایید که دسترس از کادر مشخص شده خارج نگردد. از خط خوردگی جدا جلوگیری فرمایید. به کسانی که این فرم را پر کنند، به قید قرعه یک جایزه نفیس تعلق خواهد گرفت.

سن : ۲۳ جنس : F چپ دست: [✓] راست دست: [ ]

IPOD جایزه

E-mail: lola_GooI@gmail.com

تاریخ (به صورت روز / ماه / سال) ۹ ۸ ۷ ۶ ۵ ۴ ۳ ۲ ۱ ۰ ۰ ۱ ۲ ۳ ۴ ۵ ۶ ۷ ۸ ۹

۲۰۰۷/۱۲/۰۲ ۹ ۸ ۷ ۶ ۵ ۴ ۳ ۲ ۱ ۰ ۰ ۱ ۲ ۳ ۴ ۵ ۶ ۷ ۸ ۹

۲۸ ۴۴ ۸۶ ۹۰ ۵۸ ۸۳ ۲۶ ۷۱ ۳۷ ۵/۹۴ ۸۲/۷

۲۸ ۴۴ ۸۶ ۹۰ ۵۸ ۸۳ ۲۶ ۷۱ ۳۷ ۵,۴۹ ۸۲,۷

۷۴۵ ۸۰۲ ۳۰۱۲ ۹۲۰ ۸۶۱ ۰۱۰۰ ۴۵ ۲۹ ۱۲ ۶۰

۷۴۵ ۸۰۲ ۳۰۱۲ ۹۴۰ ۸۶۱ ۰۱۰۰ ۴۵ ۲۹ ۱۲ ۶۰

۷۱۰۰۰ ۴/۴۳۲ ۹۰۰۷ ۵۵۸۶ ۱۸۳۹ ۸۵۳ ۶۲۰۳ ۶۲۵

۷۱۰۰۰ ۴,۴۳۲ ۹۰۰۷ ۵۵۸۶ ۱۸۳۹ ۸۵۲ ۶۲۰۳ ۶۲۵

۱۰۰۰۰۰۰ ۱۲۷۹۴۳ ۷۵۸۱۹۶ ۴۶۰۰۵۲۰ ۲۲۸۰۰۰ ۷۱۳۵

۱۰۰۰۰۰۰ ۱۲۷۹۴۳ ۷۵۸۱۹۶ ۴۶۰۰۵۲۰ ۳۲۸۰۰۰ ۷۱۳۵

۸۱۲۷۶۳ ۳۶۵۴۹۷ ۶۵۰۹۷۱ ۴۹۳۷۹۴ ۰۶۸۹۴۳ ۴۹۴۷۹۴

۸۱۲۷۶۳ ۳۶۵۴۹۷ ۶۵۰۹۷۱ ۴۹۳۷۹۴ ۰۶۸۹۴۳ ۴۹۴۷۹۴

ا ب پ ت ث ج چ ح خ د ذ ر ز ژ س ش

آ ب پ ت ث ج چ ح خ د ذ ر ز ژ س ش

ص ض ط ظ ع غ ف ق ک گ ل م ن و ه ی

ص ض ط ظ ع غ ف ق ک گ ل م ن و ه ی

ا ص ب ک ل ع ط ق م ف ض ص و ن ه خ

ا ص ب ک ل غ ط ق م ف ض ص ز ن ه خ

ض گ ظ پ ض آ غ ث ج چ ذ ت ث ی ز ت

ض گ ظ پ ض آ غ ث ج چ ذ ت ث ی ز ت

یک دو سه چهار پنج

یک دو سه چهار پنج

شش هفت هشت نه ده

شش هفت هشت نه ده

Figure 2-8: Farsi collection form, page two, collected and labeled as Far 0371.

2.3.4 Scanning, Noise Removal and Preprocessing Before Data Extraction

All the forms were scanned and saved as true color (24 Bit), 300 DPI lossless TIFF images. An algorithm was developed to remove the red lines from the collection forms. We had the following challenges while removing the red lines:

1. As shown in Figure 2-9 (a-c), in some cases writers wrote outside the field boundaries and overlapped the red lines.
2. Forms were filled out with black and blue ink pens but if we zoom into some handwriting, as shown in Figure 2-9 (d), we can see some red traces in pen strokes. Therefore, the algorithm had to avoid removing the red pixels in the writing.
3. Some of the forms were delivered from Iran to Canada so they contained unwanted artifacts due to folding and crumbling.
4. Some of the forms had grayish and yellowish backgrounds.

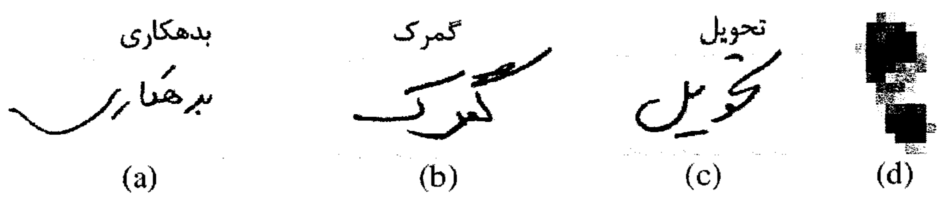


Figure 2-9: Examples of Farsi Handwritten words extending outside the red boundaries (a-c) and red traces in blue or black ink found in some handwritten words (d).

Finally, the forms were converted to grayscale before extracting the handwritten elements.

2.3.5 Data Extraction

When the forms were scanned and converted to the grayscale images, the boxes which included the handwritten samples were extracted. To do so, four small black squares were considered at the corners of each page. The Center of the top left square was then selected as the origin of the horizontal and vertical axis. These four black pixels were also used to do skew correction when it was required.

After this procedure, the correlation matching was applied between the selected black square of the template with each form, and the boxes that were extracted from all the forms, and each box was saved in its relevant class (folder).

2.3.6 Noise Removal and Preprocessing after Data Extraction

In this procedure, we removed the noise from the images. "Salt and Pepper" noise was found in our images. "Salt and Pepper" noise is represented as random black pixels on a white background image. A noisy pixel has a value that is not close to the surrounding pixels. To remove "Salt and Pepper" we used a non-linear filter, which is called a Median filter. A Median filter can preserve image details. It is called median filter because for each pixel in the image, it sorts the neighbouring pixels upon their intensities and replaces the original value of each pixel with the median value of the neighbouring pixels. [20]

The Matlab command to remove the Salt and Pepper noise is shown below. The Medfilt2 function performs median filtering of the matrix A by using the default 3-by-3 neighbourhood [13].

```
B = medfilt2(A)
```

Also, we noticed that mapping the intensity value of the grayscale image to new values could increase the contrast of the output image and consequently, could increase the quality of the image. Matlab has the following command to increase the contrast of the image:

```
B = imadjust(A)
```

"Imadjust" maps the intensity values in grayscale image I to new values in J such that 1% of the data is saturated at low and high intensities of I. This increases the contrast of the output image J [13].

Finally, we found the bounding box for each image and saved it with a 300*300 resolution in TIFF file format. All of the images were saved as binary as well as grayscale images. To find the bounding box of the word, Matlab applies the following function:

```
s = regionprops(bw , 'Image');
```

The Bounding box separated our area of interest from the rest of the image. Finding the bounding box could prevent unnecessary processes on the image. The area of interest in word recognition is the smallest rectangular area which includes the whole word image.

Figure 2-10 shows a word image and the red bounding box which surrounds it.

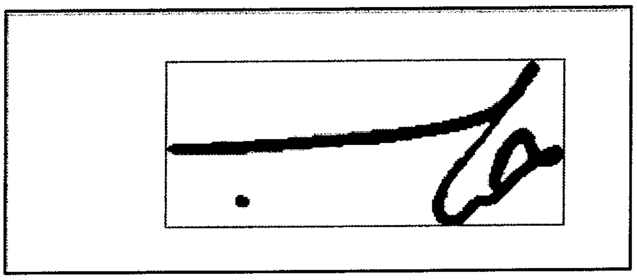


Figure 2-10: A word image and the red bounding box which surrounds it.

2.4 Dataset Structure Overview

The Farsi Dataset, as introduced in Section 2-3, follows the same structure of CENPARMI's other Indo-Iranian Datasets. The Data collected from 872 forms has been divided into four Series. These series give an advantage to the user to perform separate experiments on different parts of the Dataset. The first Series has been gathered from the form's first page and contains 63 different words and 5 special symbols. This data has been gathered mainly from Farsi native speakers living in Montreal.

The second Series of the Dataset has been collected from the two pages of Farsi forms and includes 73 Word , 5 Special Symbols, 20 Isolated Digits, 64 Isolated Characters, 2 Dates and 41 Numeral Strings with different lengths. This data has been gathered mainly from Farsi native speakers living in Canada (Montreal, Toronto, and Vancouver).

The third Series of the Dataset has the same format as Series 2. It includes the same type and number of script boxes. This data has been gathered mainly from Farsi native speakers living in Iran.

The Dataset is divided into Grouped and Ungrouped subsets, which will give the user the flexibility of whether or not to use CENPARMI's pre-divided (Grouped) Dataset (60% of the images are used as the Training set, 20% as the Validation set and the rest as the Testing set) .

All the images that can be found in Series 1, Series 2 and Series 3 have been gathered in Series 4. Figure 2-11 shows the main structure of the Dataset.
The Dataset also includes the collected forms and the Text (Document) Dataset.
In the next section, we will describe our Dataset in detail.

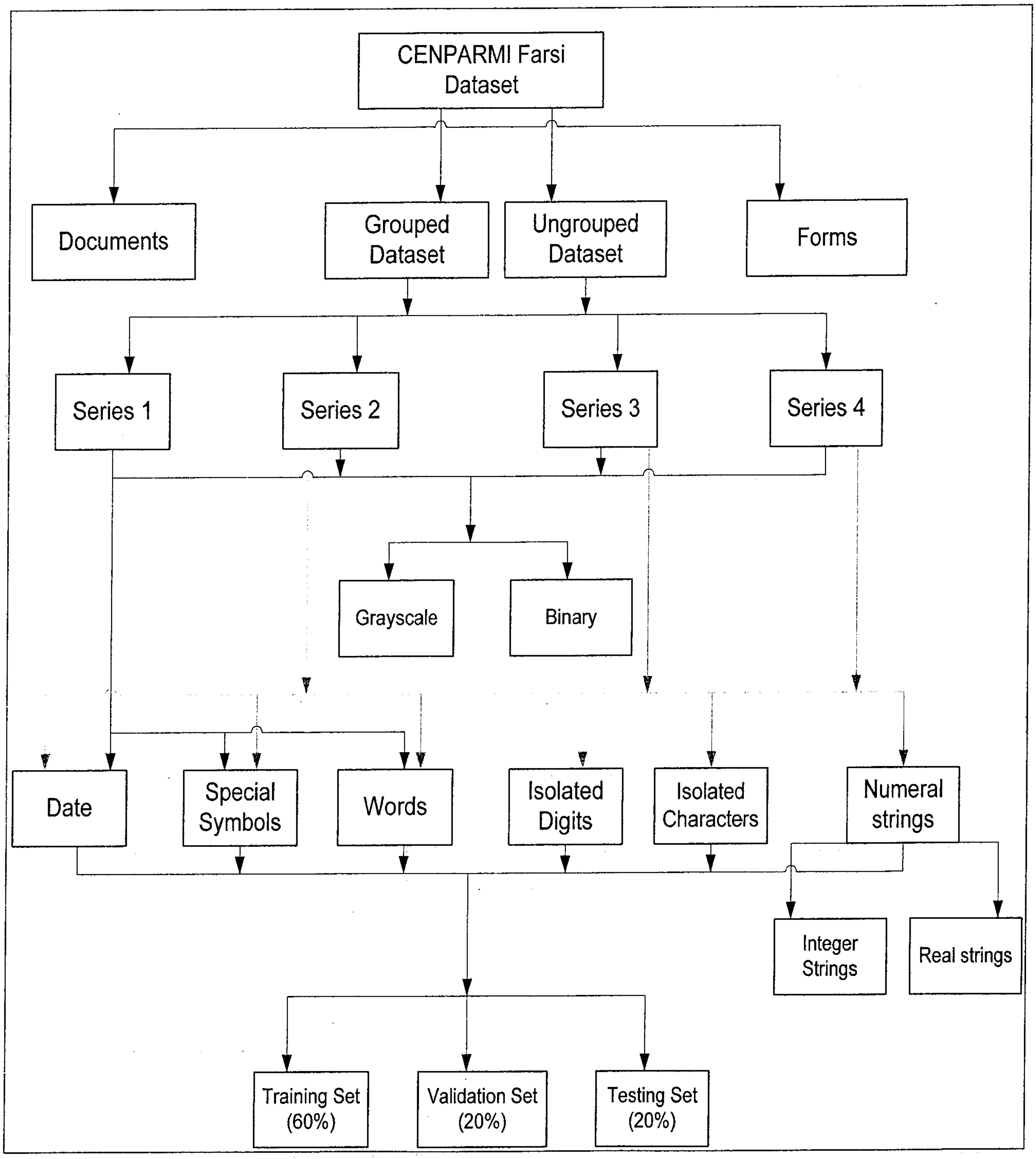


Figure 2-11 : CENPARMI's Farsi Data Set structure. Only the Grouped subset includes Training, Validation and Testing sets.

2.5 Dataset details and statistics

In this section, we will describe each subset of CENPARMI's Farsi Dataset and its statistics.

2.5.1 Farsi Words Dataset

The Farsi Words Dataset consists of 73 word classes which are officially used for measurement and counting purposes. These words include the measurement units of Distance, Volume, Weight, Currency and words which are usually used in financial documents and day-to-day business activities. The 73 categorized Farsi words are shown in Table 2-7.

Selected Words	**Farsi Equivalent**
Financial Words	
Balance	مانده
Amount	مبلغ
Cash	نقد
Expense	هزینه
Credit	اعتبار
Delivery	تحویل
Due	سررسید
Custom	گمرک
Expire	انقضا
Interest	سود
Inventory	انبار
Issue	مورد
Period	مدت

Table 2-7 : Categorized, selected Farsi words.(part a)

Selected Words	Farsi Equivalent
Financial Words (Cont'd)	
Plus	اضافه
Price	قیمت
Loan	وام
Rent	اجاره
Stock	سهام
Tax	مالیات
Total	مجموع
Money Order	حواله
Document	سند
Weight	وزن
Width	عرض
Duty	عوارض
Carton	کارتن

Table 2-7: Categorized, selected Farsi words. (part b)

Selected Words	**Farsi Equivalent**
Financial Words (Cont'd)	
Debit	بدهی
Number	شماره
Bill	صورتحساب
Account	حساب
Distance	
Centimeter	سانتی متر
Inches	اینچ
Meter	متر
Volume	
Milliliter	میلی لیتر
Liter	لیتر
Volume	حجم
Weight	
Milligram	میلی گرم
Gram	گرم
Kilogram	کیلو گرم

Table 2-7: Categorized, selected Farsi words. (part c)

Selected Words	**Farsi Equivalent**
Counting	
One	یک
Two	دو
Three	سه
Four	چهار
Five	پنج
Six	شش
Seven	هفت
Eight	هشت
Nine	نه
Ten	ده

Table 2-7: Categorized, selected Farsi words (part d).

Each word class consists of approximately 516 images. In verification and post-processing stages some images were eliminated because of the remaining noises in the images that could mislead the classifier. Some Farsi handwritten words are shown in Table 2-8.

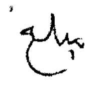
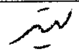
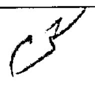
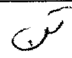
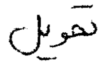
Equivalent Word in English	**Farsi Printed word**	**Equivalent Extracted Handwritten Image**
Amount	مبلغ	
Liter	لیتر	
Milligram	گرم	
Tons	تن	
Delivery	تحویل	

Table 2-8: Selected Farsi words, printed and handwritten, and their equivalents in English.

2.5.2 Farsi Numeral Strings Dataset

The collected Farsi Numerals can be divided into Integer numerals and Real numerals. Integer numerals can be found in lengths of 2, 3, 4, 5, 6 and 7. Real numerals which have floating points can be found in lengths of 3 and 4. Statistics for the numeral strings are shown in Table 2-9. Figure 2-12 and Figure 2-13 show some samples of Farsi Numeral strings.

Numeral Strings	**Number of Images**
Real Strings	**1064**
Length_3	709
Length_4	355
Integer Strings	**12270**
Length_2	4,504
Length_3	2,130
Length_4	2,486
Length_5	354
Length_6	2,085
Length_7	711

Table 2-9: Statistics for numeral strings in the Farsi Dataset.

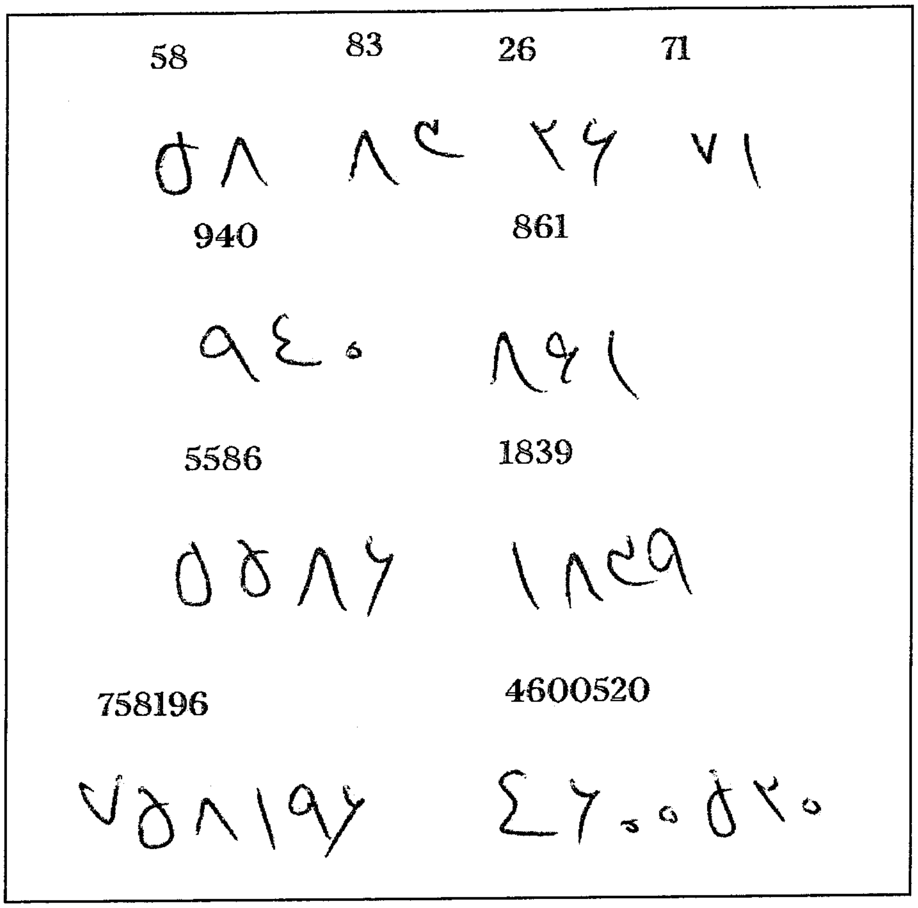


Figure 2-12: Some samples of gathered Integer numeral strings.

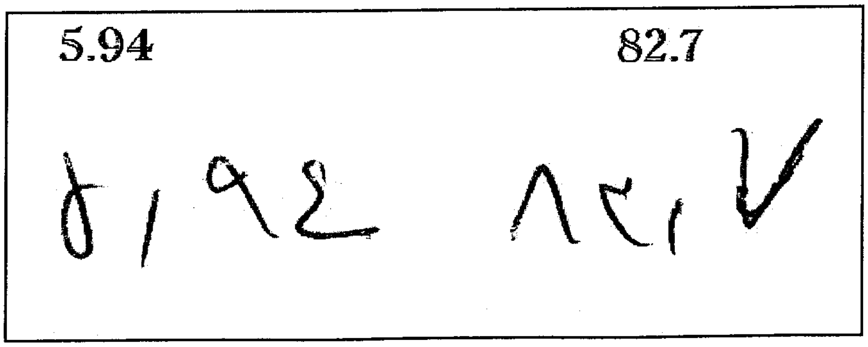


Figure 2-13: Some samples of gathered Real numeral strings.

2.5.3 Farsi Isolated Digits Dataset

The Farsi Isolated Digits Dataset includes digits which have either been written as isolated by the writers, or the digits which have been segmented from the gathered numeral strings. The two types of digits were labelled in different ways so they could be identified in the future.

Table 2-10 shows the number of occurrences of each isolated digit in each collected form. The total number of Isolated Digits gathered in all of the forms was 24,121. These digits were saved as Gray_all_0 to Gray_all_1. They were divided into 14,473 digits for the Training set, 4,824 digits for the Validation set and 4,824 digits for the Testing set. Figure 2-14 shows some Farsi isolated digits gathered from a form, and their equivalents in English.

Isolated Digit	Number of occurrences in each form
0	30
1	14
2	16
3	17
4	16
5	15
6	16
7	15
8	15
9	17

Table 2-10: The number of occurrences of isolated Farsi digits in each collected form.

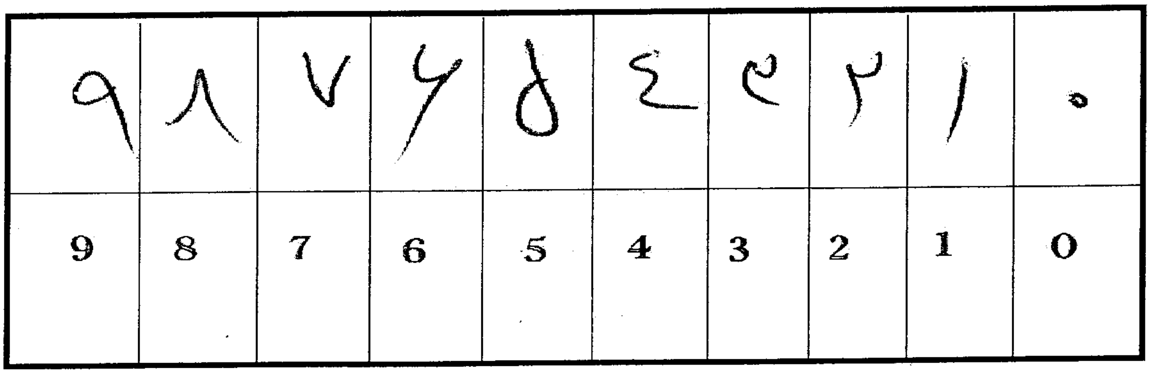


Figure 2-14: Farsi handwritten digit samples, and their equivalents in English.

2.5.4 Farsi Isolated Character Dataset

As described earlier, Farsi has 32 letters. Farsi has sets of these letters such that the letters in each set are similar and their only difference is in the number of dots they have.

These sets are shown in Figure 2-15.

{ب, پ, ت, ث, ن}, {ج, چ, ح, خ}, {د, ذ}, {ر, ز, ژ}

{ف, ق}, {س, ش}, {ص, ض}, {ط, ظ}, {ع, غ}

Figure 2-15: Similar sets of letters in the Farsi characters.They are differenciated by the number of dots they have.

In order to get more samples from the writers, we chose one letter from each set of the letters and asked each writer to rewrite it. Finally, we got two or three samples per set from each writer. If a letter was repeated in the form, it was labelled separately. For instance, the letter "Saad" was repeated twice so it was labelled as "Saad" and "Saad2". We did not combine the folders as having separate folders would make it easier to track the images in the future. In the statistics of Table 2-11, the total number of images for a letter which exists in both folders is mentioned. Figure 2-16 shows a sample of Farsi handwritten Isolated Characters.

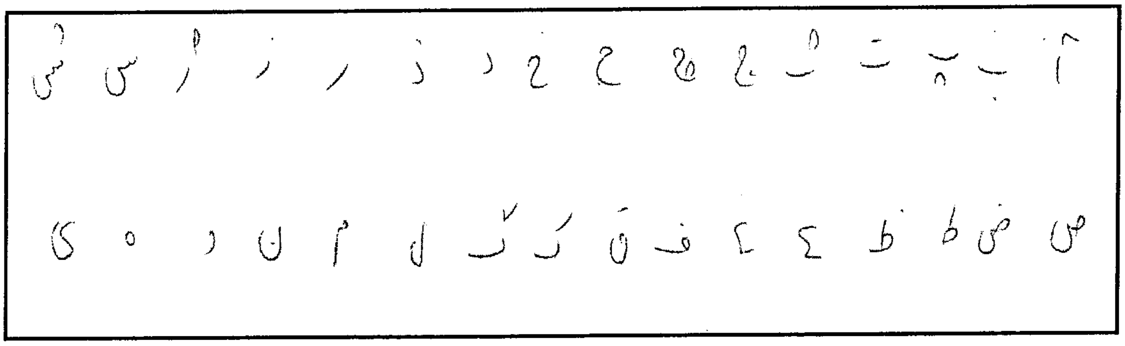


Figure 2-16: Farsi handwritten isolated characters

	Total	Training Set	Validation Set	Testing Set
Isolated Characters	21,336	12,802	4,267	4,267

Table 2-11: Statistics for Farsi handwritten Isolated Character subsets.

2.5.5 Farsi Text (Document) Dataset

The Farsi Text Dataset consists of three different documents. They have real financial and commercial contexts one of each page. Each text includes some words from the Farsi words Dataset so they can be used with the purpose of training the word

spotting systems. Eleven different writers have rewritten the documents. The number of data which was gathered in the Farsi Text subset was substantially less than the other Datasets' subsets and could be a field for further efforts in the future.
The idea behind having this subset was to test a system which could spot each of the 73 existing words in the Text and to send them to a Word Recognition System to examine the overall efficiency of the Dataset.

2.5.6 Farsi Special Symbols Dataset

The special symbols collected in the Dataset are illustrated in Table 2-12. These special symbols are the most commonly used symbols in Farsi financial documents. Statistics for the special symbols subset are shown in Table 2-13.

Selected Symbols	
Number sign	#
At sign	@
Colon sign	:
Slash sign	/
Floating Point	.

Table 2-12: Selected special symbols in the Dataset

	Total	Training Set	Validation Set	Testing Set
Special Symbols	2,738	1643	548	547

Table 2-13: Statistics for the Special Symbols subset.

2.5.7 Farsi Date Dataset

Iranians usually follow the YEAR/MONTH/DAY format in writing the dates. The CENPARMI Farsi Date Dataset includes 295 images. Table 2-14 shows some Dates collected from the forms and their equivalents in English.

Date	English Equivalent
۱۳۸۶/ ۹/ ۱۴	1386/9/14
۲۰۰۷/ ۱۱ /۳۰	2007/11/30
بیست‌وسوم نوامبر ۲۰۰۷	Twenty third of November 2007

Table 2-14: Three Different date styles collected in the forms and their equivalents in English.

2.6 Labelling

After the box extractions were done, each box was labelled in such a way that it could be identified in the future. For instance, we used “FAR0212_P03_058” to label the image in Figure 2-17. This label indicates that the image has been extracted from the form number 212 and the box number is 58.

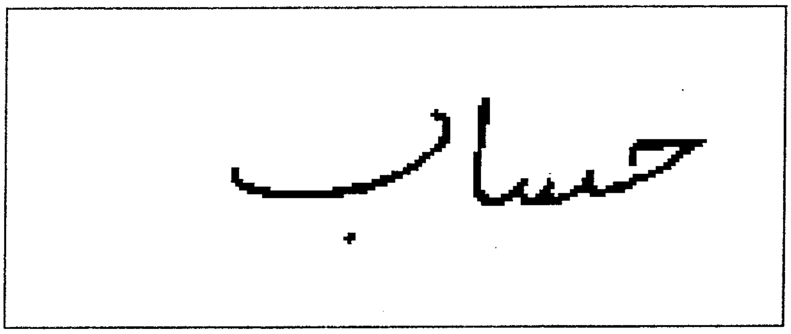


Figure 2-17: The word "account" extracted as box number 58

2.7 Verification and Post-processing

Verification and error detection by the human observer is inevitable in data gathering. Some handwriting errors may occur during the fillings of forms and some others can happen during the script box extraction or segmentation of the numeral strings. Therefore, at the end of each stage, a human observer has checked, cleaned and distributed the data with errors.

2.8 Conclusion

In this Chapter we described how we gathered the Farsi dataset to train and test our recognition system with various Farsi handwriting styles. The following chapters will describe the design and implementation of the handwriting recognition system.

Chapter 3: Word Preprocessing

After data collection, we had to process the images to make them ready for recognition. Therefore, this stage was called the Word Preprocessing stage. If the preprocessing stage was not done successfully, then the raw data would not have a good quality then it could simply mislead the classifiers, and finally the recognition system would fail.

The other reason to consider this stage as one of the most important steps in the Word Recognition Procedure is it forms the foundation in designing the real applications for the real world. To train our recognition system, we collected specific kinds of data to fulfill this purpose, in a controlled environment. In other words, the data was not gathered from the real world's texts such as Bank checks, envelopes, etc. The real world's texts are not as clean and neat as artificial ones. They may be written on folded and dirty pieces of paper. Scripts can be broken. People may be more careless in writing the scripts, etc. Therefore, a good recognition result comes from a good preprocessing which delivers clean data. This chapter will describe the preprocessing steps taken through our recognition path, such as Image Binarization, Skeletonization and Dilation.

3.1 Image Binarization

Image Binarization converts an image of 256 gray levels to a binary level, black and white image. The reasons we use a binary image instead of a gray image for the word recognition processes are as follows:

1. Image binarization segments an image into foreground and background pixels.
2. Each pixel value in a binary image is saved in a single bit instead of 8 bits for 256 gray levels, so the image will have a smaller size. The smaller size of the image will lead to less usage of the memory and processor.
3. To preprocess our image such as removing the noises, skeletonizing and dilating the image, it is much easier to deal with two values instead of 256.
4. The computer is a binary system; therefore, designing a system based on this fact makes the system more compatible with the computer.

The bit value of zero is interpreted as black while the bit value of one is interpreted as white.

To binarize an image, we should consider either a single parameter known as the intensity threshold or multiple thresholds known as a band of intensity values. Then, each pixel in the image is compared with the threshold. If the pixel's intensity is higher than the threshold, the pixel is set to one; otherwise it is set to zero.

To define the intensity threshold, Matlab uses Otsu's method that minimizes the intra-class variance [12]. The Otsu algorithm shows that minimizing the intra-class variance is the same as maximizing the inter-class variance. The threshold that minimizes the intra class variance can be calculated as follows:

$$\sigma_{\omega}^{2} = \omega_1(t) + \omega_2(t)\sigma_2^2(t).$$

where ω_i is the probability of the two classes which are separated by t as the threshold and σ_i^2 represents the variances of these classes [37].

To convert a gray-level image to a binary image, we use the following Matlab commands:

```
level = graythresh(Image);
BW = im2bw(Image,Level);
```

where the level = graythresh(Image) command computes a global threshold (level) that can be used to convert an intensity image to a binary image with im2bw. The level is a normalized intensity value that lies in the range of (0, 1) [13].

After getting the images binarized, we can start making them ready for the recognition process.

3.2 Skeletonization

The main purpose of skeletonization is to extract a region-based shape feature of the general form of an object without having to change the structure of the object. To process the word images we use the skeleton of the words to avoid dealing with the unequal stroke width.

Skeletonization in Matlab can be done by using the "bwmorph" function, which applies a specific morphological operation to the binary image. It is described as follows:

```
Image_Skeleton = bwmorph(Binary_Image,'skel',n);
```

Defining n = Infinity, pixels on the boundaries of objects are removed but it does not allow objects to break apart [13].

Figure 3-1 shows a word image and its skeleton. Matlab uses the same thinning algorithm that was used in [14], which is briefly described below:

1. Divide the image into two distinct subfields in a checkerboard pattern.
2. In the first subiteration, delete pixel p from the first subfield if and only if the conditions G1, G2, and G3 are all satisfied.
3. In the second subiteration, delete pixel p from the second subfield if and only if the conditions G1, G2, and G3 are all satisfied.

   Condition G1:

$$X_{H(p)=1}$$

where

$$X_H(p) = \sum_{i=1}^{4} b_i,$$

$$b_i = \begin{cases} 1 \; if \; x_{2i-1} = 0 \; and \; (x_{2i} = 1 \; or \; x_{2i+1} = 1) \\ 0 \qquad\qquad\qquad\qquad\qquad otherwise \end{cases}$$

$x_1, x_2, \ldots, x_8$ are the values of the eight neighbours of pixel p, starting with the most eastern neighbour and numbered in a counter-clockwise order.

Condition G2:

$$2 \le \min\{n_1(p), n_2(p)\},$$

where

$$n_1(p) = \sum_{k=1}^{4} x_{2k-1} \vee x_{2k},$$

$$n_2(p) = \sum_{k=1}^{4} x_{2k} \vee x_{2k+1}.$$

Condition G3:

$$(x_2 \vee x_3 \vee \bar{x}_8) \wedge x_1 = 0.$$

Condition G4:

$$(x_6 \vee x_7 \vee \bar{x}_4) \wedge x_5 = 0.$$

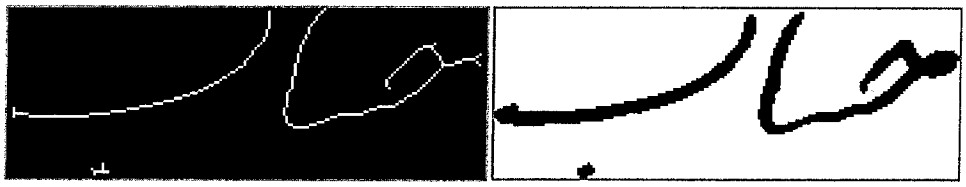


Figure 3-1: Black background image at the left shows the skeleton of the white background image at the right. They both show the word "Account" in Farsi.

3.3 Dilation

Dilation is one of the basic operations used in mathematical morphology. The dilation operation usually uses a structuring element for probing and expanding the boundary pixels of an image [16].

We dilate the skeletonized word image to make our system independent of stroke width. Matlab uses the imdilate function to perform the dilation:

```
SE = strel ('square',4)
IM2 = imdilate (IM, SE)
```

In this function call, the strel function creates a square structuring element whose width is 4 pixels.

The dilation operation takes two pieces of data as inputs. The first is the image which is to be dilated. The second is a set of coordinate points known as a structuring element (Kernel). This kernel determines the precise effect of the dilation of the input image.

In our recognition system, we have defined the strel function as follows:

SE = strel ('square', 4)

The following K matrix shows the structuring element, which is a 4 x 4 square:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

The mathematical definition of dilation is described as follows:

- Suppose that X is the set of Euclidian coordinates corresponding to the input binary image, and that K is the set of coordinates for the structuring elements.
- Let K_x denote the translation of K so that its origin is at x.
- Then, the dilation of X by K is simply the set of all points x such that the intersection of K_x with X is not empty [17].

In Matlab, the "structuring element decomposition" technique is used to improve the recognition performance. For instance, the dilation of the object by a large structuring element, with a size of 20 by 20 is computed faster by dilating first with a 1 by 20 structuring element, and then with a 20 by 1 structuring element. More information can be found in [13].

Figure 3-2 shows a written word image from the Farsi dataset and its dilated image.

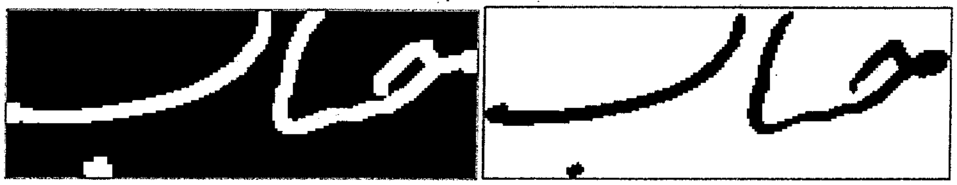


Figure 3-2: Black background image at left shows the dilated image of the white background image at right. They both show the word "Account" in Farsi.

After gathering the dataset and preprocessing the data to enhance their qualities and making them ready for the recognition processes, we can start the recognition procedure. In the next chapter, we will describe the feature extraction which is the first step in the recognition procedure.

Chapter 4: Feature Extraction

4.1 Definitions

For Feature Extraction, we first need to define the features. In the real world, we recognize everything by extracting its identifying features which can be different from one object to the other. In pattern recognition, choosing the types of features depends on the classifier which will be used to recognize them and the characteristics of the pattern. Selected features should be able to model the characteristics of the pattern. For instance, if we want to extract some features to recognize a human's face, we use the following identifying features: eyes, mouth, nose, etc.

As described earlier in Chapter 2, Section 2.2, Farsi has a cursive script which means that the letters in a word are connected, making the whole word a complex stroke [18].

Features can be extracted globally or locally from the image. In extracting the global features, we usually consider the word as a whole image, which is called the holistic approach which does not segment the word into smaller connected components. While in analytical approaches, the word is segmented into smaller units, therefore the features are extracted locally from each small unit. Examples of local features include: Percentage of foreground pixels within a window, foreground-background transition statistics, percentage of the foreground pixels in the core, and regions of ascenders and descenders. Global features can be structural or statistical features. It is better to use structural features than the statistical features in Farsi handwriting recognition because Farsi has a cursive script with lots of variability in handwriting, and statistical features (such as number of connected components, holes, ascenders and descenders) cannot tolerate a large degree of variability. Coefficients of the Fourier transform and invariant moments are considered as global statistical features [21].

4.2 Feature Selection

Classification accuracy strongly depends on the type of features we choose to recognize a word. To select the features, we had to consider the Hidden Markov Models' (HMM's) capabilities and the Farsi scripts' characteristics. Experiences have shown that HMM can make more accurate recognitions based on the structural features than the statistical features.

In our recognition system, we used Baseline Dependant Features to identify the words. Baseline dependant features emphasize the existence of descenders and ascenders. We designed a sliding window to go over the image and extract its features locally. The region which was restricted in the sliding window at the time was called a "Frame". The sliding window's height was the same height as the word image and the width was twice the stroke width of the word. Before feature extraction, one of the fundamental morphological operations, Dilation, was applied to the skeleton of the word image to make the image stroke widths at most four pixels wide to ensure proper contour generation. Therefore, the sliding window was 8 pixels wide and when we shifted the window to the right, we only shifted it by four pixels. In other words, the windows had 50% of overlap to each other. Figure 4-1 shows the overlap between the two consecutive sliding windows.

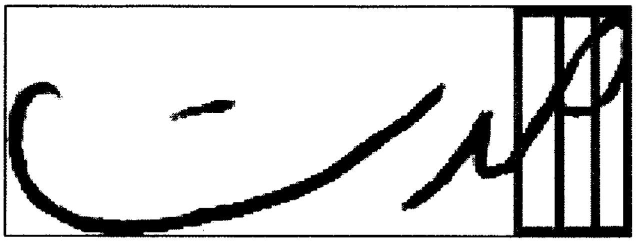


Figure 4-1: The word "Period" is shown in the figure. Red and green windows are consecutive and the 50% overlap is shown.

The more we give identifying features to the classifier, the better recognition result we can obtain, unless we mislead the classifier by overtraining it. Therefore, to extract more features, we divided each window horizontally into 15 equal blocks. Finally, the local features were extracted.

Our selected features were language-independent and some of them were calculated with reference to the baselines (Main Baseline, Upper and Lower Baseline) of the word. These baselines will be described in the next section.

4.3 Baseline Detection

Finding the baselines is a necessary step prior to doing some script preprocessing tasks, such as skew corrections, segmentation and feature extractions. We avoided skew correction in our preprocessing module because it caused distortion in our images. There were no or little skew detected in our word images so we considered them as the variations in handwritings.

For each word, the Baseline, Upper baseline and Lower baseline were detected. The Upper and Lower baselines divided the word image into three zones. The restricted zone between the Upper and Lower baselines was called the core zone. Baselines are shown in Figure 4-2 for the “Amount” word image.

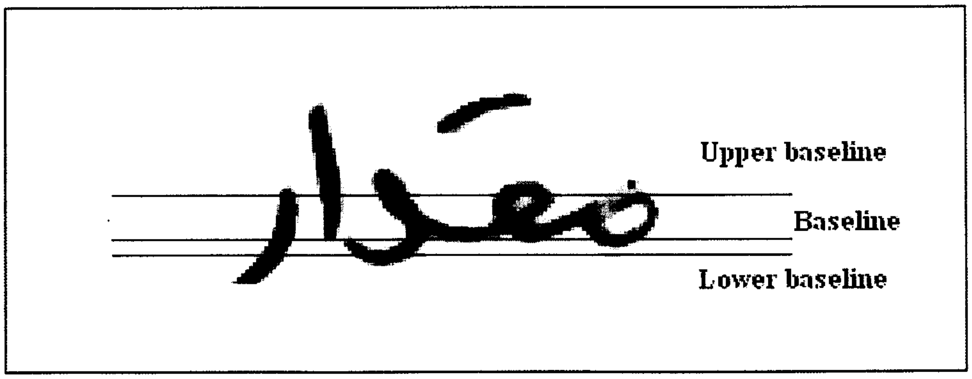


Figure 4-2: Baselines are shown for the "Amount" word image.

The techniques used for extracting the baselines in this research are as follows:

1. Scan the image from top to bottom.
2. Find the row with the most black pixels and call it the "Baseline".
3. Start scanning again from the Baseline of the image to the top of the image (exclude the Baseline).
4. Find the row with the most black pixels and call it the "Upper Baseline".
5. Start scanning from the Baseline of the image to the bottom of the image (exclude the Baseline).
6. Find the row with the most black pixels and call it the "Lower Baseline".

We have made two assumptions in this algorithm. Firstly, we processed the bounding box of the image, which was assumed to be the smallest rectangle containing the word and secondly, we kept the y coordinates of the rows assuming to be the baselines' locations.

The features we used to identify our words were baseline-dependant and they could be divided into Distribution Features and Concavity Features, which are described in the following sections. Twenty-five features were extracted for each frame.

4.4 Distribution Features

In our system, distribution features are calculated based on the foreground pixels' (white pixels') densities. Distribution features are easier to detect when compared to the topological features such as number of loops, but they are less robust to noise and local distortion. Since we removed noises in the preprocessing stage and we avoided some preprocessings which may have caused distortion such as skew correction, we mostly used the distribution features to recognize the word images. Our approach is based on the algorithm described in [20] with few alterations.

Our feature vector includes 17 distribution features per frame, which are described below:

F1: Density level of the block.

Let b the density level of the block. Then, $b = 0$ if the number of foreground pixels in the current cell is zero, else $b = 1$.

F2: Density of foreground pixels, In other words, the sum of foreground pixels for each row of the block.

F3: Number of transitions between two consecutive blocks of different density levels.

F4: Derivative feature between the current frame and the previous one which shows the difference between the y position of gravity centers of the current frame and the previous one.

F5-F12: Eight features that represent the number of white pixels in each column of the current frame.

F13: Normalized position of the centre of gravity of the foreground pixels in the current frame with respect to the lower baseline. F13 is calculated as follows:

$$F13 = \frac{g - l}{h}$$

where g is the center of gravity, l is the position of lower baseline and h is the fixed height of the block.

F14: Density of foreground pixels over the baseline in the current frame.

F15: Density of foreground pixels under the baseline in the current frame.

F16: Number of transitions between two consecutive blocks of different density levels above the lower baseline.

F17: This feature represents the zone that includes the centre of gravity. If the center of gravity is above the upper baseline F17=1, If it is between the Upper and Lower baseline F17=2 and if it is under the Lower baseline F17=3.

4.5 Concavity Features

To gather some information about the local concavity and to identify the stroke direction in each frame, we calculated the local concavity features. They were extracted by using a 3x3 window. Our approach is based on the algorithm described in [20]. We considered each 3x3 subset of pixels by centering each subset with a background pixel (which is a black pixel in our images), and tried to match the subset with one of the four templates shown in Figure 4-3. These four templates are the four possible types of concavity configurations for a background pixel.

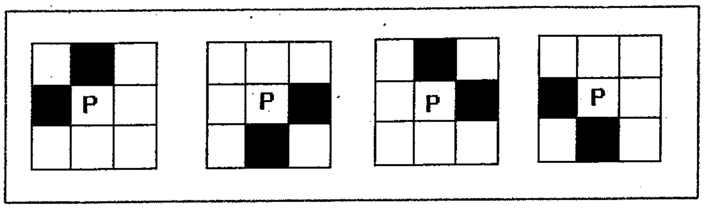


Figure 4-3: Four templates to show the concavity of a background pixel.

Then, we kept track of the number of background pixels which matched one of the above templates. Since we had different heights for each image, we normalized the heights by dividing them by the height of the image.

Therefore, F18-F21 were calculated as follows: N_{lu} stands for the number of background pixels, which are surrounded by a white pixel at the **L**eft and **U**p positions. N_{ur} stands for the number of background pixels, which are surrounded by a white pixel at the **U**p and **R**ight, N_{rd} stands for the number of background pixels, which are surrounded by a white pixel at the **R**ight and **D**own positions and N_{dl} stands for the number of background pixels, which are surrounded by a white pixel at the **D**own and **L**eft positions.

$$F18 = \frac{N_{lu}}{H} \quad F19 = \frac{N_{ur}}{H} \quad F20 = \frac{N_{rd}}{H} \quad F21 = \frac{N_{dl}}{H}$$

We also calculated these features for the core zone of the image. To normalize these values, we divided them by the height d of the core zone as follows:

d = UpperBaseline_YCoordinate - LowerBaseline_YCoordinate

$$F22 = \frac{CN_{dl}}{d} \quad F23 = \frac{CN_{rd}}{d} \quad F24 = \frac{CN_{lu}}{d} \quad F25 = \frac{CN_{ur}}{d}$$

From the beginning, we chose Hidden Markov Models (HMM) as the classifier to perform the recognition job in our system. After having studied many papers in the handwriting recognition domain, we got the idea that because of the connected nature of the Farsi script, HMM is an acceptable classifier to recognize the Farsi words. HMM systems can stochastically model sequences of variable lengths which occur very frequently in Farsi handwritten words. HMM can also cope with nonlinear distortions along one direction [21].

We chose a discrete HMM to limit the number of observation symbols. Therefore, the features should be quantized to a codebook vector. To quantize the feature vector to a codebook vector, we chose the K-means clustering algorithm, which is described in the following chapter.

Chapter 5: Clustering

Data Clustering is defined as follows: to cluster the data which have similar characteristics. The purpose is to retrieve the relevant information more quickly. The difference between clustering and classification is that in classification, we assign data to the predefined classes, while in clustering, clusters are created during the assignment [24].

5.1 Types of Clustering Methods

Clustering methods can be divided into two basic types, Hierarchical methods and Partitioning methods.

5.1.1 Hierarchical Methods

In the Hierarchical method of clustering, a tree structure is created to describe the grouping of data. There are two main strategies for Hierarchical clustering: Agglomerative and Divisive. In the Agglomerative method, each data is considered as a cluster and as the hierarchy moves up, two pairs of clusters are merged. In the Divisive method, all data are grouped into one cluster and as the hierarchy moves down, the splits are done recursively [22].

5.1.2 Partitioning Methods

In the Partitioning method, there is no tree structure to describe the data clustering and clusters are described in a single level. Each data will finally belong to a cluster and will be presented by a centroid. The Partitioning method uses the actual observation of data and not just their proximities. Therefore, it is a more suitable method for clustering large amounts of data. The K-means uses the partitioning method to cluster the data.

5.2 K-means Clustering Algorithm

The K-means clustering algorithm can cluster n observations into k mutually exclusive clusters. Each cluster is represented by a vector, which is the mean of the existing data in the cluster. Therefore, the data will finally be assigned to the cluster with the nearest mean. The purpose of the K-means algorithm is to minimize the squared Euclidean distances between the data in each cluster [23]. K-means is calculated as follows:

$$\sum_{i=1}^{k} \sum_{d_j \epsilon S_i} \|d_j - \mu_i\|^2.$$

where $\{d_1, d_2, \dots , d_n\}$ is the set of data to be clustered, k is the number of clusters, and μ_i is the mean of data in each set S_i.

The K-means algorithm is a heuristic algorithm. Therefore, the result depends strongly on the initial clusters and there is no guarantee of achieving the global optimum. The algorithm can be simply described as follows:

1. k initial means are randomly selected from the data.
2. The data with the nearest mean to the initial k randomly selected data will be assigned to the k th cluster.
3. The centroid of each cluster becomes the new mean.
4. Steps 2 and 3 are repeated until the sum of distances from each object to its cluster centroid cannot be decreased further [13].

5.3 Optimum Number of Clusters (*k*)

In the K-means algorithm, k is the number of clusters that is predefined to the algorithm and is considered as an input argument. Choosing an inappropriate value for k may lead to a bad recognition result. The proper choice of k is difficult and depends on the shape and scale of the distribution of points in a dataset and the desired clustering resolution [25].

There are different ways of calculating the proper value of *k*, such as Rule of thumb [25], The Elbow Method [27], Information Criterion Approach [28], Information Theoretic Approach [29], and Choosing *k* using the Silhouette Plot [24].
We will describe the Silhouette Plot and Rule of Thumb in this section, because the former is the way that Matlab software uses to find the proper value of *k* and the latter is a practical way of finding *k* in our research.

5.3.1 Choosing *k* using Silhouette Plot

As described, a way to find the proper value of *k* is to analyze the existing clusters. In Matlab, you can use a silhouette plot to see how well the data is separated into different clusters. There is a measure which ranges from -1 to +1 and indicates if a data point is very distant from the cluster that it does not belong to or is very close to it. Positive 1 shows a data point that is very close to another cluster and is probably wrongly clustered. The larger the quantity of data, the more time consuming and complex the calculation will be when plotting the K-means silhouette. Therefore, the silhouette plot is not a practical way of optimizing the number of clusters.

5.3.2 Rule of Thumb to find the *k*

Another way to calculate the optimum number of clusters is through the following formula: [27]

$$k \approx (\frac{n}{2})^{1/2}$$

Where n is the number of data points. Our feature vector's size is 716,596 x 25 (716,596 is the total number of blocks and 25 is the total number of features). Therefore, the number of data points is 1,791,490,012 and *k* is approximately 2993.

5.3.3 Finding the Optimum Value by Validating Data

Validating the trained HMM models gives us the advantage to find the optimum value for K-means. The process is to train the HMM models and then to try to validate the models with different values for K-means such as K=256, K=512, etc. The K value which gives us the best result will then be chosen for clustering. The validation process is described in Chapter 6.

5.4 K-means in Matlab: (Implementation Issues and Challenges)

Matlab has a built-in function to run the k-means. It is used as follows:

```
[IDX,C] = kmeans(X,k)
```

Or

```
[...] = kmeans(...,param1,val1,param2,val2,...)
```

Where IDX is the clustered data, C is the centroid of each cluster, X is the feature vector and k is the predefined number of clusters. In Matlab, some parameters and values could also be used to give the user the ability to control the iterative algorithm used by k-means. These parameters and values are described in detail in Matlab's Image Processing Toolbox Documentation [13]. We used the following command to cluster the data in our case:

```
[IDX,C] = kmeans(X, 2993,'distance','sqEuclidean','emptyaction','singleton');
```

The 'distance' parameter shows the algorithm used to calculate the Squared Euclidean distance of each point to the centroids, to cluster the data. The 'emptyaction' parameter is a very important parameter, which shows the algorithm regarding which action to take if a cluster loses all of its member data. Choosing the 'singleton' value for this parameter shows the algorithm to create a new cluster consisting of the one point furthest from its centroid [13].

Unfortunately, this built-in function only works for a small number of data with a few number of clusters. We ran it for our feature vector with the size of 716,596 x 25 (716,596 is the total number of blocks and 25 is the total number of features) and we got an "Out of Memory" error message. All of the memory management solutions suggested by Matlab documents to solve this problem did not work. Matlab explains the reasons to get the "Out of Memory" error message, as the system has run out of heap space to hold all of the variables or the problem is with the memory fragmentation. This topic is discussed more in [31].

So one way to solve the problem was to write a k-means program which could optimally use the memory and load our feature vector. We ran the program on the High Performance Computing (HPC) cluster environment (Super-Computers). Concordia's ENCS HPC Cluster Environment provides 608 2.2 GHz AMD Opteron
64-bit processor cores for job submission, utilizing the Infiniband network as the means for node-to-node communication and for I/O to the cluster filesystem. The theoretical peak performance is 4 GFLOP per second per core. For more information, refer to [30].
The other way to solve the "Out of Memory" problem was to implement the K-means algorithm in C++ language.
The selection of the Model and HMM will be described in the next chapter.

Chapter 6: Model Selection and Hidden Markov Models

After preparing the data, identifying their features and reducing their size by quantizing them, it was time to select the recognition system's model as this step was shown in Chapter 1, Figure 1-2. Model selection depends on the characteristics of the problem. A model is selected to predict output from input, which can be parametric or non-parametric. There are different types of models which can perform the mentioned task, such as linear models, classification and regression trees, neural networks, kernel and hybrid methods. The selection of the optimal model is difficult. The selected method should perform best on unseen (test) data.

Among all of the classification models, Hidden Markov Models (HMMs) were chosen because of the connected nature of the Farsi script. HMM systems stochastically model sequences of variable lengths and cope with nonlinear distortions along one direction. As described previously, the discrete HMM was selected to limit the number of observation symbols [21]. HMM has particular advantages when compared to the other models, such as embedded training, in other words, automatic training of character models on non-segmented words [41].

In this chapter, we will describe the Hidden Markov Model concepts, our HMM and its initial estimation. We will also discuss the practical issues related to the implementation and optimization of HMM.

6.1 Markov Systems

A Markov system is a chain that has different states with stochastic identities. In this chain, we have states at time *t* that are influenced by the states at time *t-1*. Hidden Markov Models are the best solution to these kinds of problems in Speech Recognition, Handwriting Recognition, Gesture Recognition, Bioinformatics, Financial Analysis, etc. In a Markov model, all the states are visible, therefore the state transition probabilities are the only parameters [33].

6.2 Hidden States

In Hidden Markov Models, the states are hidden to the observers but the outputs are clear. We call this kind of Markov Model, a "hidden" one because the sequence which leads to a specific output is hidden, even if the parameters are all precisely identified.

6.3 HMM Notations

A Hidden Markov Model has three parameters which are shown as follows:

$$\lambda = (\pi, A, B)$$

where λ is a Hidden Markov Model which is defined by π, A and B. π is the initial distribution of the states. A is the state transition matrix and B is the confusion matrix.

We consider N as the number of states in a model and M as the number of distinct observation symbols per state, which in a discrete HMM is the number of clusters, in other words, M is the number of alphabets. We also have to determine the topology of our Model.

We calculate the state transition matrix as follows:

$$A = \{a_{ii}\},$$

$$a_{ij} = P\left[q_{t+1} = S_j | q_t = S_i\right], \qquad 1 \leq i,j \leq N.$$

We denote $S = \{S_1, S_2, S_3, \dots, S_n\}$ which represents the states and the state at time t isq_t.

The topology of the HMM shows the states which can be reached through each state. For instance, in our model we considered that each state has a self-transition or a transition to the next state or the next two states, which is shown in Figure 6-1 . If any state can reach all the other states, then we have $a_{ij} > 0$ for all states [33].

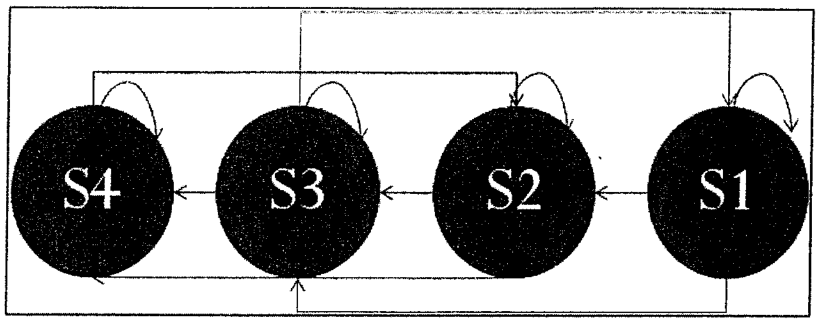


Figure 6-1 : A Hidden Markov Model with four states. Each state has a self-transition, a transition to the next state and another transition to the next two states.

The observation for symbol probability distribution is called the Confusion Matrix, which is shown by B. It is calculated as follows:

$$B = \{b_j(k)\},$$

$$b_j(k) = P[V_k \; at \;\; t | q_t = S_j], \quad 1 \le j \le N, \;\; 1 \le k \le M.$$

The Confusion matrix shows the probability of emission for symbol k at state j. Another parameter which should be defined to complete the model is π , as the vector of the initial state probabilities. It is calculated as follows:

$$\pi = \{\pi_i\},$$

$$\pi_i = P[q_1 = S_i], \quad 1 \le i \le N.$$

where π is a vector which shows the probability of being in each state at time t = 1.

6.4 Discrete or Continuous HMM

In speech recognition, Continuous HMM is more acceptable while in handwriting recognition, there has always been a challenge to select between the Continuous and Discrete HMM. In 1996, a research was conducted to show a comparison between the Continuous and Discrete HMM for cursive handwriting recognition [38]. The research showed that Discrete HMM leads to a better result in handwriting recognition.

In the Discrete Hidden Markov Model (DHMM), the output of the process is observed as a sequence of observations which belong to a finite alphabet. These observations represent the indices of a codebook. The codebook is calculated by a vector quantization method as per our description in Chapter 5. While calculating the quantized vector, some data will be lost due to the quantization error, which is called distortion. In other words, DHMM quantizes the data to a limited alphabet, which causes a loss of information.

By using the Continuous HMM (CHMM) we can eliminate the distortion problem, but CHMM has more parameters and requires more memory [21].

Therefore, we chose DHMM to design and implement our handwriting word recognition system.

6.5 Three Main Problems in HMM: Evaluation, Decoding and Learning

For the Hidden Markov Model to be useful in the real applications, three main problems should be solved:

1. **Evaluation:** Suppose we have different HMMs, each with a set of triple $\lambda = (\pi, A, B)$ and data as a sequence of observations. The problem is to find the best model which can generate the data. The Forward algorithm can solve this problem by calculating the probability of an observation sequence given an HMM model. This problem usually needs to be solved in script recognition or speech recognition processes, when we have different models and we want to match a testing script or spoken word with the existing models.
2. **Decoding:** The problem is to find the hidden states that lead to the sequence of observations. The Viterbi's algorithm is usually used to solve this problem. There is no "correct" sequence to be decoded. Therefore, we use the optimal criteria to solve the problem [37]. This problem needs to be solved widely in Natural Language Processing (NLP), where we need to tag the words with their syntactic classes as nouns, verbs, etc. So we consider the words in a sentence as observations and the syntactic classes as hidden states. The purpose is to find the best syntactic class for a word, given the context [36].

3. **Learning:** In this problem, we have the observation sequence, we know the hidden states that have led to the observations and we try to find the best (most probable) Hidden Markov Model (π, A, B) that describes the observed sequence. The Forward-Backward algorithm or Baum-Welch's algorithm is usually used to solve this problem.

In word recognition, we use the solution to problem 3 (Learning) to model the script, the solution to problem 2 (Decoding) to improve the model and the solution to problem 1 (Evaluation) to find the best matched script for a given test data.

6.6 Our Hidden Markov Model and Initial Estimation

It is very important to keep the original size of the image when we model and design our recognition system with a Hidden Markov Chain, because the number of sliding windows which should cover the whole image is an identifying feature which can be used to differentiate some of the words.

One of the most important parameters which should be defined for a Hidden Markov Model is the number of states. As described in [34], the numbers of states were chosen based on the average number of frames per class. The chosen number of states for each class is shown in Table 7-1.

For the topology of our model, we considered a right to left HMM (RTL HMM). Each state could have one self-transition or a transition to the next state or the next two states, as it was shown in Figure 6-1, for a four-state model.

To start working with the Hidden Markov Models, we also required some initial values. In other words, we had to define some initial values for A, B and π.

We considered the equal probability of staying in the current state or going to the next state or going to the next two states. The probability of going from the current state to states other than what was mentioned in the topology was zero. The following image shows a Transition matrix for the model that was illustrated in Figure 6-1:

	S_1	S_2	S_3	S_4
S_1	$1/3$	$1/3$	$1/3$	0
S_2	0	$1/3$	$1/3$	$1/3$
S_3	$1/3$	0	$1/3$	$1/3$
S_4	$1/3$	$1/3$	0	$1/3$

This Transition matrix is an (N x N) size matrix, where N is the number of states. To calculate the initial values for the Confusion matrix, we considered the equal probability of emission for each symbol in each state. Therefore, if M shows the number of symbols, the probability to emit each symbol at each state is $1/_M$. For instance, if we have 4 states in our model and 3 symbols to emit, then we will have a confusion matrix as follows:

	O_1	O_2	O_3
S_1	$^1/_3$	$^1/_3$	$^1/_3$
S_2	$^1/_3$	$^1/_3$	$^1/_3$
S_3	$^1/_3$	$^1/_3$	$^1/_3$
S_4	$^1/_3$	$^1/_3$	$^1/_3$

where O shows the observation or symbol.

Finally, to calculate π, the random values are chosen for the probability of the initial states. Then, we normalized the values to make the entries of the array add up to 1.

6.7 Train HMM Word Recognition System

There are different ways to train an HMM. A detailed description can be found in [33]. The most common criterion is called Maximum Likelihood (ML). In this criterion, during the Training stage the HMM parameters are first initialized and then iteratively re-estimated such that the likelihood of the model produced by the training sequences increases. In our recognition system, we first defined the initial estimation as per our description in section 6.6. The training process stops when the likelihood reaches a maximum value. Maximum Mutual Information (MMI) and Minimum Discrimination Information (MDI) are the alternative Training criteria [34].

In this research, we used the Baum-Welch's (BW) algorithm to train word class models. This algorithm is based on ML criterion. The first step consists of calculating $P(O|\lambda)$, which is the probability of the observation sequence O, given the model λ.

We used Forward algorithm to calculate $P(O|\lambda)$. The Forward variable, $\alpha_t(i)$, is defined as follows:

$$\alpha_t(i) = P\ (O_1, O_2, O_3, \dots, O_t, q_t = S_i|\lambda),$$

which is the probability of the partial observation sequence $O_1, O_2, O_3, \dots, O_t$, and state S_i at time *t*, given the model λ. The algorithm for inducting $\alpha_t(i)$ is described below:

Initialization:

$$\alpha_t(i) = \pi_i b_i\ (O_i), \qquad 1 \le i \le N,$$

Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i)\, a_{ij}\right] b_i\ (O_{t+1}), \qquad 1 \le t \le T-1,$$

$$1 \le j \le N,$$

Termination:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

The purpose of the BW algorithm is to adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$,. This is the most difficult problem in the HMM domain. The BW is an iterative algorithm based on the forward and backward algorithms. The backward variable $\beta_t(i)$ is defined as:

$$\beta_t(i) = P\ (O_{t+1}, O_{t+2}, O_{t+3}, \ldots, O_T, q_t = S_i, \lambda),$$

which is the probability of the observation sequence from *t+1* to the end, at state S_i at time t given the model λ. The algorithm for inducting $\alpha_t(i)$ is described below:

Initialization:

$$\beta_T(i) = 1, \qquad 1 \le i \le N,$$

Induction:

$$\beta_t(i) = \textstyle\sum_{j=1}^{N} a_{ij}\ b_j\ (O_{t+1})\ \beta_{t+1}(j), \qquad t = T-1, T-2, \ldots, 1$$

$$1 \le i \le N.$$

Now we can define the probability of being in state S_i at time *t*, and state S_j at time *t+1*, given the model and the observation sequence as follows:

$$\xi(i,j) = P\big(q_t = S_i\,, q_{t+1} = S_j\,|O, \lambda\big),$$

$$\xi(i,j) = \frac{\alpha_t(i)\ a_{ij} b_j\ (O_{t+1})\ \beta_{t+1}(j)}{P(O|\lambda)}.$$

$\gamma_t(i)$, the probability of being in state S_i at time *t* given the observation sequence *O* and model λ, is calculated as follows.

$$\gamma_t(i) = P(q_t = S_i \mid O, \lambda),$$

Or can be calculated using the Forward and Backward variables, as:

$$\gamma_t(i) = \frac{\alpha_t(i)\, \beta_t(i)}{P(O|\lambda)}.$$

The expected number of transitions from S_i is denoted by

$$\sum_{t=1}^{T-1} \gamma_t\,(i),$$

and the expected number of transitions from S_i to S_j is denoted by

$$\sum_{t=1}^{T-1} \xi_t\,(i,j).$$

To re-estimate the parameters π, *A* and *B* of an HMM, We can use the following formulas:

1. Expected frequency in state S_i at time t=1.

$$\bar{\pi}_i = \gamma_1(i).$$

2. Transition coefficient = expected number of transitions from state S_i to S_j, divided by the expected number of transitions from state S_i.

$$\overline{a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}.$$

3. observation symbol probability = expected number of times in state j, while observing symbol v_k, divided by the expected number of times in state j.

$$\overline{b_j(k)} = \frac{\sum_{t=1, s.t. O_t = v_k}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(j)}$$

The algorithm to Train the HMM Recognition System is described as follows:

1. For each folder :
    a. For each image in the Training set:
        i. Preprocess the image
        ii. Find the Feature Vector (FV) for the image [The FV matrix size is: The Number Of Sliding Windows multiplied by the Number of Features (25 in our case)].

iii. Save all FVs in a Matrix while keeping track of the number of FVs for each image. Call the Matrix as AMFV (All Main Feature Vectors).

b. Continue until all of the images in the current folder are processed.

2. Send AMFV and the number of preferred clusters to the K-means. The output matrix has the same row number as AMFV and one column because each row is now quantized to a single value and is called QFV (Quantized Feature Vector). Therefore, QFV is a one column matrix. Another output of the K-means will be the values for the centroids of the clusters.
3. Re-arrange the QFV as each row shows all the values for an image.
4. Save the values for all the images in each word class in a variable called "Data".
5. Call the "HMMTraining" function using Initial estimation and" Data" parameters. The Output will be three matrices: Transition matrix, Confusion Matrix and Prior matrix (Matlab's "hmmtrain" function can also be used to train the HMM).
6. Save these three matrices for each word class in a matrix called MainTrainedMatrices (To improve the model, keep all matrices being calculated in each of the ten iterations for each word class).
7. Continue until the folders are all processed.

The above process is shown graphically in Figure 6-2.

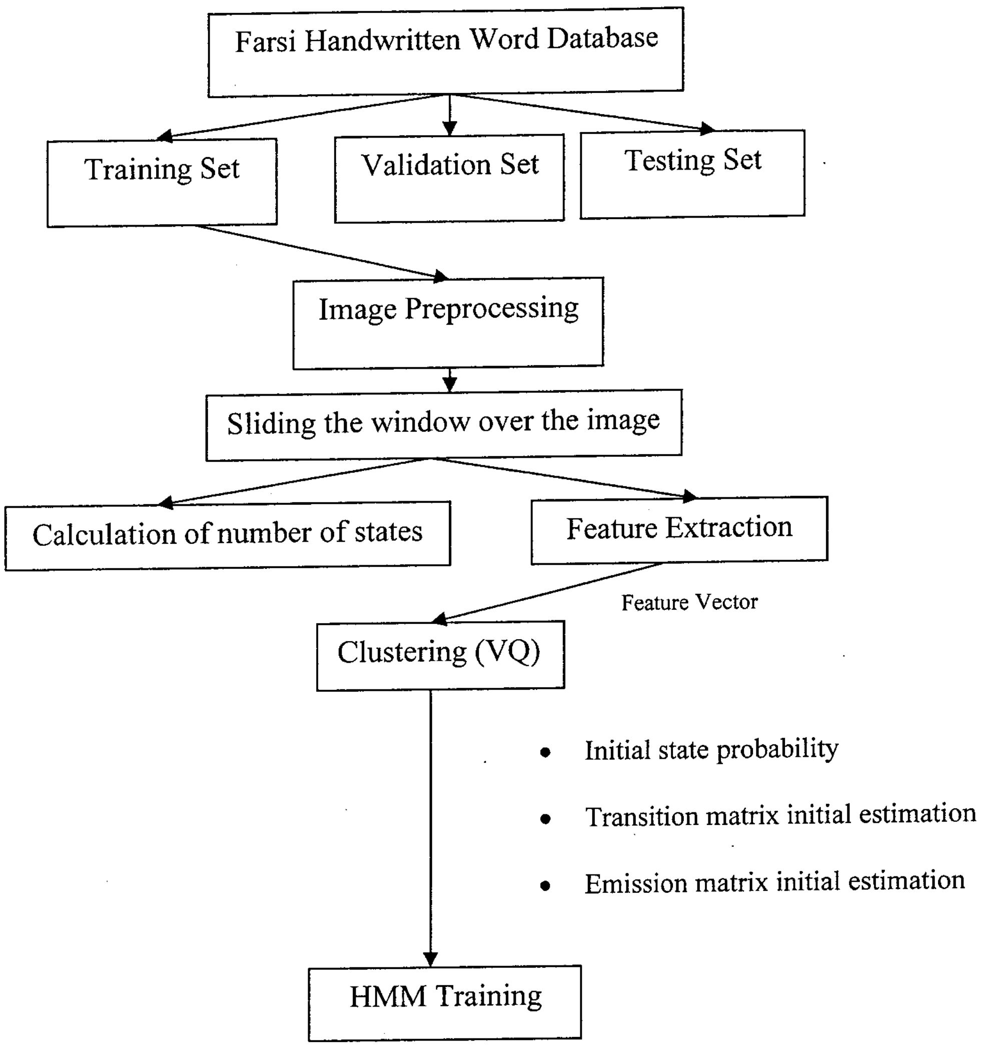

Farsi Handwritten Word Database
Training Set
Validation Set
Testing Set
Image Preprocessing
Sliding the window over the image
Calculation of number of states
Feature Extraction
Feature Vector
Clustering (VQ)
Initial state probability
Transition matrix initial estimation
Emission matrix initial estimation
HMM Training

Figure 6-2: Hidden Markov Model Training stage

6.8 Validation of HMM Word Recognition System

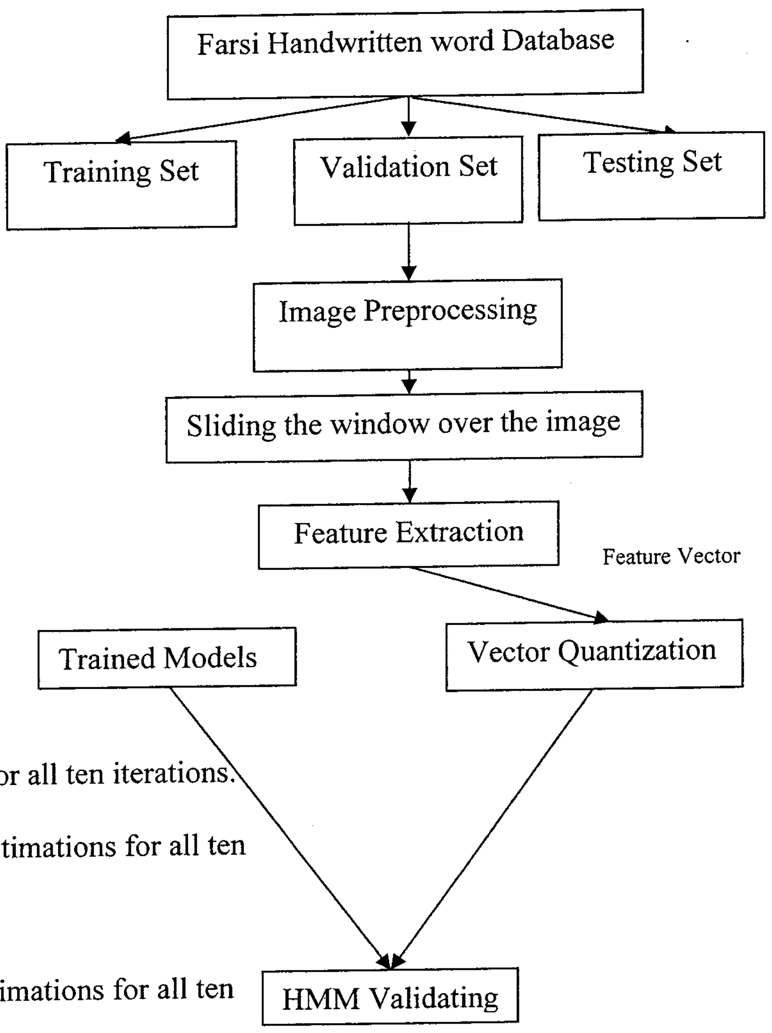
The Validation set is used to tune the HMM Word Recognition System. In validation stage, we tested samples in the Validation set with ten different models which were saved during the training of each word class. Finally we saved the model which could generate the most samples of the class. The Forward algorithm can solve this problem by calculating the probability of an observation sequence given an HMM model. We can also use the Viterbi's algorithm to solve this problem. The calculation of Forward algorithm was described in Section 6.7 and the calculation of Viterbi's algorithm will be described in Section 6.9.

The algorithm to Train the HMM Recognition System is described as follows:

1. For each folder:
    a. For each image in the Validation set:
        i. Preprocess the image
        ii. Find the Feature Vector (FV) for the image [The FV matrix size is: the Number of Sliding Windows multiplied by the Number of Features (25 in our case)].
        iii. Send the FV to the "VectorQuantization" program to find the most suitable cluster for the FV and save it as QFV (Quantized FV).

        iv. Save QFV in a variable called "Data".
        v. For the current folder's iteration from one to ten:
            1. Find the likelihood by using the Forward or Viterbi's algorithm (or Matlab's hmmdecode function).
            2. Keep the most suitable iteration number.
    b. When the images in a folder are all processed, choose the iteration number which was mostly selected as the best iteration number by the current folder's images.
    c. Save the three matrices of the best selected iteration as each word class model.
2. Continue until all the folders are processed.

The above process is shown graphically in Figure 6-3.



- State probabilities for all ten iterations.
- Transition matrix estimations for all ten iterations.
- Emission matrix estimations for all ten iterations.

Figure 6-3: Hidden Markov Model Validation stage

6.9 Testing of HMM Word Recognition System

In the Testing stage our problem is to find the best model which can generate the data. Viterbi's algorithm is able to match a single model to an observed sequence of symbols.

Consider the following variables:

- $\delta_t(i)$: scores the likelihood of the observation sequence $O_1, O_2, O_3, \dots, O_t$, having been produced by the most likely sequence of model states, which ends at state *i* at time *t*;
- $\psi_t(i)$: The array used to trace the maximum likelihood path which keeps a record of the states which maximized the likelihood from time *1* to *t*.

Viterbi's algorithm is described as follows [34]:

Initialization:

$$\delta_1(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N,$$

$$\psi_1(i) = 0.$$

Recursion:

$$\delta_t(i) = \max_{1\leq i\leq N}\left[\delta_{t-1}(i)a_{ij}\right] b_j(O_t), \qquad 2 \leq t \leq T,\quad 1 \leq j \leq N$$

$$\psi_t(i) = \arg\max_{1\leq i\leq N}\left[\delta_{t-1}(i)a_{ij}\right], \qquad 2 \leq t \leq T,\quad 1 \leq j \leq N.$$

Termination:

$$P^* = \max_{1 \le i \le N} [\delta_T(i)],$$

$$q_T{}^* = arg \max_{1 \le i \le N} [\delta_T(i)].$$

Backtracking for state sequence:

$$q_T{}^* = \psi_{t+1}(q^*_{t+1}), \qquad t = T-1, T-2, \ldots, 1.$$

where P* is the probability of the sequence being produced by each model. The model that has the greatest likelihood of producing this observation sequence defines the word class.

To test the new data image with the validated Hidden Markov Models, we used the following algorithm:

1. For each folder:
    a. For each image in the Testing set:
        i. Preprocess the image
        ii. Find the feature vector (FV) for the image [The FV matrix size is: the Number of SlidingWindows multiplied by the Number of Features (25 in our case)].
        iii. Send the FV to the "VectorQuantization" program to find the most suitable cluster for the FV and save it as QFV(Quantized FV)

    iv. Save QFV in a variable called "Data".
    v. For each word class:
        1. Send the "Data" and three selected matrices (Validated HMM) to the hmm test function (or call Matlab's hmmdecode function).
        2. Find the likelihood value and keep it.
    vi. Continue until the likelihood for all word classes are found.
    vii. Find the highest likelihood.
    viii. Assign the word class with the highest likelihood to the Recognized Word Class.
    ix. If the Recognized Word Class is equal to the Real Class, add one value to the Correctly Recognized variable.
    x. Continue until all images in the current folder are processed.

2. Continue until all the folders are processed.
3. Find the recognition rate by using the following formula:

   (Correctly Recognized / Total Number of the Images) * 100%

The above process is shown graphically in Figure 6-4.

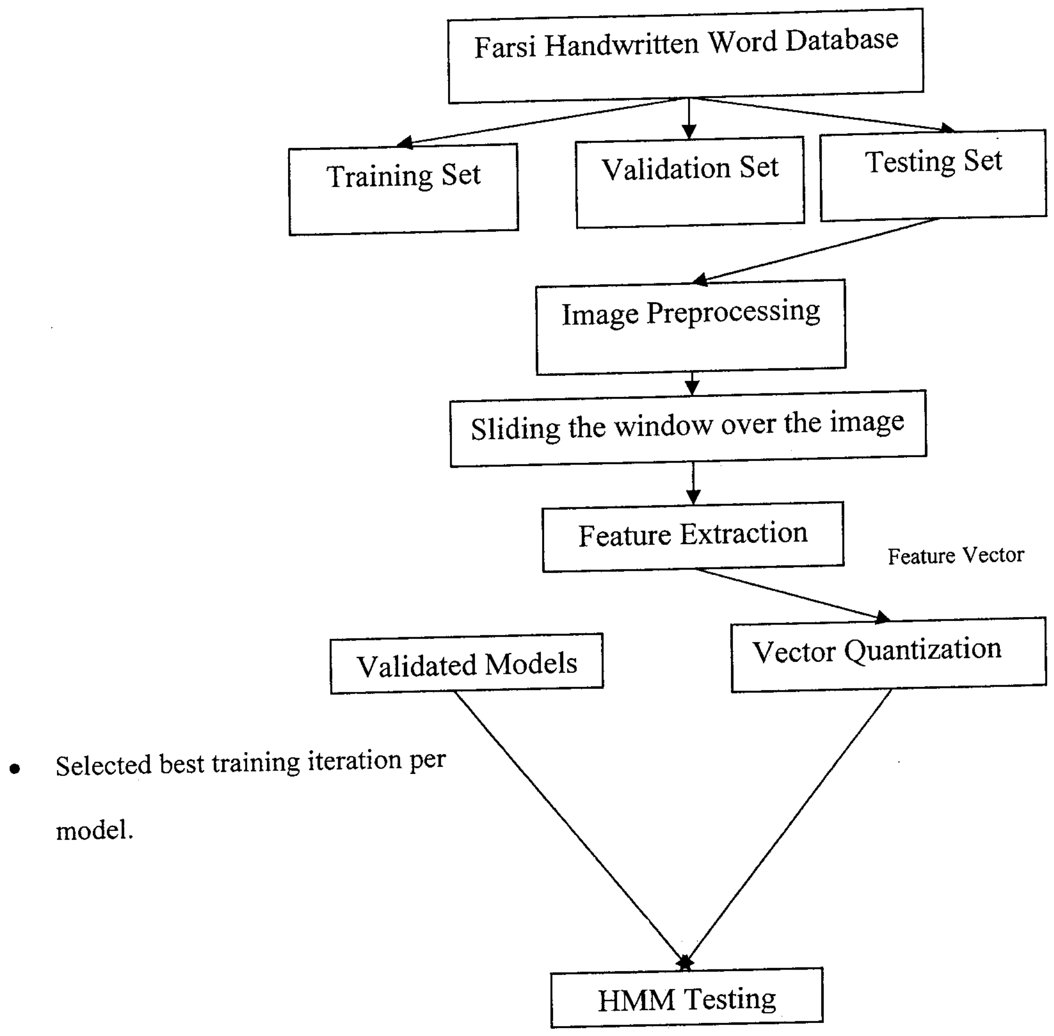


Figure 6-4: Hidden Markov Model Testing Stage.

6.10 HMM Toolboxes

Having calculated the initial estimation (as in Section 6.6), we can use the existing HMM toolboxes. There are different HMM Toolboxes. Some of them have been designed to be used with Matlab, such as Kevin Murphy's Toolbox, Mendel's Toolbox, Mathworks stats Toolbox 4.1, Zoubin Ghahramani's Toolbox, Matlab Speech Processing's Toolbox, Gnu HMM Toolbox and there are some built-in functions in Matlab for HMM. Some others have been written to be used with C++ programming language, such as HTK and MVN-HMM.

Matlab has some functions which can be used to train, validate and test the Hidden Markov Model. To train the model, we can use hmmtrain and to validate and test the data, we can use hmmdecode functions, which are described in more details as follows:
[ESTTR,ESTEMIT] = hmmtrain(seq,TRGUESS,EMITGUESS) [13].
As described previously, the initial estimation for the Transition matrix and the Confusion matrix will be sent to the function as TRGUESS and EMITGUESS, respectively.

The seq should be the quantized data matrix for all images of a class. Each row in the seq matrix will represent the clustered feature vector for an image.
The hmmtrain uses the Baum-Welch's algorithm for HMM training and can be changed to the Viterbi's algorithm. The 'Maxiterations' parameter specifies the maximum number of iterations. There are also some other parameters that can adjust the function and are described in details in Matlab Image Processing's Toolbox Document.

The hmmtrain will return the calculated Transition and Confusion matrices for each class of data. Matlab's hmmtrain and hmmdecode do not create the model that considers the vector of the initial state probabilities (π), therefore the recognition accuracy of the HMM system, modeled with Matlab's built-in functions, is less when compared to Kevin Murphy's HMM Toolbox in Matlab. Hmmdecode function is used as follows:

```
[PSTATES,logpseq] = hmmdecode(seq,TRANS,EMIS)
```

The hmmdecode can validate or test the model. The "Seq" is a one row array with the quantized values of an image's feature vectors. The TRANS and EMIS are the calculated Transition and Confusion matrices for each class. To find the most likely class for the image which has been tested, hmmdecode can find the posterior state probabilities (PSTATES) and logpseq returns the logarithm of the probability of the "Seq" by testing all existing classes' Transition and Confusion matrices. The class which gives the highest likelihood is the selected class for the tested image.

We also tried Kevin Murphy's HMM Toolbox[1] for matlab. The toolbox should be copied to the Matlab main directory, and before using it in the program, there should be a change of directory to the address where it is saved. To train the model with Kevin Murphy' s toolbox, we also needed to find the initial distribution of the states (π). To train the HMM with discrete outputs (dhmm), we used the following function:

[1] Kevin Murphy's HMM Toolbox can be downloaded for free from the following website:

http://people.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

```
[TrainedMatrices,LL, prior2, transmat2, obsmat2] = dhmm_em(seq, prior1,
TRGUESS, EMITGUESS, 'max_iter', 10)
```

The `dhmm_em` uses the Baum-Welch's algorithm to train the model. Prior1 is the π vector, TRGUESS is the initial estimation for the transition matrix and EMITGUESS is the initial estimation of the Confusion matrix. Max_iter defines the maximum number of iterations to be completed to find the local maxima. The `dhmm_em` improves our initial estimation by using 10 iterations of the Baum-Welch's algorithm. LL (t) shows the likelihood after iteration t. Therefore, we can plot the learning curve. [3635]

To validate or test the model, Kevin Murphy's toolbox offers the following function:

```
loglik = dhmm_logprob(data, prior2, transmat2, obsmat2)
```

Our motivations to choose the Hidden Markov Model for the handwriting word recognition system rather than the other classifiers such as Neural Networks or Support Vector Machines are described as follows: 1-HMMs can consider a wide variability in writing, which exists in cursive scripts such as the Farsi language and 2- HMMs can consider the whole word as an observation sequence. Therefore, no segmentation is needed for word recognition. The next chapter discusses our experimental results and the error analysis.

Chapter 7: Results and Error Analysis

As described in detail in Chapter 2, our experiments were conducted on the CENPARMI Farsi dataset with a lexicon size of 73 words most frequently used in Farsi financial documents. A holistic approach was chosen to model each word as a Hidden Markov Model. In the literature, words in small size lexicons were modeled separately, while for the large size lexicons the path discriminant method could be used to reduce the memory usage and process time.

A right to left HMM was designed to consider the nature of the Farsi script. The numbers of states were chosen based on the average number of states per class. The chosen number of states for each class is shown in Table 7-1.

Word Class		Chosen Number of states	Word Class		Chosen Number of states	Word Class		Chosen Number of states
account	حساب	41	expense	هزینه	32	nine	نه	12
amount	مبلغ	24	expire	انقضا	39	number	عدد	32
article	جنس	34	file	پرونده	36	one	یک	29
balance	مانده	29	five	پنج	20	pay	پرداخت	45
Balance2	موجودی	45	four	چهار	30	payment	قسط	29
bank	بانک	33	gallons	گالن	36	period	مدت	33
barrels	بشکه	29	gram	گرم	30	plus	اضافه	32
bill	صورتحساب	63	inches	اینچ	26	product	محصول	42
carton	کارتن	37	increase	افزایش	42	quantity	کمیت	33
cash	نقد	20	interest	سود	27	rent	اجاره	30
cc	سی سی	37	inventory	انبار	29	seven	هفت	36
Centim-eter	سانتی متر	53	issue	مورد	28	six	شش	30
cost	قیمت	35	item	کالا	26	slice	قطعه	30
credit	اعتبار	37	kilogram	کیلوگرم	46	stock	سهام	30
custom	گمرک	42	kilometer	کیلومتر	47	tax	مالیات	41
date	تاریخ	35	length	طول	28	ten	ده	14
debit	بدهی	29	letter of intent	قولنامه	39	three	سه	19

Table 7-1: The chosen number of states for each word class. (part a)

Word Class		Chosen Number of states	Word Class		Chosen Number of states	Word Class		states
decrease	کاهش	44	Liability	بدهکاری	46	tons	تن	22
delivery	تحویل	34	liter	لیتر	24	total	مجموع	33
demand	بستانکاری	54	loan	وام	21	two	دو	19
document	سند	26	meter	متر	23	volume	حجم	23
dozen	دوجین	38	milligram	میلی گرم	42	weight	وزن	31
due	سررسید	43	milliliter	میلی لیتر	44			
duty	عوارض	41	millimeter	میلی متر	43	width	عرض	31
eight	هشت	39	Money order	حواله	29			

Table 7-1: The chosen number of states for each word class. (part b)

The number of iterations to train each model was ten and the training results improved steadily using the Baum-Welch's algorithm. In our research we tried to show the importance of the Baseline-related features. The successful experiments with the high recognition rates showed that the "Distribution" and "Concavity" features could improve the recognition system performance.

7.1 Results

Our system was trained on CENPARMI's Word Training sets. Each of the 73 classes had a maximum 306 images. This approximation comes from the fact that some images had to be removed from the folders because of their noisy nature, which could mislead the classifier even after the preprocessing stage. The system was then validated on the validation sets with a maximum 104 images per class and the testing sets were not touched before the testing stage. Each testing set included a maximum 104 images per class and the results are shown in the confusion matrix. The accuracy and the reliability of the recognition experiment conducted on the Testing set are also presented in Table 7-2.

Finally, encouraging recognition rates of 98.76% and 96.02% have been obtained for the Training and Testing sets, respectively. Our designed system is reliable, robust to the noises and is reasonably fast. Speed of the Testing process is about 3 sec on a single image scanned at 300 dpi and converted to binary. The experiments executed on an Intel Duo Core CPU , 2.66 GHZ with 3.25 GB of RAM.

OUTPUT / INPUT	account	amount	article	balance	balance2	bank	barrels	bill	carton	cash	cc	centimeter	cost	credit	custom	date	debit	decrease	delivery
account	101	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
amount	0	96	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0
article	0	0	90	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
balance	0	0	0	96	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
balance2	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bank	0	0	0	0	0	94	0	0	0	0	0	0	0	0	0	0	0	0	0
barrels	0	0	0	0	0	0	94	0	0	0	0	0	0	0	0	0	0	0	0
bill	0	0	0	0	0	0	0	98	0	0	0	0	0	0	0	0	0	0	0
carton	0	0	0	0	0	0	0	0	94	0	1	0	0	0	0	0	0	0	0
cash	0	0	0	0	0	0	0	0	0	92	0	1	1	0	0	1	1	0	0
cc	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
centimeter	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0
cost	0	0	0	0	0	1	0	0	0	0	1	0	98	0	0	0	0	0	0
credit	0	0	0	0	0	0	0	1	0	0	0	0	0	102	0	0	0	0	0
custom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	102	0	0	0	0
date	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98	0	0	0
debit	0	0	0	0	0	0	0	0	0	0	0	2	1	0	0	0	98	0	0
decrease	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	100	1
delivery	0	0	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0	95
demand	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
document	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
dozen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
due	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
duty	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eight	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
expense	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0
expire	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
file	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
five	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
four	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
gallons	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
gram	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
inches	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
increase	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
interest	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
inventory	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
issue	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
item	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
kilogram	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
kilometer	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
length	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1
letter of intent	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
liability	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
liter	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
loan	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
meter	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
milligram	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
milliliter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
millimeter	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Money order	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
nine	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
number	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
one	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
pay	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
payment	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
period	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
plus	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
product	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
quantity	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
rent	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
seven	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
six	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
slice	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
stock	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
tax	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ten	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
three	0	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
tons	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
total	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
two	0	0	0	0	1	0	0	0	0	0	0	0	2	0	0	0	0	0	0
volume	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
weight	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Width	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Correct	101	96	90	96	100	94	94	98	94	92	100	100	98	102	102	98	98	100	95
Inc.	0	1	3	1	5	6	6	2	0	1	4	10	13	0	4	2	3	1	3
Total	101	97	93	97	105	100	100	100	94	93	104	110	111	102	106	100	101	101	98
Reliability	100.0%	99.0%	96.8%	99.0%	95.2%	94.0%	94.0%	98.0%	100.0%	98.9%	96.2%	90.9%	88.3%	100.0%	96.2%	98.0%	97.0%	99.0%	96.9%
Rank	1	15	42	15	52	58	58	31	1	18	46	67	73	1	44	31	38	12	40

Table 7-2 : Confusion matrix on Testing Set.(part a)

OUTPUT / INPUT	demand	document	dozen	due	duty	eight	expense	expire	file	five	four	gallons	gram	inches	increase	interest	inventory	issue	item
account	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
amount	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	1	0
article	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
balance	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
balance2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bank	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
barrels	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bill	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
carton	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cash	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cc	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
centimeter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cost	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
credit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
custom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
date	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
debit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
decrease	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
delivery	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
demand	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
document	0	91	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
dozen	0	0	102	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
due	0	0	1	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
duty	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eight	0	0	0	0	0	67	0	0	0	0	0	0	0	0	0	0	0	0	0
expense	0	0	0	0	1	0	95	0	0	0	0	0	0	0	0	0	0	0	0
expire	0	0	0	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0
file	0	0	0	1	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0
five	0	0	0	0	0	0	0	0	0	48	0	0	0	0	0	0	0	0	0
four	0	0	0	0	0	0	0	0	0	0	68	0	0	0	0	0	0	0	0
gallons	0	0	0	0	0	0	0	0	0	0	0	98	0	0	0	0	0	0	0
gram	0	0	0	0	0	0	0	0	0	0	0	0	99	0	0	0	0	0	0
inches	0	0	0	0	0	0	0	0	0	0	0	0	0	97	0	0	0	0	0
increase	0	0	0	0	0	0	0	0	0	0	0	0	0	0	94	0	0	0	0
interest	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98	0	0	0
inventory	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	99	0	0
issue	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	101	0
item	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	99
kilogram	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
kilometer	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0
length	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
letter of intent	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
liability	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
liter	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
loan	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
meter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
milligram	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
milliliter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
millimeter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Money order	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
nine	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1
number	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
one	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
pay	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
payment	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
period	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
plus	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
product	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
quantity	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rent	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
seven	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
six	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
slice	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
stock	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
tax	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ten	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0
three	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0
tons	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
total	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
two	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
volume	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
weight	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Width	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Correct	99	91	102	100	100	67	95	99	99	48	68	98	99	97	94	98	99	101	99
Inc.	2	4	13	9	5	0	3	5	2	0	1	3	0	1	0	2	2	5	5
Total	101	95	115	109	105	67	98	104	101	48	69	101	99	98	94	100	101	106	104
Reliability	98.0%	95.8%	88.7%	91.7%	95.2%	100.0%	96.9%	95.2%	98.0%	100.0%	98.6%	97.0%	100.0%	99.0%	100.0%	98.0%	98.0%	95.3%	95.2%
Rank	27	48	69	64	52	1	40	54	27	1	21	38	1	14	1	31	27	50	54

Table 7-2 : Confusion matrix on Testing Set. (part b)

OUTPUT INPUT	kilogram	kilometer	length	letter of intent	liability	liter	loan	meter	milligram	milliliter	millimeter	money order	nine	number	one	pay	payment	period	plus	product	quantity	rent
account	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
amount	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
article	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
balance	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
balance2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bank	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
barrels	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
bill	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
carton	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
cash	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
cc	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
centimeter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cost	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
credit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
custom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
date	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
debit	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
decrease	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
delivery	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
demand	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
document	0	0	0	0	1	0	0	1	0	0	2	0	0	0	0	0	0	1	0	0	0	0
dozen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
due	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
duty	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
eight	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
expense	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0
expire	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
file	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
five	0	0	0	0	0	0	1	0	0	0	0	2	1	0	0	0	0	1	0	0	0	0
four	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
gallons	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
gram	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
inches	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
increase	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
interest	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1
inventory	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
issue	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
item	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
kilogram	98	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
kilometer	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
length	0	1	83	0	0	1	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	0
letter of intent	0	0	0	97	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
liability	0	0	0	0	103	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
liter	0	0	0	1	1	85	3	1	1	0	0	2	0	0	0	0	0	1	0	1	0	0
loan	0	0	0	1	0	0	102	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
meter	1	0	0	0	0	0	0	94	0	0	2	0	0	0	0	0	0	0	0	0	0	0
milligram	0	0	0	0	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0
milliliter	0	0	0	0	0	0	0	0	0	69	0	0	0	0	0	0	0	0	0	0	0	0
millimeter	0	0	0	0	0	0	0	0	0	0	101	0	0	0	0	0	0	0	0	0	0	0
Money order	0	0	0	2	0	0	0	0	1	0	1	91	0	0	0	1	0	1	1	0	0	0
nine	0	0	0	0	1	0	0	0	0	0	0	1	58	0	0	0	0	0	2	0	0	0
number	0	0	0	0	0	0	1	1	0	0	0	1	0	97	0	0	1	0	0	0	0	0
one	0	0	0	0	1	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0
pay	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	101	1	0	0	0	0	0
payment	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	96	0	0	0	1	0
period	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	104	0	0	0	0
plus	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	96	0	0	0
product	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	101	0	0
quantity	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	97	0
rent	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	102
seven	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
six	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
slice	2	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
stock	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
tax	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ten	0	0	1	0	0	0	2	0	0	0	0	1	0	1	0	0	0	0	1	1	0	0
three	1	0	0	0	0	0	0	0	0	0	2	0	0	2	0	0	1	0	0	0	0	0
tons	0	1	0	1	0	0	1	0	0	1	1	0	0	0	0	1	0	0	0	1	0	1
total	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
two	1	0	0	0	0	0	1	0	0	0	0	0	0	2	0	0	1	0	0	0	1	0
volume	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0
weight	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Width	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0
Correct	98	99	83	97	103	85	102	94	99	69	101	91	58	97	70	101	96	104	96	101	97	102
Inc.	8	2	2	5	8	1	13	8	9	2	13	12	1	5	1	5	9	9	4	4	2	2
Total	106	101	85	102	111	86	115	102	108	71	114	103	59	102	71	106	105	113	100	105	99	104
Reliability	92.5%	98.0%	97.6%	95.1%	92.8%	98.8%	88.7%	92.2%	91.7%	97.2%	88.6%	88.3%	98.3%	95.1%	98.6%	95.3%	91.4%	92.0%	96.0%	96.2%	98.0%	98.1%
Rank	61	27	36	56	60	19	69	62	65	37	71	72	23	56	20	50	66	63	47	45	34	25

Table 7-2 : Confusion matrix on Testing Set. (part c)

OUTPUT / INPUT	seven	six	slice	stock	tax	ten	three	tons	total	two	volume	weight	width	Correct	Inc.	Total	Pct.	Rank
account	0	0	0	0	0	0	0	0	0	0	0	0	0	101	2	103	98.06%	30
amount	0	0	0	0	1	0	0	0	0	0	0	0	0	96	8	104	92.31%	61
article	0	0	0	0	0	0	0	0	0	0	0	0	0	90	5	95	94.74%	57
balance	0	0	0	0	0	0	0	0	0	0	0	1	0	96	5	101	95.05%	52
balance2	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	100	100.00%	1
bank	0	0	0	0	0	0	0	0	1	0	0	0	0	94	5	99	94.95%	54
barrels	0	0	0	0	0	0	0	0	0	0	0	0	0	94	2	96	97.92%	36
bill	0	0	0	0	0	0	0	0	0	0	0	0	0	98	1	99	98.99%	22
carton	0	0	0	0	0	0	0	0	0	0	0	0	0	94	5	99	94.95%	54
cash	0	0	0	0	0	1	0	0	0	0	0	0	0	92	7	99	92.93%	60
cc	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	100	100.00%	1
centimeter	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	100	100.00%	1
cost	0	0	0	0	1	0	0	0	0	0	0	0	0	98	4	102	96.08%	46
credit	0	0	0	0	0	0	0	0	0	0	0	0	0	102	1	103	99.03%	16
custom	0	0	0	0	0	0	0	0	0	0	0	0	0	102	0	102	100.00%	1
date	0	0	0	0	0	0	0	0	0	0	0	0	0	98	1	99	98.99%	22
debit	0	0	0	0	0	0	0	0	0	0	0	0	0	98	4	102	96.08%	46
decrease	0	0	0	0	0	0	0	0	0	0	0	0	0	100	2	102	98.04%	31
delivery	0	0	0	0	0	0	0	0	0	0	0	0	0	95	5	100	95.00%	53
demand	0	0	0	0	0	0	0	0	0	0	0	0	0	99	0	99	100.00%	1
document	0	0	0	1	1	0	0	0	1	0	0	0	0	91	12	103	88.35%	65
dozen	0	0	0	0	0	0	0	0	0	0	0	0	0	102	0	102	100.00%	1
due	0	0	0	0	0	0	0	0	0	0	0	0	0	100	2	102	98.04%	31
duty	0	0	0	0	0	0	0	0	0	0	0	0	0	100	2	102	98.04%	31
eight	0	0	0	0	0	0	0	0	0	0	0	0	1	67	2	69	97.10%	39
expense	0	0	0	0	0	0	0	0	0	0	0	0	0	95	8	103	92.23%	62
expire	0	0	0	0	0	0	0	0	0	0	0	0	0	99	0	99	100.00%	1
file	0	0	0	0	0	0	0	0	0	0	0	0	0	99	2	101	98.02%	34
five	0	0	0	0	1	0	0	0	0	0	0	0	0	48	7	55	87.27%	68
four	0	0	0	0	0	0	0	0	0	0	0	0	0	68	1	69	98.55%	28
gallons	0	0	0	0	0	0	0	0	0	0	0	0	0	98	3	101	97.03%	43
gram	0	0	0	0	0	0	0	0	0	0	0	0	0	99	1	100	99.00%	20
inches	0	0	0	0	0	0	0	1	1	0	0	0	0	97	4	101	96.04%	48
increase	0	0	0	0	0	0	0	0	0	0	0	0	0	94	1	95	98.95%	25
interest	0	0	0	0	0	0	0	0	0	0	0	0	0	98	5	103	95.15%	50
inventory	0	0	0	0	0	0	0	0	0	0	0	0	0	99	3	102	97.06%	41
issue	0	0	0	0	0	0	0	0	0	0	0	0	0	101	3	104	97.12%	37
item	0	0	0	0	0	0	0	0	0	0	0	0	0	99	3	102	97.06%	41
kilogram	0	0	0	0	0	0	0	0	0	0	0	0	0	98	0	98	100.00%	1
kilometer	0	0	0	0	0	0	0	0	0	0	0	0	0	99	1	100	99.00%	20
length	0	0	0	0	0	0	0	0	1	0	0	0	0	83	12	95	87.37%	67
letter of intent	0	0	0	0	0	0	0	0	0	0	0	0	0	97	2	99	97.98%	35
liability	0	0	0	0	0	0	0	0	0	0	0	0	0	103	0	103	100.00%	1
liter	0	0	0	0	0	0	0	0	0	0	0	0	0	85	16	101	84.16%	69
loan	0	0	0	0	0	0	0	0	1	0	0	0	0	102	2	104	98.08%	29
meter	0	0	0	0	0	0	0	0	0	0	0	0	0	94	5	99	94.95%	54
milligram	0	0	0	0	0	0	0	0	0	0	0	0	0	99	0	99	100.00%	1
milliliter	0	0	0	0	0	0	0	0	0	0	0	0	0	69	0	69	100.00%	1
millimeter	0	0	0	0	0	0	0	0	0	0	0	0	0	101	1	102	99.02%	17
Money order	0	0	0	0	0	0	0	0	1	0	0	0	0	91	12	103	88.35%	65
nine	0	1	1	0	0	1	0	1	0	0	0	0	0	58	13	71	81.69%	71
number	0	0	0	0	0	0	0	0	0	0	0	0	0	97	4	101	96.04%	48
one	0	0	0	0	0	0	0	0	0	0	0	0	0	70	1	71	98.59%	26
pay	0	0	0	0	0	0	0	0	0	0	0	0	0	101	3	104	97.12%	37
payment	0	0	0	0	0	0	0	0	0	0	0	0	0	96	6	102	94.12%	58
period	0	0	0	0	0	0	0	0	0	0	0	0	0	104	0	104	100.00%	1
plus	0	0	0	0	0	0	0	0	0	0	0	0	0	96	1	97	98.97%	24
product	0	0	0	0	0	0	0	0	0	0	0	0	0	101	1	102	99.02%	17
quantity	0	0	0	0	0	0	0	0	0	0	0	0	0	97	5	102	95.10%	51
rent	0	0	0	0	0	0	0	0	0	0	0	0	0	102	0	102	100.00%	1
seven	69	0	0	0	0	0	0	0	0	0	0	0	0	69	1	70	98.57%	27
six	0	60	0	0	0	0	0	0	0	0	0	0	0	60	2	62	96.77%	44
slice	0	0	93	0	0	0	0	0	0	0	0	0	0	93	11	104	89.42%	64
stock	0	0	0	100	0	0	0	0	2	0	0	0	0	100	4	104	96.15%	45
tax	0	0	0	0	104	0	0	0	0	0	0	0	0	104	0	104	100.00%	1
ten	0	0	0	0	0	55	1	0	0	0	0	0	1	55	14	69	79.71%	73
three	0	0	0	0	0	0	54	0	0	0	0	0	0	54	13	67	80.60%	72
tons	0	0	0	0	0	0	0	92	0	0	0	0	0	92	10	102	90.20%	63
total	0	0	0	0	0	0	0	0	104	0	0	0	0	104	0	104	100.00%	1
two	0	0	0	0	0	0	0	0	2	58	0	0	0	58	12	70	82.86%	70
volume	0	0	0	0	0	0	0	0	1	0	94	0	0	94	7	101	93.07%	59
weight	0	0	0	0	0	0	0	0	0	0	0	101	0	101	1	102	99.02%	17
Width	0	0	0	0	1	0	0	0	0	0	0	0	100	100	3	103	97.09%	40
Correct	69	60	93	100	104	55	54	92	104	58	94	101	100	6728	279	7007	96.02%	
Inc.	0	1	1	1	5	2	1	2	11	0	0	1	2					
Total	69	61	94	101	109	57	55	94	115	58	94	102	102					
Reliability	100.0%	98.4%	98.9%	99.0%	95.4%	96.5%	98.2%	97.9%	90.4%	100.0%	100.0%	99.0%	98.0%					
Rank	1	22	17	12	49	43	24	35	68	1	1	11	26					

Table 7-2 : Confusion matrix on Testing Set. (Part d)

7.2 Error Analysis

As shown in Table 7-2, the Recognition Rate was 96.02% over all the word classes. This table also shows the accuracy and reliability for each word class. Accuracy is the number of correct predictions compared with the accepted value. Reliability is defined as repeatability or consistency of a prediction. If an experiment is repeated many times and it is reliable it will give identical results. Reliability shows how trustworthy the classifier is [72].

The word classes which had the highest accurate recognition are "balance2", "cc", "Centimeter", "Custom", "Demand", "Dozen", "Expire", "Kilogram", "Liability", "Milligram", "Milliliter", "Period", "Rent", "Tax" and "Total; each with 100% recognition rate. The word classes with the most reliability are "Account", "Carton", "Cost", "Eight", "Five", "gram", "Inches", "Seven", "Total" and "Volume", each with 100% reliability. The word classes with the least accurate results are "Ten" and "Three" and the word classes with the lowest reliability are "Centimeter" and "Nine".

As we described previously in Chapter 2, Section 2.2.2, Farsi handwritten words have numerous style variations. These varieties not only can mislead the recognition system, but they also can cause humans to misrecognize some Farsi words. Generally, the errors that occurred in recognition can be classified as follows:

- Possible "Salt and Pepper" noise.
- Less coverage of the different varieties in the Training set for a word class compared to the other word classes.
- Possible similarity of the word's topology to the other words' topologies.
- The nature of the Farsi cursive handwritten words. Diacritical marks (dots) and extensions are usually not at the exact position of the word. Also, descenders or ascenders of a letter can be extended over or under one or more letters of the same word. Therefore, the sliding window may split letters from their diacritical marks and it can reduce the word recognition accuracy.
- Some compound words in our lexicon with common words such as "Centimeter", "Meter", "Kilometer" and "Millimeter" or "Liter" and "Milliliter"

For instance, the word "Five" has only 168 images in the training set, therefore the classifier is trained with less number of samples. The accuracy of the experiment on the testing set was 87.27%. This accuracy is ranked as the 68th accurately recognized word class among all 73 classes. As another example, the word "Ten" can have some handwritten samples that have similar topologies with the words "Width", "Tons", "Product", "Plus", "Number", "Money order", "Liter", "Length", "Expire", "Expense", "Credit" and "Cash". Therefore the recognition rate for this class was 79.71% which is the lowest recognition rate among the other classes. Figure 7-1 shows two variants of the words "Ten" and "Cash". As depicted in Figure 7-1, these two words can be written similarly especially at the beginning parts of the words which misled the classifier.

Also shown in Table 7-2, the word “Account” had 103 images correctly recognized while one image was misrecognized as the word “Bank”. Written samples of the words “Account” and “Bank” are shown in Figure 7-2. As depicted in Figure 7-2, they have some similarities in topologies, especially when “salt and pepper” noise was found in the word “Bank” image. They are also similar to some letters such as “ح” [H], “ب” [be] and “ک” [Kaaf] which some people write in a similar way. These similarities can mislead the classifier.

As described earlier Some words in our lexicon have common words such as “Centimeter”, “Meter”, “Kilometer” and “Millimeter” or “Liter” and “Milliliter” which may cause confusion errors. For instance, as shown in Table 7-2, the word “Meter” had 94 images correctly recognized while two images were misrecognized as the word “Millimeter”. Samples for the words “Meter” and “Millimeter” are depicted in Figure 7-3.

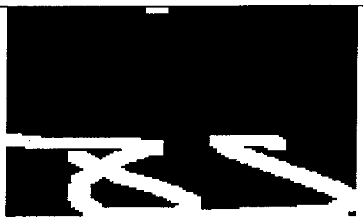
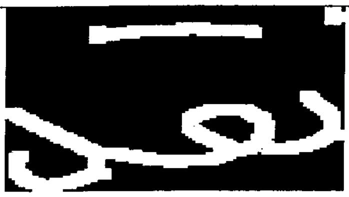
Ten	Cash
	

Figure 7-1: Two variants of the words "Ten" and "Cash".

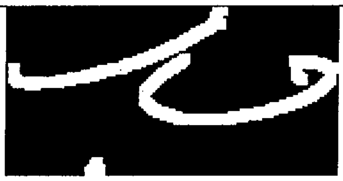
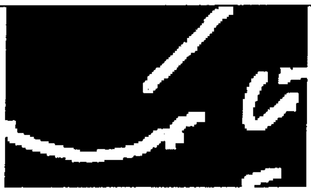
Account	Bank
	

Figure 7-2: Two variants of the words "Account" and "Bank".

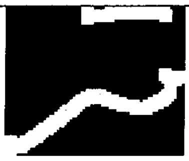
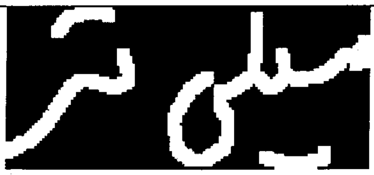
Meter	Millimeter
	

Figure 7-3: Two variants of the words "Meter" and "Millimeter ".

7.3 Related works

Comparing our research with similar research using the same classifier, the same type of script (words) for recognition and the holistic method, we obtained a better recognition result since we trained and tested our system on a better quality dataset and with more samples. We also used the validation set to optimize some of our parameters such as the number of states and the best iteration values to choose the model. Table 7-3 shows this comparison.

Research	Used Database	Lexicon Size (Words)	Number Of Images	Number Of Writers	Resolution of Images	Recognition Rate (%)	Classifier	Type of Features
M. Dehghan et al., 2000 [11]	Handwritten city names extracted from postal address blocks.	198	17,000	Unknown	Unknown	62.96	HMM	Contour-related features
M. Pechwitz et al. 2003 [40]	IFN/ENIT database	946	26,459	411	300 dpi	89	HMM	Baseline dependant features

Table 7-3: A comparison between our research and some existing research. (part a)

Research	Used Database	Lexicon Size (Words)	Number Of Images	Number Of Writers	Resolution of Images	Recognition Rate (%)	Classifier	Type of Features
Y. Kessentini et al. 2008 [41]	IFN/ENIT database (Arabic script) And IRONOFF database (Latin script)	500	36,396	Unknown	300 dpi	84.2	HMM	Contour-related and Density Features
Current Research	**CENPARMI FARSI database**	**73**	**52,286**	**400**	**300 dpi**	**96.02**	HMM	**Distribution and Concavity Features**

Table 7-3 : A comparison between our research and some existing research. (part b)

Chapter 8: Conclusions

8.1 Summary

A Handwriting word recognition system was presented for Farsi Language. In order to train, validate and test our recognition system, we had to design and gather a new Farsi dataset. This new dataset is currently unique in terms of its huge number of images (432,357 combined grayscale and binary), inclusion of all possible handwriting types (Freestyle Dates, Words, Isolated Characters, Isolated Digits, Numeral strings, Special Symbols, Documents), the variety of cursive styles, the number of writers (400) and the exclusive participation of Native Farsi speakers in the gathering of data.

The images were preprocessed before being used for the training and evaluation of the recognition system. Preprocessings were done in the Collecting Forms' level, the Extraction level and before the Feature Extraction stage.

Feature extraction was applied to reduce the dimensions of the data used for recognition. Distribution and Concavity features were chosen to represent the word. A Discrete Hidden Markov Model (DHMM) was selected as the classifier model because of the connected nature of the Farsi script and to limit the number of observation symbols.

Finally, encouraging recognition rates of 98.76% and 96.02% have been obtained for the Training and Testing sets, respectively. Some of the errors in recognition occurred due to similar topologies and numerous style varieties in Farsi handwritten words. These variations can mislead both the humans and recognition systems to misrecognize some Farsi words. The other reasons of misclassification are possible "Salt and Pepper" noise in the word image, the nature of the Farsi cursive handwritten words, common words in some compound names such as "Centimeter", "Meter", "Kilometer" and "Millimeter" or "Liter" and "Milliliter", which may cause confusion errors.

8.2 Future Works

The proposed handwritten recognition system can be improved in terms of the size of the lexicon, the coverage of multi-scripts, increasing the recognition rate and decreasing the processing time.

Some other methods could be used for large size lexicons while they decrease the process time, such as path discriminant HMM. This method considers a word as a pattern, which is classified to the word which has the maximum path probability over all possible paths [43]. Different approaches have been suggested for HMM multi-script recognition as described in [41] and could be improved. The Recognition rate and process time could also be improved by

- Using more advanced preprocessing techniques.

- Using the fusion of classifiers such as HMM-NN (Neural Networks), HMM-SVM (Support Vector Machines), etc.
- Using lexicon pruning techniques to decrease the selected words.
- Extraction of better features such as adding the Contour-related features to the Concavity and Distribution Features.
- Using the more advanced HMM with optimized Viterbi's and Baum-Welch's algorithms.
- Using more data to train the system.

References

1. P. Clawson, Eternal Iran, Palgrave Macmillan, New York, NY, USA, 2004.
2. S.W. Nicholas, Indo-Iranian Languages and Peoples, Oxford University Press, Oxford, England, 2002.
3. W.M. Thackston, An Introduction to Persian, 3rd Ed. Ibex Publishers, Washington, DC, USA, 1993.
4. J.D. Allen and J. Becker, The Unicode Standard, 5th Ed. Addison-Wesley, SC, USA, 2006.
5. S. Mozaffari, K. Faez, F. Faraji, M. Ziaratban, and S.M. Golzan, "A comprehensive isolated Farsi/Arabic character database for handwritten OCR research," in proc. International Workshop on Frontiers in Handwriting Recognition (IWFHR), Paris, France, Oct. 23-26, 2006, pp. 385-389.
6. H. Alamri, J. Sadri, N. Nobile, and C.Y. Suen, "A novel comprehensive database for Arabic Off-Line handwriting recognition," In. Proc. 11th International Conference on Frontiers in Handwriting Recognition (ICFHR 11), Montreal, Canada, 2008, pp. 664-669.
7. F. Solimanpour, J. Sadri, and C.Y. Suen, "Standard databases for recognition of handwritten digits, numerical strings, legal amounts, letters and dates in Farsi language," In proc. 10th International Workshop on Frontiers in Handwriting Recognition (IWFHR 10), La Baule, France, 2006, pp. 743-751.

8. H. Khosravi and E. Kabir, “Introducing a very large Dataset of handwritten Farsi digits and a study on their varieties,” Pattern Recognition Letters, vol. 28, pp. 1133-1141, Feb. 2007.

9. S. Mozaffari, H. El Abed, V. Margner, K. Faez, and A. Amirshahi, “IfN/Farsi-database: a database of farsi handwritten city names,” In proc. 11th International Conference on Frontiers in Handwriting Recognition (ICFHR 11), Montreal, Canada, 2008, pp. 397-402.

10. A.M. Bidgoli and M. Sarhadi, “AUT/PHCN: Azad University of Tehran / Persian handwritten city names, a very large database of handwritten Persian word,” In proc. 11th International Conference on Frontiers in Handwriting Recognition (ICFHR 11), Montreal, Canada, 2008, pp. 192-197.

11. M. Dehghan, K. Faez, M. Ahmadi, and M. shridhar, “Holistic handwritten word recognition using discrete HMM and self-organizing feature map,” In proc. IEEE Conference on Systems, Man, and Cybernetics, Nashville, TN, USA, 2000, pp. 2735-2739.

12. N. Otsu, “A threshold selection method from gray-level histograms,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, pp. 62-66, Jan. 1979.

13. Matlab Image Processing Toolbox, http://www.mathworks.com/access/helpdesk/help/pdf_doc/images/images_tb.pdf

14. P. Liu and H. Li, Fuzzy Neural Network Theory and Application, World Scientific Publishing Company, Hackensack, NJ, USA, 2004.

15. L. Lam, S.W. Lee, and C.Y. Suen, "Thinning methodologies - a comprehensive survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, pp. 869-885, Sep. 1992.

16. E.R. Dougherty, An Introduction to Morphological Image Processing, SPIE Optical Engineering Press, Washington, DC, USA, 1992.

17. R. Gonzales and R. Woods, Digital Image Processing, Addison-Wesley, London, UK, 1992.

18. The American Heritage® Dictionary of the English Language, 4th Ed, Houghton Mifflin Company, Orlando, FL, USA, 2004, http://dictionary.reference.com/browse/cursive.

19. P.J. Haghighi, N. Nobile, C.L. He and C.Y. Suen, "A new large-scale multi-purpose handwritten Farsi database," In proc. International Conference on Image Analysis and Recognition, Halifax, NS, Canada, 2009, pp. 278-286.

20. R. El-Hajj, "Arabic handwriting recognition using baseline dependant features and hidden Markov models," In proc. 8th International Conference on Document Analysis and Recognition (ICDAR), Seoul, Korea, 2005, pp. 893-897.

21. M. Haji, "Farsi Handwritten word recognition using continuous hidden markov models and structural features," M.S. thesis, Shiraz University, Iran, 2005.

22. M. Blumenstein, C.K. Cheng, and X.Y. Liu, "New preprocessing techniques for handwritten word recognition," In proc. 2nd IASTED conference on visualization, Imaging and Image Processing, Marbella, Spain, 2002, pp. 480-484.

23. D. MacKay, Information Theory, Inference and Learning Algorithms, Cambridge University Press, New York, NY, USA, 2003.

24. J.H. Ward, "Hierarchical grouping to optimize an objective function," Journal of the American Statistical Association, vol. 58, pp. 236–244, 1963.

25. R. Ravichandra, "Data mining and clustering techniques," In proc. Documentation Research and Training Centre (DRTC) workshop on Semantic Web, DRTC, Bangalore, 2003.

26. K.V. Mardia, J.T. Kent, and J.M. Bibby, Multivariate Analysis, Academic Press, Maryland Heights, MO, USA, 1979.

27. D.J. Ketchen and C.L. Shook, "The application of cluster analysis in Strategic management research: an analysis and critique," Strategic Management Journal, vol. 17, pp. 441–458, June 1996.

28. M. Mahajan, P. Nimbhorkar, and K. Varadarajan, "The Planar k-Means problem is NP-Hard," Lecture Notes in Computer Science, vol. 5431, pp. 274–285, Feb. 2009.

29. M. Inaba and N. Katoh, H. Imai, "Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering," In proc. 10th ACM Symposium on Computational Geometry, Stony Brook, NY, USA, 1994, pp. 332–339.

30. "Memory management", The MathWorks' Product Support, http://www.math works.com/support/tech-notes/1100/1106.html.

31. Concordia University, "HPC cluster specifications," http://users.encs.concordia .ca/~cirrus/1.html.

32. R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison, Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids, Cambridge University Press, New York, NY, USA, 1999.

33. L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," Proceedings of the IEEE, vol. 77, pp. 257-286, Feb. 1989.

34. A. Britto JR., "A two-stage HMM-based method for recognizing handwritten numeral strings," Ph.D. dissertation, Pontifical Catholic University of Paraná, Curitiba, Brazil, 2001.

35. K. Murphy, "How to use HMM toolbox," 1998, http://people.cs.ubc.ca/~murphyk/Software/HMM/hmm_usage.html

36. R. Boyle, "Hidden Markov models," http://www.comp.leeds.ac.uk/HiddenMarkovModels/html_dev/main.html.

37. L.R. Rabiner and B.H. Juang, "An introduction to hidden Markov models," IEEE Acoustics, Speech and Signal Processing Magazine, pp. 4–15, Jan. 1986.

38. G. Rigoll, A. Kosmala, J. Rottland, and C. Neukirchen, "A comparison between continuous and discrete density hidden markov models for cursive handwriting recognition," In proc. 13th International Conference on Pattern Recognition (ICPR'96), Vienna, Austria, 1996, pp. 39-48.

39. M. Pechwitz, S. S. Maddouri, V. Maergner, N. Ellouze, and H. Amiri, "IFN/ENIT database of handwritten Arabic words," In proc. Colloque International Francophone sur l'Écrit et le Document, (CIFED), Tunisia, October 2002, pp. 129-136.

40. M. Pechwitz and V. Maergner, "HMM based approach for handwritten Arabic word recognition using the IFN/ENIT- database," In proc. 7th International Conference on Document Analysis and Recognition (ICDAR), Edinburgh, Scotland, 2003 pp. 37-43.

41. Y. Kessentini, T. Paquet, A. Benhamadou, "A multi-stream HMM-based approach for offline multi-script handwritten word recognition," In proc. 11th International Conference on Frontiers in Handwriting Recognition (ICFHR 11), Montreal, Canada, 2008, pp. 147-152.

42. S. Alma'adeed, C. Higgens, and D. Elliman, "Recognition of off-line handwritten Arabic words using hidden Markov model approach," In proc. 16th International Conference on Pattern Recognition, Quebec, Canada, 2002, pp. 481 – 484.

43. M. Khorsheed, "Automatic recognition of words in Arabic manuscripts," Ph.D. dissertation, Churchill College, University of Cambridge, Cambridge, UK, 2000.

44. A. Amin, "Recognition of printed Arabic text based on global features and decision tree learning techniques," Pattern Recognition, vol. 33, pp. 1309-1323, Aug. 2000.

45. A. Amin, "Recognition of hand-printed characters based on structural description and inductive logic programming," Pattern Recognition Letters, vol. 24, pp. 3187-3196, Dec. 2003.

46. M. Dehghan, K. Faez, M. Ahmadi, and M. Shridhar," Unconstrained Farsi handwritten word recognition using fuzzy vector quantization and hidden Markov models, Pattern Recognition Letters, vol. 2, pp. 209-214, Feb. 2001.

47. M. Dehghan, K. Faez, M. Ahmadi, and M. Shridhar, "Handwritten Farsi (Arabic) word recognition: a holistic approach using discrete HMM," Pattern Recognition, vol. 34, pp. 1057-1065, May 2001.

48. N. N. Kharma and R. K. Ward, "A novel invariant mapping applied to hand-written Arabic character recognition," Pattern Recognition, vol. 34, pp. 2115-2120, Nov. 2001.

49. M.S. Khorsheed, "Recognising handwritten Arabic manuscripts using a single hidden Markov model," Pattern Recognition Letters, vol. 24, pp. 2235-2242, Oct. 2003.

50. R. Al-Alawi, "A neural network recognition system for isolated handwritten Arabic characters," In proc. 13th International Conference on Artificial Neural Networks And 10th International Conference on Neural Information Processing (ICANN/ICONIP), Istanbul, Turkey, 2003, pp. 262-265.

51. L.M. Lorigo and V. Govindaraju, "Offline Arabic handwriting recognition: A Survey," IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 28, pp. 712-724, May 2006.

52. A. Broumandnia and J. Shanbehzadeh, "Fast Zernike wavelet moments for Farsi character recognition," Image and Vision Computing, vol. 25. pp. 717-726, May 2007.

53. A. Broumandnia, J. Shanbehzadeh, and M. Rezakhah-Varnoosfaderani, "Persian/Arabic handwritten word recognition using M-band packet wavelet transform," Image and Vision Computing, vol. 26. pp. 829-842, June 2008.

54. M.S. Khorsheed, "Offline recognition of omnifont Arabic text using the HMM ToolKit (HTK)," Pattern Recognition Letters, vol. 28, pp. 1563-1571, Sep. 2007.

55. A. Ebrahimi and E. Kabir, "A pictorial dictionary for printed Farsi subwords," Pattern Recognition Letters, vol. 29, pp. 656-663, Apr. 2008.

56. A. Benouareth, A. Ennaji, and M. Sellami, "Semi-continuous Hmms with explicit state duration for unconstrained Arabic word modeling and recognition," Pattern Recognition Letters, vol. 29, pp. 1742-1752, Sep. 2008.

57. R.A.H. Mohamad, L. Likforman-Sulem, and C. Mokbel, " Combining slanted-frame classifiers for improved HMM-based Arabic handwriting recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, pp. 1165-1177, July 2009.

58. J. Sternby, J. Morwing, J. Andersson, and C. Friberg," On-line Arabic handwriting recognition with templates," Pattern Recognition, vol. 42, pp. 3278-3286, Dec. 2009.

59. Y. Chang, D.T. Chen, Y. Zhang, and J. Yang, "An image-based automatic Arabic translation system," Pattern Recognition, vol. 42, pp. 2127-2134, Sep. 2009.

60. B. Verma, P. Gader, and W.Chen, "Fusion of multiple handwritten word recognition techniques," Pattern Recognition Letters, vol. 22, pp. 991-998, July 2001.

61. S. Madhvanath, G. Kim, and V. Govindaraju, "Chaincode contour processing for handwritten word recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, pp. 928-932, Sep. 1999.

62. M. Mohammed and P. Gader, "Handwritten word recognition using segmentation-free hidden markov modeling and segmentation-based dynamic programming techniques," IEEE Transactions on Pattern Analysis and Machine Intelligence archive, vol. 18, pp. 548 – 554, May 1996.

63. J. T. Favata, "Offline general handwritten word recognition using an approximate BEAM matching algorithm," IEEE Transactions on Pattern Analysis and Machine Intelligence archive, vol. 23, pp. 1009-1021, Sep. 2001.

64. H. Bunke, P.S.P Wang, and H.S. Baird, Hand Book of Character Recognition and Document Image Analysis, World Scientific Publishing Company, Hackensack, NJ, USA, 1997.

65. S. Procter, J. Illingworth, and F. Mokhtarian, "Cursive handwriting recognition using hidden Markov models and a lexicon-driven level building algorithm," Computer Vision and Computer Graphics. Theory and Applications, vol. 147, pp. 332-339, Dec. 2000.

66. N. Arica and F.T. Yarman-Vural, "One-dimensional representation of two-dimensional information for HMM based handwriting recognition," Pattern Recognition Letters, vol. 21, pp. 583-592, June 2000.

67. N. Arica and F.T. Yarman-Vural, "Optical character recognition for cursive handwriting," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, pp. 801-813, June 2002.

68. E. Kavallieratou, N. Fakotakis, and G. Kokkinakis, "An unconstrained handwriting recognition system," International Journal on Document Analysis and Recognition, vol. 4, pp. 226-242, July 2002.

69. A. El-Nasan, S. Veeramachaneni, G. Nagy, "Handwriting recognition using position sensitive letter N-gram matching," In proc. International Conference on Document Analysis and Recognition (ICDAR), Montreal, Canada, 1995, pp. 577-582.

70. C.A. Higgins and D.M. Ford, "Online recognition of connected handwriting by segmentation and template matching," In proc. International Conference on Pattern Recognition (ICPR), Hague, Netherlands, 1992, pp. 200-203.

71. R.O. Duda, P.E. Hart, and D.G. Stork, Pattern Classification, 2nd Edition, Wiley Interscience Publication, Malden, MA, USA, 2001.

72. M. Butler, "Optimizing student engagement and results in the quanta to quarks option," In proc. 11th Biennial Science Teachers' Workshop, Sydney, Australia, 2004, pp. 100-110.