Recognition of Off-line Arabic Handwritten Dates and Numeral Strings

Huda Alamri

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 at

Concordia University

Montreal, Quebec, Canada

July 2009

Your file Votre référence
ISBN: 978-0-494-63149-2
Our file Notre référence
ISBN: 978-0-494-63149-2

ABSTRACT

Recognition of Off-line Arabic Handwritten Dates and Numeral Strings

Huda Alamri

In this thesis, we present an automatic recognition system for CENPARMI off-line Arabic handwritten dates collected from Arabic Nationalities. This system consists of modules that segment and recognize an Arabic handwritten date image. First, in the segmentation module, the system explicitly segments a date image into a sequence of basic constituents or segments. As a part of this module, a special sub-module was developed to over-segment any constituent that is a candidate for a touching pair. The proposed touching pair segmentation sub-module has been tested on three different datasets of handwritten numeral touching pairs: The CENPARMI Arabic [6], Urdu, and Dari [24] datasets. The final recognition rates of 92.22%, 90.43%, and 86.10% were achieved for Arabic, Urdu and Dari, respectively. Afterwards, the segments are preprocessed and sent to the classification module. In this stage, feature vectors are extracted and then recognized by an isolated numeral classifier. This recognition system has been tested in five different isolated numeral databases: The CENPARMI Arabic [6], Urdu, Dari [24], Farsi, and Pashto databases with overall recognition rates of

97.29% 97.75%, 97.75%, 97.95% and 98.36%, respectively. Finally, a date post processing module is developed to improve the recognition results. This post processing module is used in two different stages. First, in the date stage, to verify that the segmentation/recognition output represents a valid date image and it chooses the best date format to be assigned to this image. Second, in the sub-field stage, to evaluate the values for the date three parts: day, month and year. Experiments on two different databases of Arabic handwritten dates: CENPARMI Arabic database [6] and the CENPARMI Arabic Bank Cheques database [7], show encouraging results with overall recognition rates of 85.05% and 66.49, respectively.

ACKNOWLEDGEMENTS

First, I would like to express my gratitude to my supervisor, Dr. Ching Y. Suen for providing the original inspiration of my work, for his great support, and his valuable guidance and encouragement.

I would like to thank everyone else in CENPARMI group. The group has been an ideal environment, both socially and technically, in which to conduct research. Those in the group who have helped me during my master study for the last two years are numerous to mention individually. Special thanks to the manger of the group, Mr. Nicola Nobile, for his technical assistance and for his help in creating the CENPARMI Arabic Database. Also special thanks to Ms. Shira Katz and Ms. Marleah Blom for their edit-proof.

I would like to thank my husband Abdullah for all the unconditional love and support that he has been providing me with. He has been my biggest supportive and he has believed in me.

Finally, the greatest thanks go to my parents who have raised me to believe in myself and follow my dreams and to them I owe everything.

To

My father ... my mother... and my husband

Contents:

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

There has never been a time when the expression of thoughts, knowledge and information has been easily recorded, distributed and shared as it is today. The long and complicated process of generating, composing and publishing all different forms of written, visual and audible knowledge has been simplified into basic processing skills that can be easily executed using personal computers, and a wide range of readily available software [41]. The impressive growth of the digital world has led to long standing predictions that hard portable mediums like paper would no longer exist, and the age of handwriting would be outdated. Despite all that, the writing era has strongly continued its existence for many reasons, one is that the convenience of pen and paper makes it a natural medium for a lot of important tasks, and it also provides an easy way for knowledge and data to be transported, copied, annotated, and filed.

The handwriting recognition field is one of the most interesting fields in pattern recognition and computer vision. The aim of this field is to improve the human-computer interface by enabling computers to read, process and store the handwritten documents. Ultimately, the improvement of handwriting recognition provides benefits for a wide range of automatic information handling, such as: the machine reading of

bank cheques, the manual processing of tax forms, and the automatic mail sorting of machines for postal code identification in postal offices [2].

Based on the data acquisition process, handwriting recognition can be classified into on-line or off-line handwriting recognition. In on-line handwriting recognition, a computer recognizes the characters as they are drawn on an electronic input device (such as a tablet) with a stylus. Off-line recognition, on the other hand, is performed off-line after the writing has been completed. The use of electronic input devices in the on-line recognition systems enables different dynamic information to be captured, such as stroke sequence, number of strokes, pressure, acceleration, and direction of each stroke [1]. The lack of all of the above information makes off-line handwriting recognition much more challenging.

The improvement in handwritten recognition performance ultimately benefits information retrieval applications [42]. One of the most promising commercial applications for handwriting recognition is the automatic processing of handwritten bank cheques. An enormous number of cheques must be processed on a regular basis every day, worldwide. Among the different parts that should be processed on the cheque, the handwritten date is a very important part because cheques cannot be processed before the specified dates. Moreover, the date information emerges on many different kinds of forms. Therefore, the automatic recognition of handwritten dates is a fundamental problem in various recognition systems. In this thesis, we focus on the problem of Arabic off-line handwritten date recognition. For this problem, we have developed an automatic recognition system that consists of a set of different modules.

In the context of this recognition system, we have explored a number of different techniques for the segmentation and recognition of Arabic numeral strings.

1.2 Our Approach

The first step of our research was to build a comprehensive database for Arabic handwriting recognition. The database includes isolated Indian numerals, Indian numeral strings, Arabic isolated letters, a collection of Arabic words, and a free format sample of an Arabic date. A data entry form was carefully designed to collect the samples from Arabic native speakers.

Also in this thesis, we propose an automatic recognition system for Arabic handwritten dates. The recognition system consists of a set of modules. The first module is the segmentation module, where an input image is segmented into a sequence of constituents. As a part of this module, a special sub-module was developed to over-segment any constituent that is a candidate for a touching pair. By applying a 2-dimensional parameter search, the sub-module extracts two different regions that could represent each numeral. The search results in a set of extracted regions for the isolated numeral. After that, a special voting method is applied to rank the recognition results for each model. The models with the best voting results are chosen as the final segmentation results. The proposed touching pair segmentation approach was tested on different handwritten touching numerals from three different Arabic-script databases: Arabic, Urdu, and Dari. The final recognition rates of 92.22%, 90.43%, and 86.10% were achieved for Arabic, Urdu and Dari, respectively.

After the segmentation process is completed, a classification module is applied to recognize each segmented constituent. For this purpose, we have implemented a recognition system for off-line handwritten isolated numerals. Given an input image of an isolated numeral, the recognition system preprocesses the input image by applying a set of preprocessing procedures. The, it extracts a set of gradient features. The feature extraction is done by convolving each pixel neighborhood with the Roberts operators to calculate the gradient strength and gradient direction, respectively. The feature vector is generated as a composition of the gradient strengths accumulated in different directions. As a result, a feature vector of size 400 (5 vertical, 5 horizontal and 16 directional resolutions) is produced. And finally, the feature vector is passed to an SVM classifier to recognize the input image and classify it to one of the ten classes {0,1,...,9}. This recognition system has been tested in five different isolated numeral databases: The CENPARMI Arabic [6], Urdu, Dari [24], Farsi, and Pashto databases with overall recognition rates of 97.29% 97.75%, 97.75%, 97.95% and 98.36%, respectively. Finally, we have introduced a rule-based verification module that is used to improve the recognition precision by validating the outcomes from the segmentation and the classification models at different stages. Experiments on two different databases of Arabic handwritten dates: CENPARMI Arabic database [6] and the CENPARMI Arabic Bank Cheques [7], show encouraging results with overall recognition rates of 85.05% and 66.49, respectively.

1.3 Our Goal

The main objectives of this thesis are:

1. To propose a standard comprehensive database for Arabic handwriting recognition. This database could be used for different research purposes, such as Arabic word segmentation and recognition, Arabic numeral segmentation and recognition, and Arabic character recognition.
2. To investigate and explore some of the state-of-the-art feature extraction, segmentation and classification techniques and analyze their performance for the recognition of handwritten numerals.
3. To propose a high accuracy handwritten numeral recognition system based on the preceding analysis, that is invariant to numeral scaling, translation, pen thickness and most importantly, for different writing styles.
4. To propose an automatic recognition system for Arabic handwritten dates. The development of an effective date recognition system is a very challenging subject. To the best of our knowledge, there has not been any work towards the recognition of Arabic handwritten dates.

1.4 Organization of This Thesis

Chapter 2 presents an overview of the off-line handwriting recognition field. It describes a number of different state-of-the-art techniques in handwritten numeral segmentation and recognition. It also gives a general overview of Arabic numeral handwritten recognition. Finally, it describes some of the literature in handwritten date recognition. In Chapter 3, the CENPARMI Arabic Database for handwriting

recognition is presented. Different classes of the database are described in detail along with the statistics for each class. In chapter 4, a recognition system for off-line handwritten isolated numerals is described. In this chapter, preprocessing, feature extraction and classification models of the recognition system are presented. In Chapter 5, a segmentation approach for touching numeral pairs is presented. In chapter 6, our automatic recognition system for Arabic handwritten dates is described. Experimental results and analysis are presented in Chapter 7. Finally, conclusions and directions of possible future work are discussed in Chapter 8.

Chapter 2

Literature Review

2.1 Introduction

Pattern recognition is a scientific discipline for classifying data into discernible classes [44]. A very important branch of pattern recognition is handwriting recognition and it can be defined as the task of transforming a language from its spatial form of graphical marks into its symbolic representation [1]. In this chapter, we will explore some of the state-of-the-art techniques in off-line handwritten isolated numeral recognition and numeral string segmentation. We will also review the literature in handwritten date recognition techniques.

Handwritten numeral string recognition has received a high interest from the research community in the handwritten recognition field. This is due to its many applications, such as bank cheque processing, postal code recognition, tax form reading, and share certificate sorting [2]. The varieties in handwriting styles, the low quality of scanned documents, and all other factors have made the recognition of handwritten numeral strings quite challenging. Various approaches and techniques have been developed to solve the problem of numeral string segmentation and recognition [3,4,5].

2.2 Off-Line Isolated Numeral Recognition System

Since most of the numeral string recognition approaches have attempted to segment the numeral string into a sequence of hypotheses, each hypothesis represents an isolated numeral that will be further classified into one of ten numeral classes. Therefore, the isolated handwritten digit recognition module is the backbone for most of the numeral string recognition systems. In general, an off-line isolated numeral recognition system, like any other pattern recognition system, can be divided into three major phases, as shown in Figure 2.1, below. The three main phases are: the Preprocessing phase, the feature extraction phase, and the classification phase. For each of these units, we will describe the various algorithms and approaches which have been proposed and implemented.

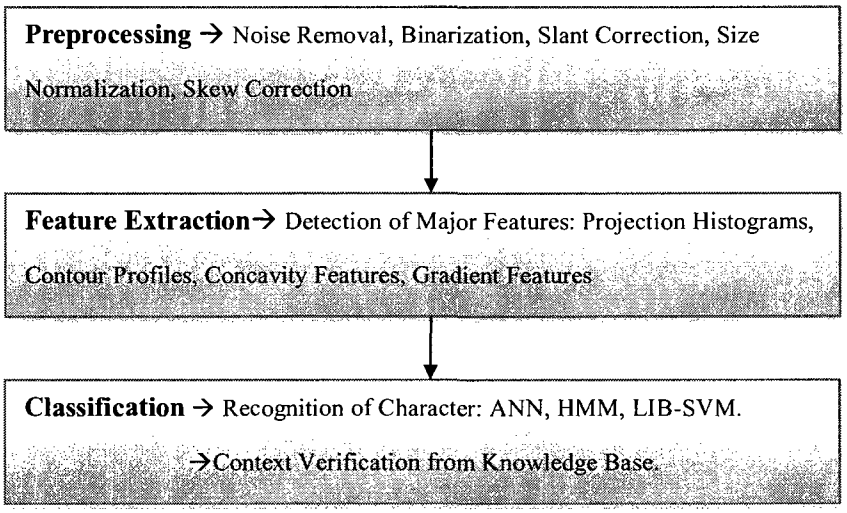


Figure 2.1: General overview of the three main phases of a pattern recognition system.

2.2.1 Preprocessing Phase

The main purpose of the Preprocessing phase is to enhance the quality of an input image by reducing the noise, since the presence of noise will alter the features of the image and thus hamper the recognition result. Different preprocessing procedures can be applied, such as size normalization, slant correction, smoothing, and thinning. In our work, we only used size normalization and smoothing in the Preprocessing phase. Here, we review some methods found in the literature for size normalization and slant correction.

Normalizing all the input images to a certain size helps reduce the shape variations without affecting the identity of each image. In 2005, Chun et al. studied the effect of size normalization in the performance of handwritten numeral recognition. In this study, experiments were conducted on the MNIST numeral database [60] using two different classification methods: Neural Networks and Support Vector Machines. The study showed that by enlarging the size of the image from 20*20 to 26*26 pixels using bilinear interpolation, the recognition performance can be considerably improved. When the normalization size was increased from 20*20 to 26*26, the substitution rate was decreased from 4.56% to 1.15%, and from 1.02% to 0.84% with ANN and SVM, respectively. Moreover, when the normalization size was increased from 26 * 26 to 41 * 41, the substitution rate continued to decrease, but the differences were much smaller [45].

Another important preprocessing procedure is slant correction. The main objective of slant correction is to reduce the script's variability and sometimes to identify some specific objects from the input image. Alceu et al. proposed a method to improve the

recognition of handwritten numeral strings by using slant normalization and contextual information to train a segmentation-based HMM recognition system. In this method, numeral strings were slant-normalized in order to reduce the overlap between adjacent numerals. A slope angel was estimated from the whole numeral string image. The numeral recognition was achieved by matching numeral HMMs against the normalized string by means of dynamic programming. Experiments on the NIST SD19 database showed a 4.71% improvement in the global numeral string recognition performance, using slant normalization without contextual information. An improvement of 9.85% was achieved with slant normalization based on contextual information [20].

2.2.2 Feature Selection and Extraction Phase

The second phase is the feature extraction phase. Feature extraction is the process of extracting a set of parameters that define the shape of the underlying character, as precisely and uniquely as possible [1]. This phase is very important because the performance of a recognition system relies very much on the features that are extracted from the images. For the feature extraction of handwritten characters, various approaches have been proposed [47]. A comprehensive study on the state-of-the-art techniques in feature extraction and classification for the problem of handwritten digit recognition can be found in [2]. Some of the efficient features for extraction from handwritten characters are: structural features, concavity features and gradient features.

Structural features such as projection histograms (see Figure 2.2), contour profile (see Figure 2.3) and zones could capture middle-level geometric characteristics, including the presence of corners and lines at several directions [48,52]. However,

these types of features are extracted from binary images, which cause jags on the stroke edges and affect the precision of the extracted features [51]. Therefore, more generalized concavity and gradient features have been used. Concavity features are able to capture high-level topological and geometrical features including the direction of bays, the presence of holes, and large vertical and horizontal strokes [33]. Gradient features represent local characteristics properly, but they are sensitive to the deformation of handwritten characters [8,46,51].

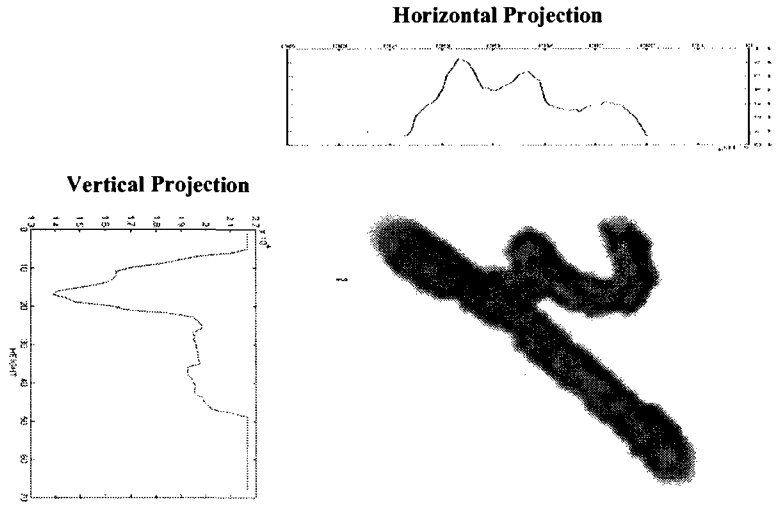


Figure 2.2: Vertical and horizontal projections of handwritten Indian digit image.

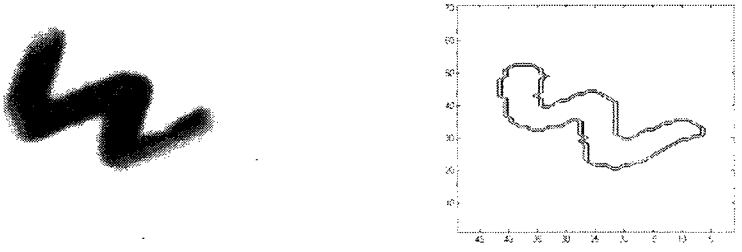


Figure 2.3: (A) A handwritten Indian digit, (B) Contour image of digit (A).

Sharma et al. [49] proposed a recognition system for handwritten Kannada numerals. Directional chain code information was extracted from the contour points of the characters using a quadratic classifier-based scheme. The bounding box of a character was segmented into blocks and the chain code histogram was computed in each of the blocks. The proposed scheme was tested on 2300 data samples. The authors obtained 97.87% and 98.45% recognition accuracy, using 64 dimensional and 100 dimensional features respectively, using the five-fold cross-validation technique [49].

Extracting different types of local and global characteristics as combined features can improve the discrimination process. Zhang et al. [52] proposed combining gradient and wavelet features for the handwritten character recognition. Three combination schemes were proposed. Experiments were conducted on two Chinese character databases, ETL8B and HKBU-SC110. The proposed feature vector achieved recognition rates of 95.53% and 93.77%, for ETL8B and HKBU-SC110 databases, respectively.

Shi et al. [8] studied the use of gradient and curvature features to improve the accuracy of handwritten numeral recognition. The proposed procedures for calculating the curvature of the gray scale input image were: curvature coefficient calculation, bi-quadratic interpolation and gradient vector interpolation, and gradient feature extraction. They composed a feature vector of the gradient and curvature by using concatenation and cross product technique. Experiments were conducted on the following handwritten numeral databases: IPTP CDROM1, NIST SD3, and SD7. The results showed that the direction of the gradient is very useful for character shape

discrimination. Recognition rates of 99.49% and 98.25% were achieved using cross product technique.

2.2.3 Classification Phase

The classification phase is the process of analyzing the input feature vectors of the character image and classifying the input image according to its corresponding character. Various classification techniques have been used for character recognition, such as: Artificial Neural Networks, Hidden Markov Model, Support Vector Machines K-Nearest Neighbor, and Decision Tree Classifiers [15,16,17].

Artificial Neural Network (ANN) can be defined as a network of weighted, additive values with nonlinear transfer functions. The terms "Neural Network" (NN) or "Artificial Neural Network" (ANN) usually refer to a Multilayer Perceptron Network, which consists of three following layers:

1. Input Layer: This layer receives a vector of predictor variable values ($x_1,\ldots,x_p$), and it distributes the values to each of the neurons in the hidden layer.
2. Hidden Layer: Arriving at a neuron in the hidden layer, the value from each input neuron is multiplied by a weight (w_{ij}) and the resulting weighted values are added together to produce a combined value (u_j). Then, the weighted sum(u_j) is fed into a transfer function (σ), which outputs a value (h_j). The final outputs from the hidden layer are distributed to the last layer (the output layer).
3. Output Layer: In this layer, the value from each hidden layer neuron is multiplied by a weight (w_{kj}), which produces weighted values. These values are added together to produce the combined weighted sum value (v_j). The weighted

sum (v_j) is fed into a transfer function (σ), which outputs a value (y_k). The y values are the final outputs of the network.

A Perceptron network with three layers is illustrated in Figure 2.4 below.

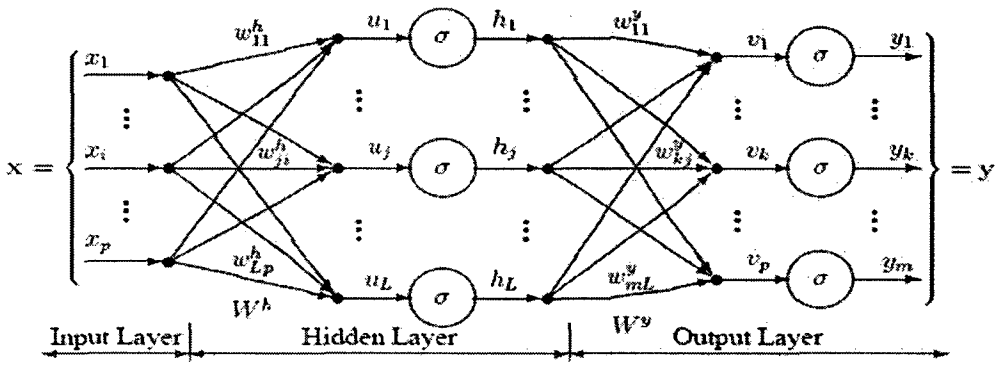


Figure 2.4: A perceptron network with three layers.

Artificial Neural Network classifiers (ANN) have been used extensively in character recognition [43, 45,46, 51,52]. These networks can be used as a combination feature extractor and classifier, where the inputs are scaled or sub-sampled input images or as a "pure" classifier where the inputs are extracted features. One problem of using neural networks in character recognition is that it is difficult to analyze and fully understand the decision making process.

J. CAO et al. (1997) proposed a hierarchical neural network architecture for handwritten numeral recognition. In this scheme, two separately trained neural networks are connected in a series. The networks use the pixels of the numeral image as input and yield ten outputs, the largest of which identifies the class to which the numeral image belongs [54].

G.Y. Chen et al. (2003) developed a handwritten numeral recognition descriptor using multiwavelets and neural networks. First, the contour of the numeral image is traced and normalized. Then, a multiwavelets orthonomal shell expansion is applied on the contour to get several resolution levels and the average. The shell coefficients are used as features to be input into a feed-forward neural network to recognize the handwritten numerals.

Another popular classifier is the Hidden Markov Model (HMM). HMMs are powerful tools in the fields of signal processing and pattern recognition, particularly in modern speech recognition systems. The application of HMMs has been extended to handwriting recognition because of the similarities between speech and handwritten texts, where both consist of symbols with ambiguous boundaries and variations in appearance. HMM does not generate a single feature vector for the whole image. Instead, it explores the relationship between consecutive segments of an input image [17]. An HMM can be considered as a nondeterministic finite state machine where each state is associated with a random function. Within a certain period of time ***t***, the signal is assumed to be in some state it and generates an observation by the random function of the state. Based on the transition probability of the current state, the underlying- Markov-chain changes from the current state to another state at time t+1. HMM can be characterized by the following probabilities:

$\Pi = \{\Pi_i\}$, Where $\Pi_i = P(q_1 = s_i)$ is the initial state probability.

$A = \{a_{ij}\}$, Where $a_{ij} = P(q_{t+1} = s_j \mid q_t = s_i$ is the state transition probability.

$T = \{\gamma_j\}$, Where $\gamma_j = P(q_t = s_j)$ is the last state probability.

$B = \{ b_j(k)\}$, Where $b_j(k) = P(o_t = v_k \mid q_t = j)$ is the symbol probability.

All of these states should satisfy the probability constraints.

The HMM study consists of the following three major problems: a scoring problem, a training problem, and recognition problem. The recognition problem requires finding the optimal state sequence of an HMM. HMMs have been used in the literature for isolated numeral recognition [56,58]. However, they have been used more in the segmentation and recognition of numeral strings as implicit segmentation-based recognition methods [20,57].

One of the most promising classification methods which deliver a state-of-the-art performance for machine learning and data mining is Support Vector Machines (SVMs). Based on the idea of VC dimension and the structural risk minimization principle [2], SVMs can determine a hyperplane that separates two classes with the largest margin between the vectors of the two classes, giving a very good generalization to new data samples. The decision boundary is basically defined in terms of the hyperplane as the following function:

$$f(x) = w \cdot \Phi(x) + b \qquad (1)$$

where Φ is a kernel that lifts the feature space into higher (possibly infinite) feature space, in which the classes are more easily separable than in the original feature space. Function (1) is a general form of the decision boundary. It can be expressed as one of the following:

$$K(a,b) = \exp\left(-||a-b||2\right), > 0 \ldots\ldots\ldots\ldots (Radial) \qquad (2)$$

$$K(a,b) = \left((a.b) + r\right)d, > 0 \ldots\ldots\ldots\ldots (Polynomial) \qquad (3)$$

$$K(a,b) = \tanh((a.b) + r) \ \ldots\ldots\ldots\ldots (Sigmoid) \qquad (4)$$

SVM is primarily a binary classifier. However, several SVM classifiers can be combined to perform a multi-class classification. There are two main techniques used to combine binary classifiers to build multi-classifiers: One-against-all, one-against-one techniques. Figure 2.5 illustrates an example of a Support Vector Machine. In this example, the decision boundary (a solid line) separates the two classes in such a way that the margin between the classes is maximized.

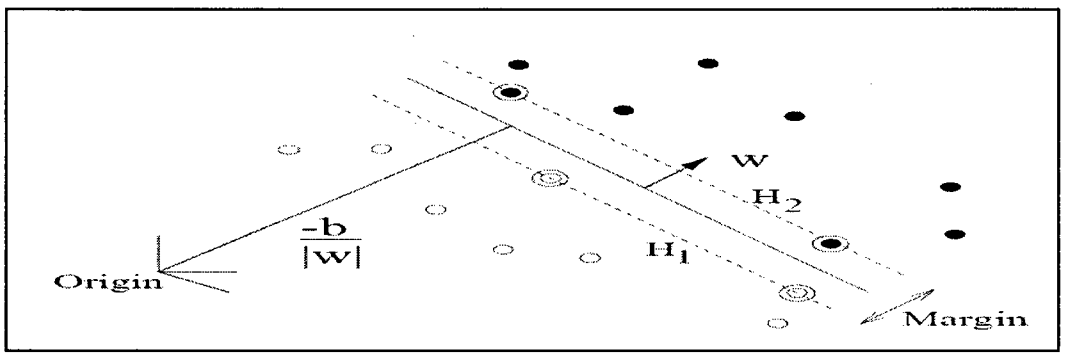


Figure 2.5: SVM the decision plane (in this case a simple line) to the two classes. Filled circles are data of class 1 and empty circles are data of class 2.

2.2.4 Indian Handwritten Numeral Recognition

In Arabic-script languages, such as Arabic, Urdu, Dari, Farsi, and Pashto, Indian numerals are used, as Arabic numerals are used in Latin scripts. As in Latin scripts, numeral strings in Arabic are written from left to right (see the Indian and the Arabic isolated numerals in Figure 2.6). For the last decade, a lot of researchers have addressed the problem of Indian handwritten numeral recognition [32,46]. More recent works have reported high recognition rates [56]. In 2004, Harifi et al. [46] proposed an asymmetrical segmentation pattern to obtain a feature vector for the recognition of

handwritten Persian/Arabic numerals. A recognition rate of 97.6 % was reported on a database of 730 digits written by 73 participants.

Sameh et al. (2009) proposed a multiple feature/resolution scheme for Indian numeral recognition using Hidden Markov Models (HHMs). The multiple features included gradient, structural, and concavity features. The proposed scheme was tested on a database of 21,120 images written by 44 writers with 48 samples per numeral and an average recognition rate of 99% was achieved [56].

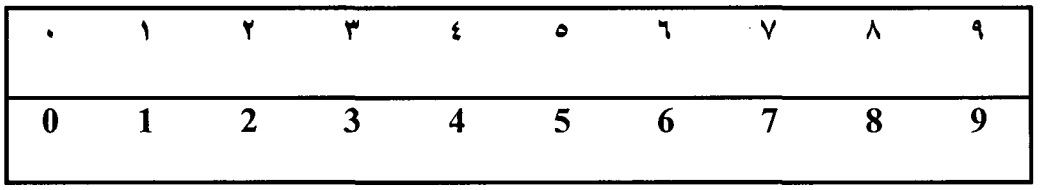


Figure 2.6: The basic isolated Indian numerals (first row), and the corresponding Arabic numerals (second row).

2.3 Handwritten Numeral String Segmentation

In general, we can classify all of the different numeral string recognition approaches into one of the following categories: segmentation-then-recognition approaches [23], segmentation-based recognition approaches [20, 22, 43], and holistic approaches [25]. In the first two categories, an input image of a numeral string is segmented into a sequence of isolated numerals and each numeral is classified to one of the ten numeral classes {0, 1,...,9}. On the other hand, the holistic approaches attempt to recognize the whole numeral string as one unit, but in this case, the huge number of classes (100 classes for only a 2-digit length) makes the segmentation process very essential.

However, the segmentation process is not a trivial task, since overlapped or touched digits may frequently exist in the numeral strings.

A lot of approaches have been proposed for the numeral string segmentation problem [2,22,23,43]. Some of these works have focused on the problem of segmenting and recognizing the touched and overlapped numerals. In 2006, Dipankar et al. [43] proposed an approach for the segmentation and recognition of unconstrained off-line Bangla handwritten numerals. A projection profile-based heuristic technique was used to segment the numerals. This method was tested on 500 Bangla numerals and it was able to correctly segment 89% of the touching numerals, with a 10% rejection ratio. Another approach was proposed by Wang et al. (2008). The authors proposed a model-based holistic approach to recognize the handwritten numeral touching pairs. A recognition rate of 93.6% was achieved using 1000 test images for the NIST SD19 database [25].

2.4 Handwritten Date Recognition

In 2003, Xu et al. proposed a recognition system for handwritten dates on Canadian bank cheques. A segmentation-based strategy was adopted in this system. Moreover, a knowledge-based module was proposed for the date segmentation, and a cursive month -word recognition module was implemented based on a combination of classifiers. The proposed system was tested on the CENPARMI_IRIS Cheque database, which includes samples of real handwritten cheques in English and French. The best overall recognition rate of the system was 62.34% with a 1.81% rejection rate. The errors of the date processing system mainly came from the segmentation, month-word

misrecognition and/or numeral misrecognition [26]. Morita et al. (2003) presented an HMM-MLP hybrid system to recognize handwritten dates written on Brazilian bank cheques. The system makes use of the HMMs to segment the different sub-fields in the date image and considers different classifiers to recognize the three obligatory subfields: the day, the month and the year subfield. Also in this system, the concept of meta-classes of digits was introduced to reduce the lexicon size of the day, the month and the year [28]. To the best of our knowledge, no work has been done toward the recognition of Arabic handwritten dates.

Chapter 3

CENPARMI Arabic Database for Handwriting Recognition

3.1 Introduction

One of the most challenging aspects of off-line handwriting recognition is finding a good database that well represents a variety of handwriting styles and contains the most important classes in the target language. To the best of our knowledge, such a database of Arabic handwriting that contains all the different classes of the language (isolated numerals, numeral strings, isolated characters, and words) is not publicly available.

The Arabic language is a member of the Semitic languages. It is ranked as the sixth most widely spoken language in the world [32]. It is spoken by more than 234 million people in over 20 countries such as Algeria, Bahrain, The Comoros, Chad, Egypt, Eritrea, Iraq, Jordan, Lebanon, Libya, Mauritania, Morocco, Oman, Saudi Arabia, Sudan, Syria, Tunisia, the United Arab Emirates (UAE), and Yemen [31]. The distribution of the countries with Arabic as the exclusive official language or as one of several official languages is shown in Figure 3.1. The Arabic language is important to the culture of many different countries, especially in the Islamic world, since it is the language of the Qur'an (the holy book for Muslims). The Arabic language of the

Qur'an is known as the classical Arabic, and from the classical one, a modern one has been adopted. It is called Modern Standard Arabic (MSA), which is used now in most Arabic official documents [31].

The modern Arabic language has 28 basic letters (see Figure 3.2). Arabic words are written in a cursive-script manner from right to left, where the letters in each word are connected in a certain manner. Therefore, the shape of a letter may change significantly depending on its position within a word. Besides the Arabic letters, Indian numerals are used in the Arabic scripts, whereas Arabic digits are used in Latin scripts.

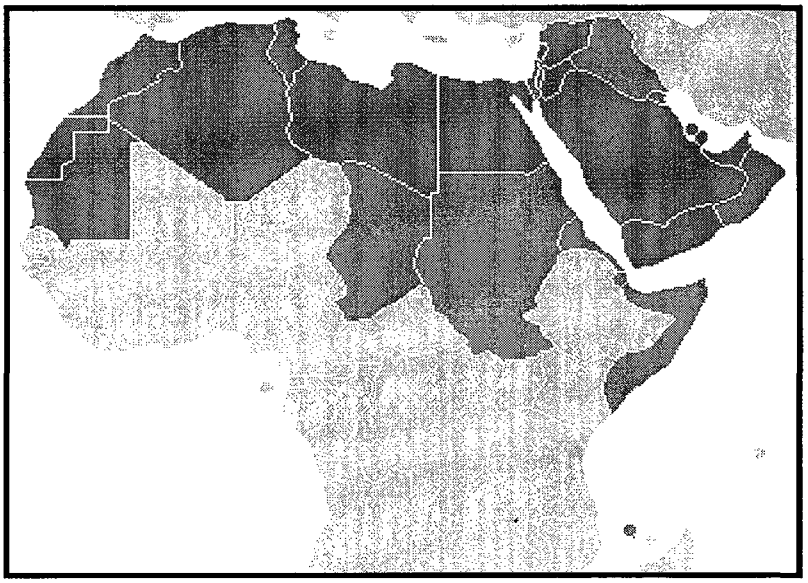


Figure 3.1: Distribution of countries with Arabic as the sole official language (green) and as one of several official languages (blue) [31].

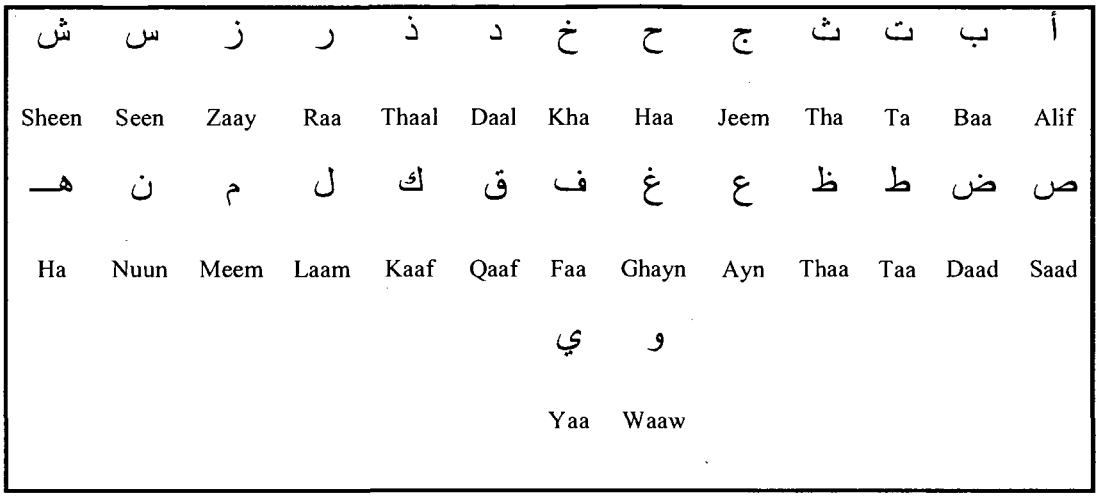


Figure 3.2: The basic isolated Arabic alphabetic letters.

This chapter represents the work towards developing a new comprehensive database for Arabic off-line handwriting recognition. The database includes the isolated Indian numerals, numerical strings, Arabic isolated letters, and a collection of Arabic key words. Finally, the database includes many free format samples of an Arabic date. A data entry form has been designed to collect the samples from Arabic native speakers.

The database is comprehensive in terms of the variety of classes, and the number of the participants involved. Also, the database has been divided into respective training, testing and validation sets.

3.2 Related Works

In the last ten years, a lot of research has been devoted to the development of databases for handwritten Latin-scripts [33,34,35]. However, there has not been much effort toward developing comprehensive databases for Arabic handwriting recognition [32].

For Arabic handwritten word recognition, the IFN/ENIT[1] database was developed in 2002. It consists of 26,549 images of Tunisian town/village names written by 411 writers [35].

Another Arabic database for handwritten words is the AHDB database, developed in 2003 by Alma'adeed et al. It includes images of words that are used to describe numbers and quantities in cheques, images of the most frequent words used in Arabic handwriting, and images of sentences used in writing the legal amount on Arabic cheques [36]. In 2003, Al-Ohali et al. of the Center for Pattern Recognition and Machine Intelligence (CENPARMI) developed an Arabic cheques database for research in the recognition of Arabic handwritten cheques. The database includes images for Arabic legal amounts, and Arabic sub-words (mainly used in writing legal amounts, courtesy amounts, and Indian digits) [38].

3.3 Data Collection

Finding a real resource to collect samples for the target Arabic sets could be quite difficult and could involve a lot of complicated Preprocessing tasks, such as removing noises and unwanted data. It could also involve a lot of segmentation processes. Therefore, the ideal solution would be to design a specific data-entry form and collect samples for those data sets from Arabic native speakers. We designed a form consists of two pages. The first page includes: a sample of an Arabic date, a total of 20 isolated numerals (2 samples for each numeral), 38 numerical strings with different lengths, 35 isolated letters with one sample of each isolated letter and the first 14 words of an

[1] The database was developed by the Institute of Communications Technology (IFN) at Technical University Braunschweing in Germany and the National School of Engineers in Tunis (ENIT).

Arabic word dataset, as shown in Figure 3.3. The second page includes the rest of the candidate words, as shown in Figure 3.4.

The forms have been filled in by participants, in two countries. The first one was filled in, in Montreal, Canada, by 100 randomly selected Arabic writers of different genders, ages, educational levels and nationalities. The second form was filled in, in Saudi Arabia, by 228 randomly selected participants. In the second form, more words were added. Participants were asked to write the samples within the box boundaries using a dark pen and to try to make the images quite readable. For this reason, the database has been divided into 3 different series. Series_1 contains the samples of the first 100 writers. Series_2 contains the samples from the last 228 writers. Finally, Series_3 contains the combination of samples from series 1 and 2. In total, there were 656 pages of filled forms.

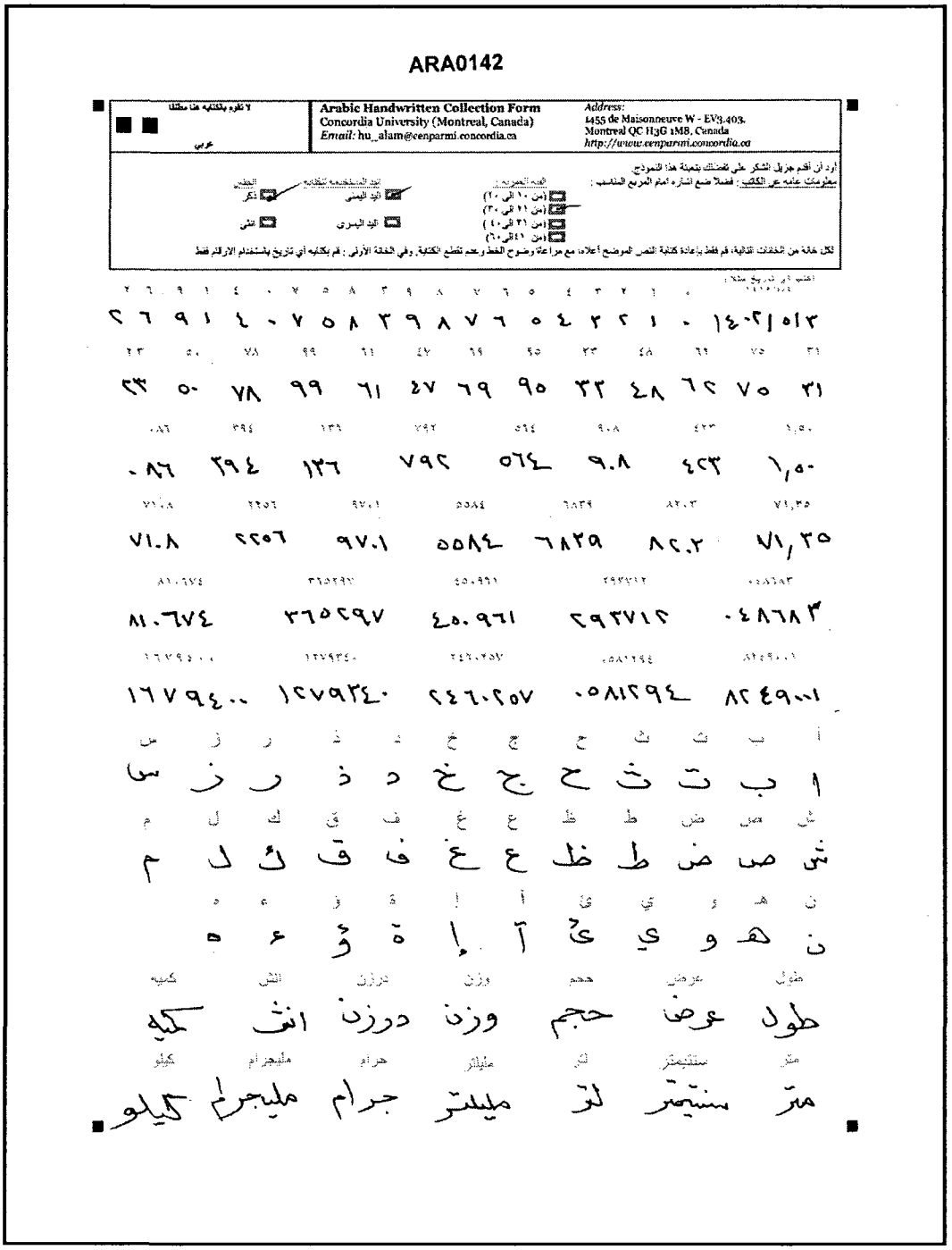


Figure 3.3: Sample of a filled form (first page).

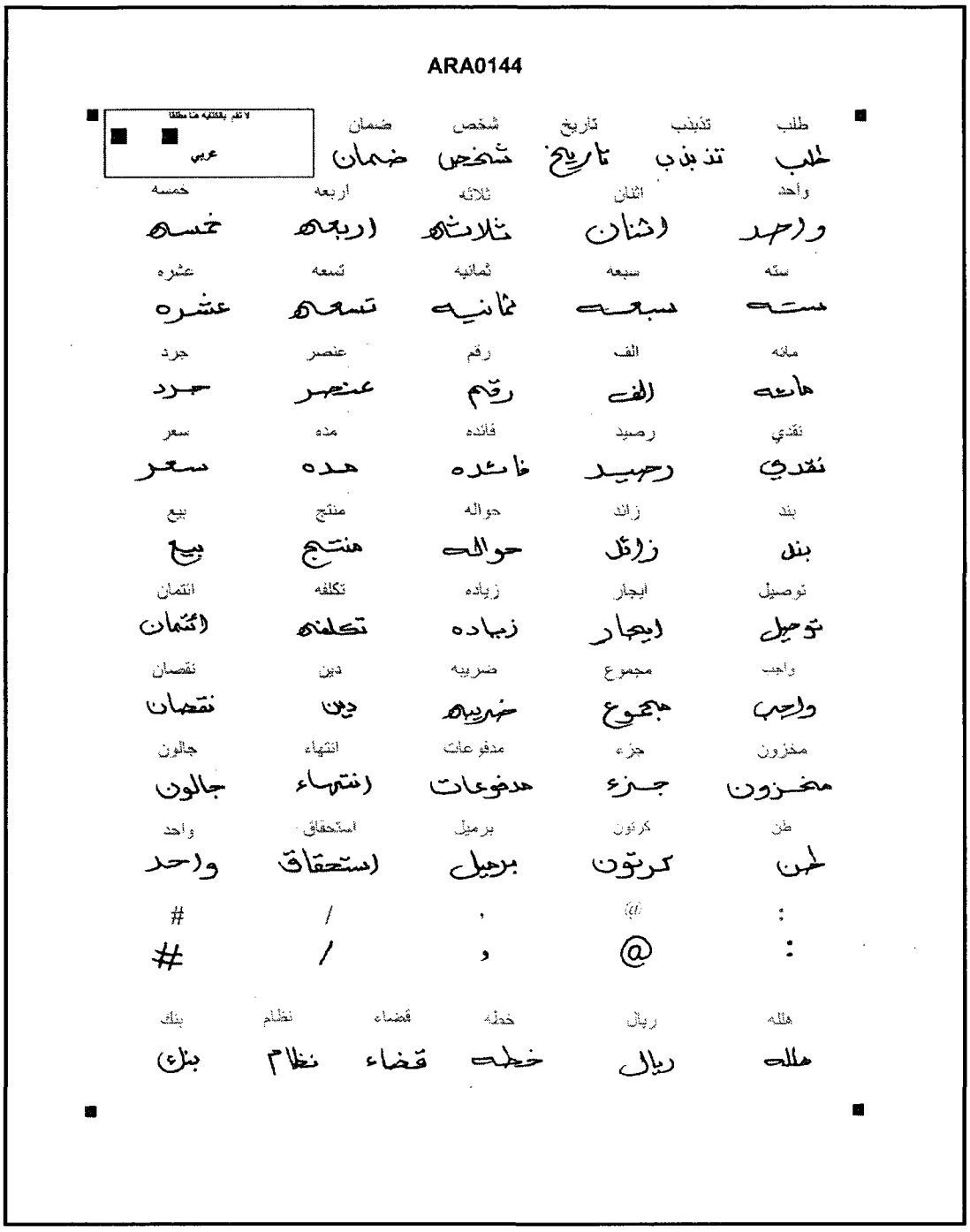


Figure 3.4: Sample of a filled form (second page).

3.4 Data Extraction and Preprocessing

After the forms were filled in by the writers, they were scanned in true color images of 300 dpi resolution. First, a special filter was applied to remove all the red borders of the boxes that contained the target images. After that, the true color forms were converted to grayscale forms. A special program was developed to automate the data extracting process. In the design stage, four black boxes were added to the corners of the forms. The program used the coordinates of these boxes first to re-skew the form if it needed to, and then to locate the target areas of the forms. After extracting all the handwritten samples, a special filter was applied to remove the salt and pepper noises.

3.5 Database Overview

The general structure of the database is shown in Figure 3.5. The database has been divided into three series: Series_1, Series_2 and Series_3. Every series has the gray and the binary versions of the datasets. Each series consists of five basic datasets: Dates, Isolated Letters, Numerical Strings, Words, and Special Symbols. In the following section, a complete description of each dataset, as well as the statistics for each dataset, are presented.

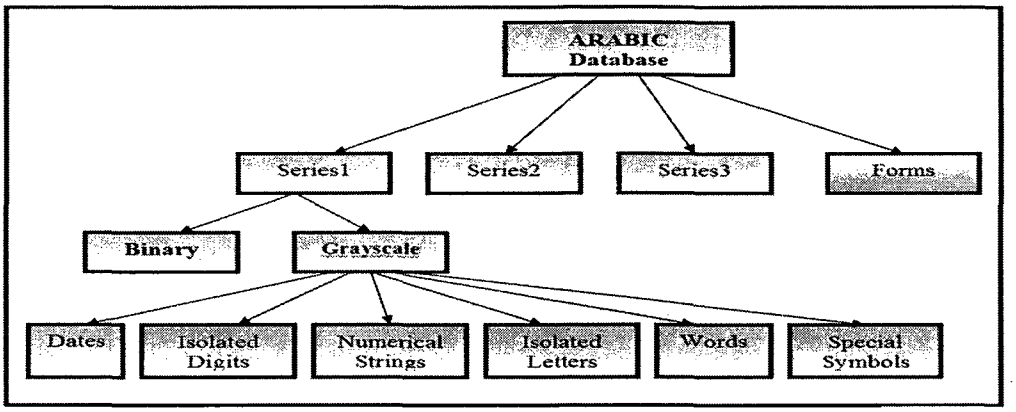


Figure 3.5: General structure of the CENPARMI Arabic database.

3.5.1 Arabic Date Dataset

The Arabic date is not always written in a uniform way. Although each date basically displays the day, the month and the year, their presentation-order and separators vary greatly. For this reason, one free-format sample of an Arabic date was included, where each participant wrote a date with the format (presentation-order and separators) and calendar that he or she would like to use. Some participants chose to write the date using the Arabic/Islamic (Hijri) calendar, while others chose to use Gregorian calendar. All the different samples were accepted as long as they were written in Indian numerals (see Figure 3.6). Some of the participants, who chose to write the date using the Arabic/Islamic calendar, added the letter "هـ" at the end of the date[2]. The statistics for the Date dataset for Series_3 are shown in Table3.1.

[2] Hijri is the name of the Arabic/Islamic calendar. First letter of this word is (هـ).

Table3.1: Statistics for the dates dataset for Series_3.

Number of Writers	Training Set	Verifying Set	Testing Set
286	172	57	57

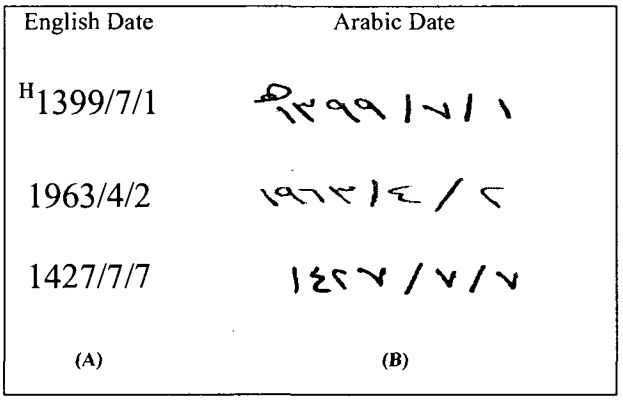


Figure 3.6: Different samples of Arabic dates (right column), and the corresponding English dates (left column). Different writers used different calendars. In the first row, the date ends with the letter "هـ" that represents the Hijri calendar.

Since the format for the data was an open free format, every writer wrote the data in a different way. Therefore, in the ground truth data, the format of the date is included. Examples of the ground truth data for the Dates dataset are shown in Table 3.2.

Table 3.2: Examples of the ground truth data for the dates dataset.

Image Name	Format	Content	Writer No.	Gender	Hand Orientation	Age
ARA0274_P01_001.tif	yyyy/mm/dd	1426/6/6	ARA0274	Female	Right-handed	41-60
ARA0109_P01_001.tif	yyyy/mm/dd	1400/1/5	ARA0109	Male	Left-handed	31-40
ARA0207_P01_001.tif	Yyyy/mm/dd	1415/4/4	ARA0207	Female	Right-handed	10-20

3.5.2 Isolated Indian Numerals Dataset

In the first two rows of the data-entry form, each participant wrote two samples of each of the basic isolated Indian numerals (see Figure 3.7). The participant also wrote samples of different numeral strings with different lengths. Within these strings, each isolated numeral was repeated from 13 to 17 times in different positions: at the beginning, at the middle and at the end of a string. In order to get an advantage from having this collection of numerical strings, a segmentation algorithm was developed to extract the isolated numerals from the numerical strings. The extracted isolated numerals were added to the isolated numerals dataset, which helped to expand the size of the dataset. Statistics for this dataset from Series_3 are shown in Table 3.3.

A common problem in the handwritten Indian numerals is the ambiguity between the numerals 2 and 3 (see Figure 3.8). Both of the two numerals have a very similar shape.

Handwritten Indian Digits	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Printed Indian Digits	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Equivalent Arabic Digits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

Figure 3.7: The basic isolated Indian numerals.

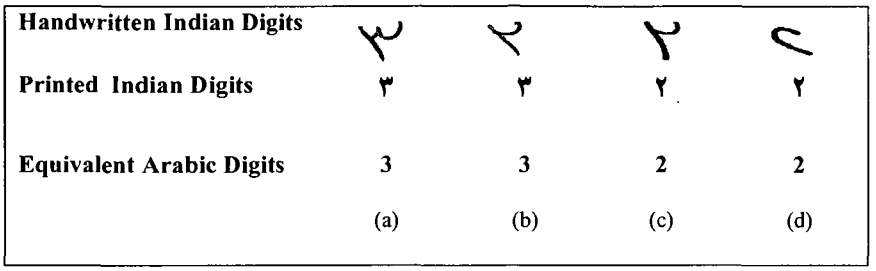


Figure 3.8: Different handwritten samples of the Indian numerals 3 and 2. In columns (b) and (c), we can see the ambiguity between 2 and 3, where both numerals have very similar shapes.

Table 3.3: Statistics for the isolated Indian numerals dataset for Series_3.

Number of Writers	Training Set	Verifying Set	Testing Set
328	28000	9400	9400

3.5.3 Numerical Strings Dataset

Each participant wrote 38 samples of different numerical strings with different lengths: 2, 3, 4, 6, and 7 digits per string. All of the digits have been included in those numerical strings in all the different positions. In addition to the samples of the numeral strings, two decimal numbers were included (see Figure 3.9). Samples of numerical strings are shown in Figure 3.10 and statistics for this dataset are shown in Table 3.4. This database is divided into two folders: Integer_Folder and Real_Folder. In the first folder, different numerical strings with different lengths are included. In the second one, decimal numbers with the lengths 3 and 4 are included.

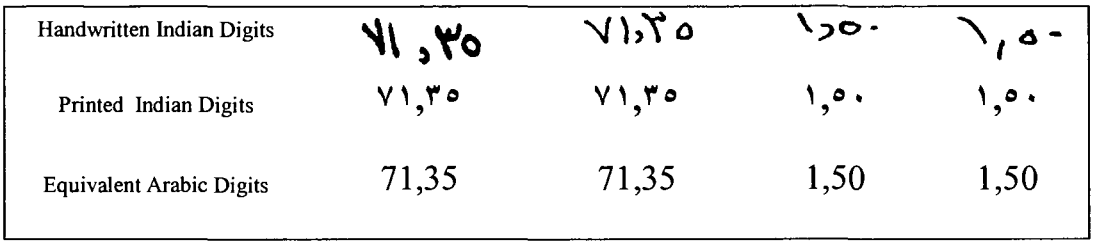


Figure 3.9: Samples of handwritten numeral strings with decimal points.

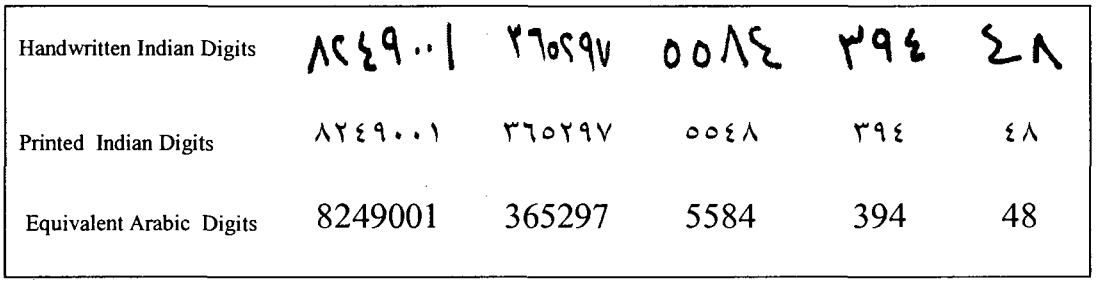


Figure 3.10: Samples of integer strings with different lengths.

Table 3.4: Statistics for the numerical strings dataset for Series_3.

Number of Writers	Training Set	Verifying Set	Testing Set
328	8033	2702	2704

3.5.4 Arabic Isolated Letters Dataset

The Arabic alphabet consists of 28 basic letters and there are no distinct upper and lower case letter forms. Words in Arabic are written in a cursive matter, where most of the letters are directly connected to the letter that immediately follows. A few letters do not connect to the following letter, even in the middle of a word. Due to this connectivity manner of the Arabic script, each letter can have up to four distinct forms, based on its position within a word or a group of letters: at the beginning, in the middle, at the end, or isolated [7] (see Figure 3.11).

In the Arabic script, some letters are usually accompanied by important components called Harakat (see Figure 3.12). Harakat is a group of vocalization diacritics that mark vowels and other sounds that cannot be represented by Arabic letters. One of the most important vocalization diacritics is called the Hamzah (see Figure 3.14), which indicates a glottal stop. The Hamzah can appear sometimes by itself or over other letters [32].

Some Arabic letters can be written in different styles. Therefore, collecting samples of those different styles is significantly important. As a result, the data entry form includes one sample of the 34 Arabic isolated letters and one sample of Hamzah. Samples of those letters are shown in Figure 3.13, and statistics for this dataset are shown in Table 3.5.

IPA	Value	Name	Final	Medial	Initial	Isolated	IPA	Value	Name	Final	Medial	Initial	Isolated
[ɖ]	ḍ	ḍād	ض	ضـ	ضـ	ض	[ʔ]	ʾ(a)	alif	ا	—	—	ا
[ʈ]	ṭ	ṭā'	ط	طـ	طـ	ط	[b]	b	bā'	ب	بـ	بـ	ب
[ʐ]	ẓ	ẓā'	ظ	ظـ	ظـ	ظ	[t]	t	tā'	ت	تـ	تـ	ت
[ʕ]	ʿ	ʿayn	ع	عـ	عـ	ع	[θ]	th	thā'	ث	ثـ	ثـ	ث
[ɣ]	gh	ghayn	غ	غـ	غـ	غ	[ʒ]	j	jīm	ج	جـ	جـ	ج
[f]	f	fā'	ف	فـ	فـ	ف	[ħ]	ḥ	ḥā'	ح	حـ	حـ	ح
[q]	q	qāf	ق	قـ	قـ	ق	[x]	kh	khā'	خ	خـ	خـ	خ
[k]	k	kāf	ك	كـ	كـ	ك	[d]	d	dāl	د	—	—	د
[l]	l	lām	ل	لـ	لـ	ل	[ð]	dh	dhāl	ذ	—	—	ذ
[m]	m	mīm	م	مـ	مـ	م	[r]	r	rā'	ر	—	—	ر
[n]	n	nūn	ن	نـ	نـ	ن	[z]	z	zāy	ز	—	—	ز
[h]	h	hā'	ه	هـ	هـ	ه	[s]	s	sīn	س	سـ	سـ	س
[w]	w	wāw	و	—	—	و	[ʃ]	š	shīn	ش	شـ	شـ	ش
[j]	y	yā'	ي	يـ	يـ	ي	[s]	ṣ	ṣād	ص	صـ	صـ	ص

Figure 3.11: Isolated, initial, middle, and final shapes for the Arabic letters.

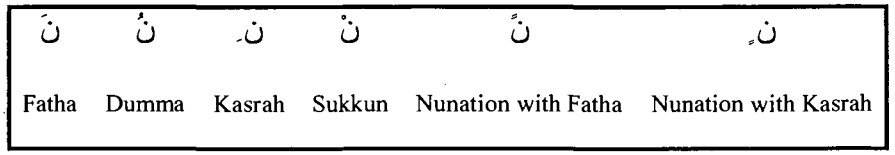


Figure 3.12: The basic Arabic diacritical marks called Harakat.

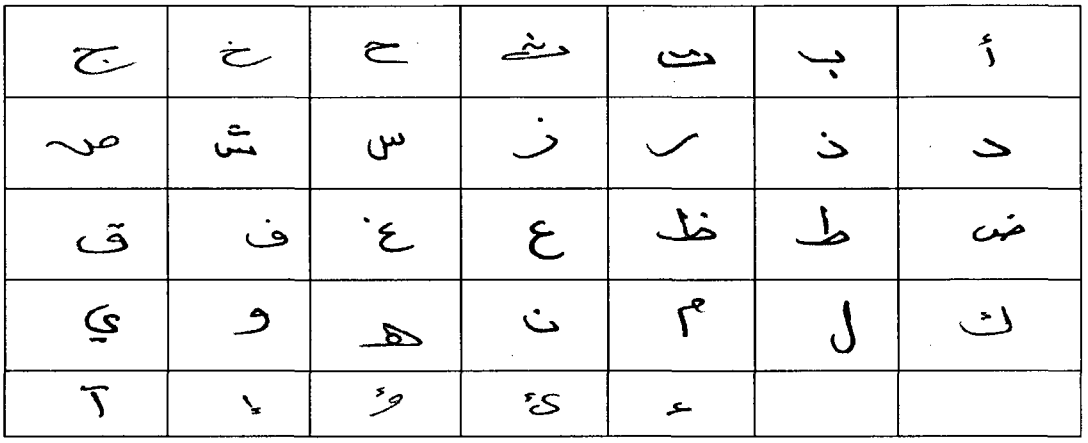


Figure 3.13: Sample of handwritten Arabic isolated letters.

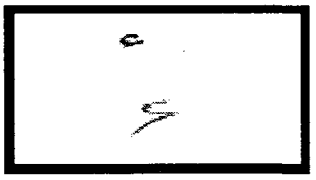


Figure 3.14: Sample of printed Hamzah (first row), and handwritten Hamzah (second row).

Table 3.5: Statistics for Arabic isolated letters dataset for Series_3.

Number of Writers	Training Set	Verifying Set	Testing Set
328	12693	4367	4366

3.5.5 Arabic Words Dataset

A collection of 70 Arabic words were selected to form the Arabic word dataset. The aim of creating this dataset was to target a new group of words that have never been collected in any previous Arabic handwriting database.

The dataset includes the following: weights, measurements, and currencies. Samples of some of these Arabic words are shown in Figure 3.15. The currencies used in this database are the Saudi Arabian Riyal (SAR), and the Hallalh (1 Riyal =100 Hallalh). Statistics for this dataset are shown in Table 3.6 This dataset could provide new challenges to the recognition of Arabic handwritten word recognition.

Printed	Printed	Handwritten
Sale	بيع	
Total	مجموع	
Price	سعر	
Cost	تكلفه	
Credit	ائتمان	
Gram	جرام	
Tax	ضريبه	
Kilo	كيلو	
Cash	نقدي	

Figure 3.15: Samples of Arabic word dataset: the printed English and Arabic words, and the handwritten Arabic words.

Table 3.6: Statistics for Arabic words dataset for Series_3.

Number of Writers	Training Set	Verifying Set	Testing Set
328	6790	2275	2310

3.5.6 Special Symbols Dataset

The data-entry form includes general symbols that may appear in any Arabic document. Although these symbols are non-language related, they were included in the Arabic database in order to capture the style of writing them from Arabic native

speakers. These samples include: comma (,), colon (:), at (@), slash (/), and no. (#) (see Figure 3.16). Statistics for this dataset are shown in Table 3.7.

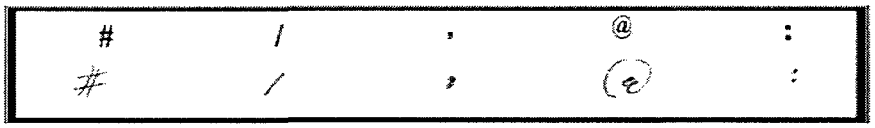


Figure 3.16: Samples of the general symbols.

Table 3.7: Statistics for symbols dataset for Series_3.

Number of Writers	Training Set	Verifying Set	Testing Set
328	980	330	330

3.5.7 The Ground Truth Data

In the final structure of our Arabic database, each folder that contains handwritten samples also contains the ground truth data file for these samples. The ground truth data file includes the following information about each sample: image name, content, number of CCs (Connected Components), writer number, writer age, writer gender, and writer handwriting orientation (left-handed or right-handed). For the best of our knowledge, there is no publicly available Arabic handwriting database that includes the writer, gender, age, and hand-orientation in the ground truth data. Examples of the ground truth data for the dates dataset are shown in Figure 3.17.

Image Name	ARA0274_P01_001.tif	ARA0109_P01_001.tif
Format	yyyy/mm/dd	yyyy/mm/dd
Content	1426/6/6	1400/1/5
Writer No.	ARA0274	ARA0109
Gender	Female	Male
Hand Orientation	Right-handed	Left-handed
Age	41-60	31-40
Length(No. of CCs)	10	10
Image Type	DATE	DATE

Figure 3.17: Two examples of the ground-truth data for the date dataset.

Chapter 4

Isolated Digit Recognition

4.1 Introduction

This chapter describes our implementation of a recognition system for handwritten isolated digits. In general, any recognition system for offline handwritten characters consists of three main units as described in Chapter 2: a Preprocessing unit, a feature extraction unit, and a classification unit. Some recognition systems could also include a segmentation unit. Our recognition system includes the first three units.

4.2 Preprocessing

For our isolated digit recognition system, the Preprocessing unit involves the following three main Preprocessing stages:

1. Median and Gaussian Filters are applied to smooth the images and to remove the salt and pepper noise.
2. The real boundary is localized around the digit image, to eliminate the unnecessary bordering white space.
3. All of the images in the training and the testing sets are normalized to a standard size (64-by-64 pixels) while preserving the aspect ratio of the original size. Figure 4.1 summarizes these Preprocessing steps.

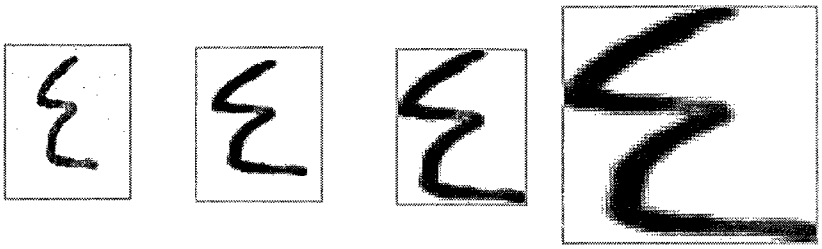


Figure 4.1: The preprocessing steps: (a) Original image (b) Smoothed image (c) Real boundary is localized, and (d) Resized Image.

4.3 Feature Extraction

The performance of any character recognition system largely depends on the feature extraction approach. Various approaches have been proposed for the character recognition problem, such as extraction of profile structure features, curvature features, concavity features, and gradient features [4,8,9]. Gradient features are high-resolution directional features that can be extracted from grayscale images. They are extracted by computing the magnitude and direction of the greatest change in intensity in a small neighborhood of each pixel [2]. Many experiments have shown that the gradient features are very significant features for handwritten character recognition because they yield high performances [8,10].

To extract the gradient features from the preprocessed grayscale images, we used Roberts Cross operators (Figure 4.2) [11,12]. Roberts Cross operators perform a simple, quick to compute, 2-D spatial gradient measurement on a grayscale image. They thus highlight regions of high spatial frequency which often correspond to edges.

Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient from the input [2].

$$\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

Figure 4.2: Roberts cross gradient operators.

The extraction of gradient features is calculated as follows: Given an input image *f(i,j)* of size M×N, each pixel neighborhood is convolved with the Roberts operators to determine the Δu and Δv components, respectively:

$$\Delta u = f(i+1,j+1) - f(i,j) \qquad (2)$$

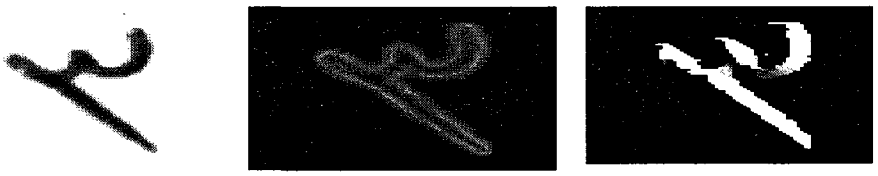
$$\Delta v = f(i+1,j) - f(i,j+1) \qquad (3)$$

Afterwards, we can calculate the gradient strength and direction in each pixel *f(i,j)*, as follows:

$$\textit{Direction} \quad \theta(i,j) = \tan^{-1}(\Delta v / \Delta u) \qquad (4)$$

$$\textit{Strength} \quad f(i,j) = (\Delta v)^2 + (\Delta u)^2 \qquad (5)$$

Examples of the gradient magnitude and direction are depicted in Figure 4.3, below.



(a) (b) (c)

Figure 4.3: (a) Grayscale image, (b) Gradient magnitude, and (c) Gradient direction.

After calculating the gradient strength and direction for each pixel of the grayscale image, the feature vector is generated. The feature vector is a composition of the gradient strengths accumulated in different directions, as described in the following steps:

1. The gradient directions calculated in (3) are quantized to 32 different levels with intervals equal to $\pi/16$
2. After normalizing the grayscale image, we divide it into zones, 9 vertical and 9 horizontal, for a total of 81 zones. For each zone, the gradient strength accumulated in each direction is calculated separately to produce local spectra of directions for each zone (81 local spectra).
3. By this step, we will have a feature vector of size $9 \times 9 \times 32$. The next step is to normalize the size of the feature vector. First, the spatial resolution is reduced from 9×9 to 5×5 using a 5×5 Gaussian filter. Second, the local spectra is reduced from 32 to 16 by down sampling with a weight vector $[1\ 4\ 6\ 4\ 1]^T$.
4. To make the distribution of the features Gaussian-like, the variable transformation $y=x^{0.4}$ is applied [13,14].

As a result, a feature vector of size 400 (5 vertical, 5 horizontal and 16 directional resolutions) is produced.

4.4 Classification

For the problem of isolated digit classification, we chose the SVM classifier. SVMs are able to outperform other classifiers in different real-world applications, such as text

categorization, handwriting recognition, etc. [18]. The kernel function chosen for this classifier was a Radial Basic Function (RBF).

$$RBF = e^{-\lambda \|x-y\|^2} \quad (6)$$

Before we can train the SVM classifier, we had to choose the two optimal parameters λ and c, where c was a penalty parameter of the error term and λ was a parameter in RBF. We used v-fold cross-validation to find the best parameter λ and c. In v-fold cross validation, the training set is first divided into v subsets of equal size. Sequentially, one subset is tested using the classifier that was trained on the remaining v-1 subsets. Thus, each instance of the whole training set is predicted once so that the cross-validation accuracy is the percentage of data which is correctly classified. Another advantage of the cross-validation procedure is that it can prevent the overfitting problem.

After finding the best values for the two parameters λ and c, we used them to train the whole training set, using the feature vectors provided by the feature extraction module described in the previous section.

Chapter 5

Segmentation of Handwritten Numeral Touching Pairs

5.1 Introduction

Segmentation can be defined as the process of dividing an image into regions, where each region contains a single object or a group of objects. Different methods have been developed for the segmentation of handwritten numeral strings [20, 21, 22, 23]. Among all the different challenges in the segmentation of handwritten numeral strings, the segmentation of touched or overlapped numerals is the most difficult one to solve.

As mentioned earlier in Chapter 2 (Section 2.3), Wang et al. [25] proposed a model-based holistic approach to recognize handwritten numeral pairs. Given an input image of a numeral pair, a set of models of this numeral pair were generated as the combinations of the two corresponding isolated numerals. Each numeral was modeled as a set of polygonal lines. The best recognition rate of 93.6% was achieved on 1000 test images from the NIST SD19 database.

In our work, we focus on solving the problem of segmenting numeral strings that are completely touching and cannot be segmented through basic connected component

analysis[3]. Given an input image of a touching pair, we applied the model generation approach as presented in [25]. The two regions that well represented the digits in the image were individually detected by searching a 2-dimensional parameter space. After that, the detected digits were scaled and preprocessed to be ready for the segmentation stage and later for the recognition stage. For classification, we applied the isolated digit classifier described in Chapter 4. To test our method, we used a subset of The CENPARMI Arabic, Dari, and Urdu Numeral String Databases [6,24]. The subsets included only samples of complete touching digits. Examples of this set are shown in Figure 5.1.

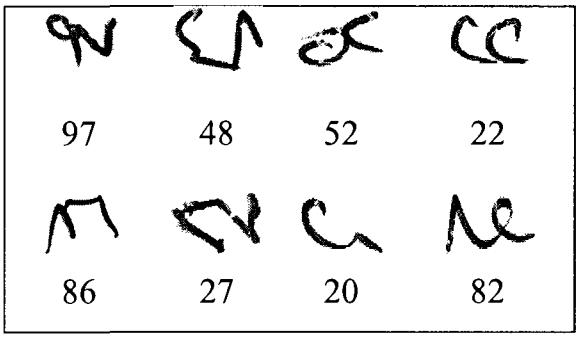


Figure 5.1: Samples of touched pairs from the CENPARMI Arabic numeral dataset and their equivalent Latin labels.

5.1. Segmentation of Numeral Touching Pairs

Segmentation of touching pairs begins by extracting the bounding box to eliminate the unwanted white space area around the image, reducing the space to be searched, and placing the touching digits in the center of the image. Then, we search for the areas

[3] See Chapter 6 for details on Connected Component Analysis.

that represent each digit individually. We can think of each digit in the image as surrounded by a rectangular box. In order to find a rectangular box in a 2-dimensional space, we need to determine its dimensions (height and width). For this purpose, we consider two parameters. Let α be first parameter to represent the ratio of the first digit width (F_w) to the whole image width (I_w). Let β, the second parameter be the horizontal distance between the first digit and the second one. Using a set of values for α and β, we can calculate the dimensions for the rectangular boxes [25]. As shown in Figure 5.2, for an image with (I_w) width and (I_h) height, we calculate the first digit width (F_w), and the second digit width (S_w) as follows:

$$F_w = \alpha . I_w \qquad (5)$$

$$S_w = I_w - \alpha . I_w - \beta \qquad (6)$$

The height of the bounding box around each digit can be defined as the vertical distance between the lower-pixel and the higher-pixel in the corresponding area of the image (Figure 5.2). Accordingly, by searching different values in (αi, βj), we generate segmented candidate images. For feasible computation a set of five values was chosen for each parameter. The chosen sets were: [0.3, 0.4, 0.5, 0.6, and 0.7] for α, and [-6, -3, 0, 3, and 6] for β. Therefore, 25 different models are extracted to individually represent the two segmented digits. We put all images with the same (αi, βj) in the same folder as one model for future recognition.

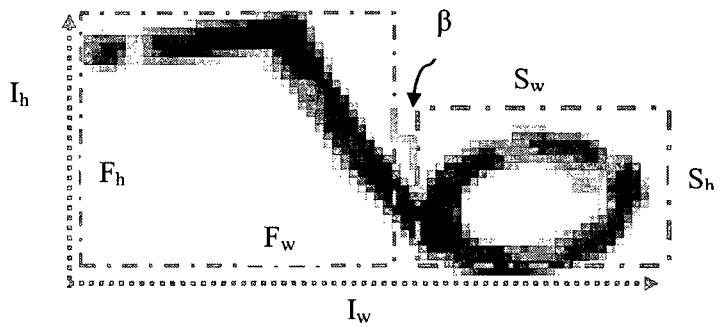


Figure 5.2: Example of two rectangular boxes representing the digits in a touching pair.

In order to test the proposed approach, we conducted the recognition experiments on touching numeral pairs from the numeral string datasets. We selected only the touching pairs from the datasets. We applied the segmentation method based on connected component analysis and selected only the samples that could not be segmented using this method. Details about the used datasets are presented in the following section.

In each model, all segmented images are recognized by the isolated numeral classifier presented in Chapter 4. Since the recognition output for each image has confidence values (probabilities) on all classes, post processing based on the given probabilities in different models is able to verify and improve the final recognition rates. When the highest probability of the given input in all 25 models is greater than a certain threshold (*Th)*, the final result obeys on the recognition result with the highest probability. Otherwise, the final recognition result is based on ranking the outputs of all the other models using different ranking schemes. Finally, the output with the highest rank is chosen. One ranking scheme ranks the results based on the majority

votes *(M_V)*. It chooses the result with the highest number of votes from 25 models. The second ranking scheme is to consider the estimated probability from each result in different models and choose the result with the highest probability *(H_P)* as the final one. The third voting scheme is a combination of both of the probability estimation and the majority vote *(H_P + M_V)*.

Experiments and results for the touched numeral segmentation are presented in Chapter 8.

5.2 Datasets of Numeral Touching Segmentation

In order to test the proposed approach, we applied it to Arabic, Urdu, and Dari CENPARMI Numeral String Databases [6, 24]. Since non-touching is not an issue for our segmentation approach, we only selected the touching pairs from the numeral string databases. They were selected by applying a segmentation method based on Connected Component Analysis (CCA) on the numeral strings databases. Those which could not be segmented by CCA were selected. A total of 721 touching pairs were found in the databases, including 132 pairs from the Arabic database, 400 pairs from the Dari database, and 189 pairs from the Urdu database. Samples of touching numerals for CENPARMI Arabic, Dari, and Urdu numeral databases are shown in, Figure 5.3, Figure 5.4, and Figure 5.5, respectively. We also applied CCA on the CENPARMI Farsi and Pashto numeral strings database, and the number of touching pair samples was very small (less than 50 samples for each language). For this reason, the purposed segmentation method was tested using samples of touching numerals from CENPARMI Arabic, Dari, and Urdu databases.

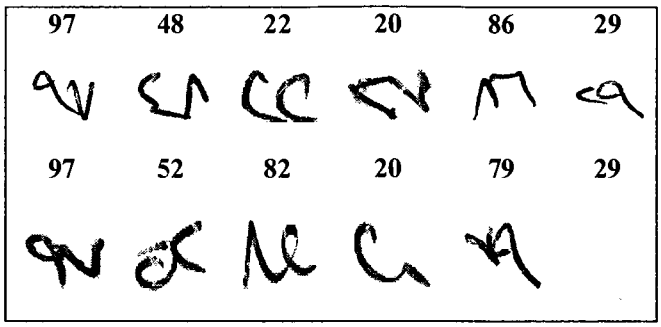


Figure 5.3: Samples of touched pairs from CENPARMI Arabic numeral database and the equivalent Latin numerals.

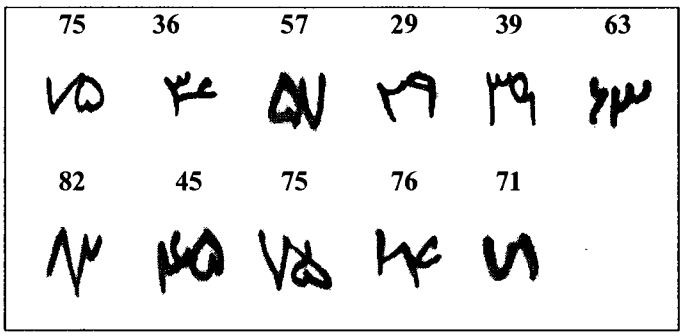


Figure 5.4: Samples of touched pairs from CENPARMI Dari numeral database and the equivalent Latin numerals.

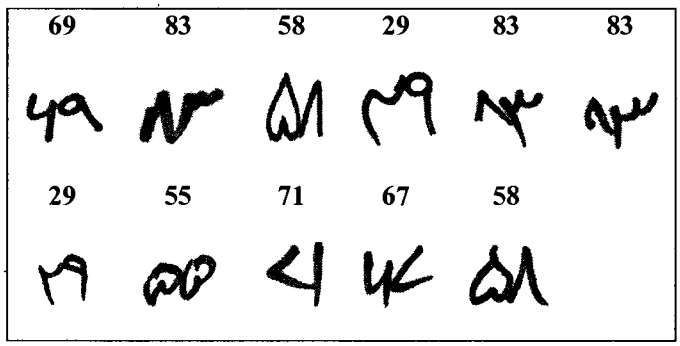


Figure 5.5: Samples of touched pairs from CENPARMI Urdu numeral database and the equivalent Latin digits.

Chapter 6

Recognition of Handwritten Dates

6.1 Our Approach

For the recognition of Arabic handwritten dates, we propose a segmentation-based recognition system. The system consists of three modules. The first module is for segmentation. In this module, an input image of an Arabic handwritten date is explicitly segmented into a sequence of basic constituents. Then, any constituent that can be a candidate for a touched-pair will be further segmented into isolated constituents using the touched-pair segmentation module presented in Chapter 5.

The second module is for recognition. This module consists of two isolated numeral classifiers. The last module is the post processing module, which performs verification that is used to improve the recognition precision by validating the outcomes from the segmentation and the recognition models at different stages. A general flowchart of the Arabic handwritten date recognition system is shown in Figure 6.1. In this chapter, we will describe these three modules in details.

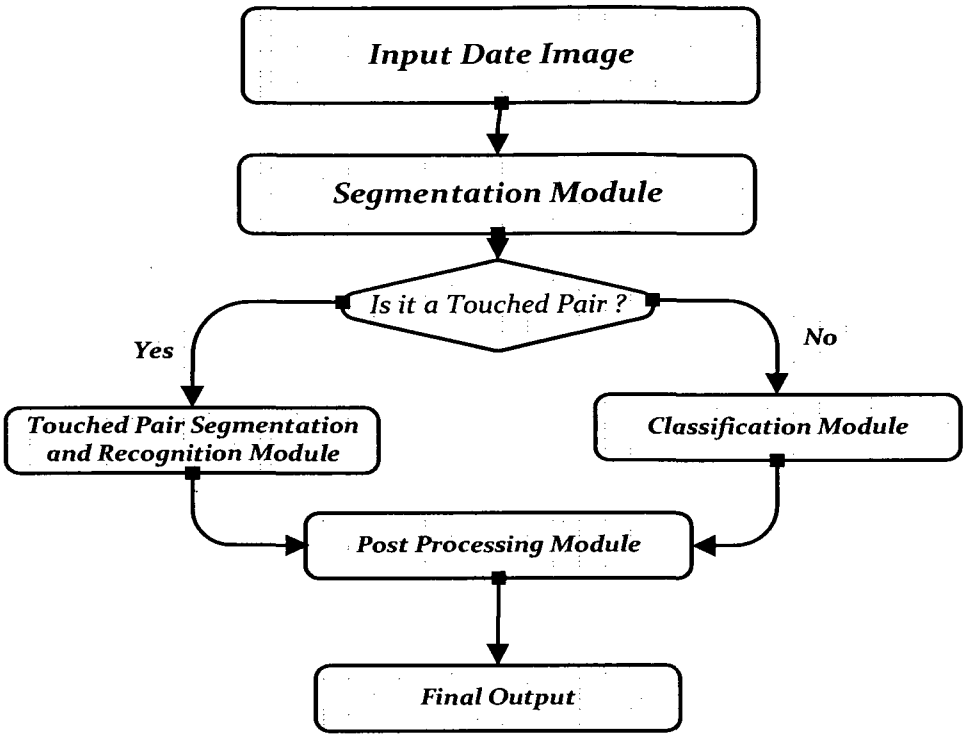


Figure 6.1: General diagram of the Arabic date recognition system.

6.2 Arabic Handwritten Dates

Arabic dates are written from right to left and can consist of the following sub-fields: day, separator1 (S1), month, separator2 (S2), and year (see Figure 6.2). Different types of separators can be used with dates, such as slash (/), back slash (\), dash (-), or comma (,). From our observation of different Arabic databases, numeral strings appear more often than words when writing dates. For the purpose of our research, we focused on the recognition of Arabic handwritten dates that consist of numerals.

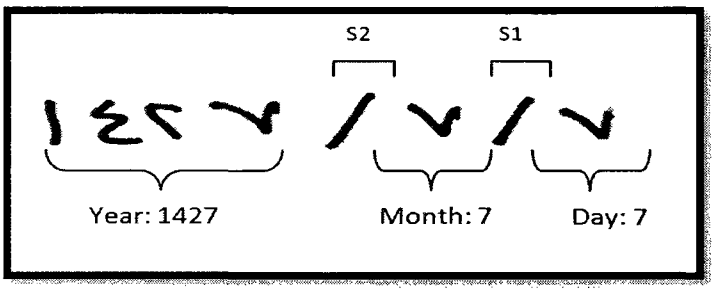


Figure 6.2: The Arabic date consists of three parts: year, month, and day.

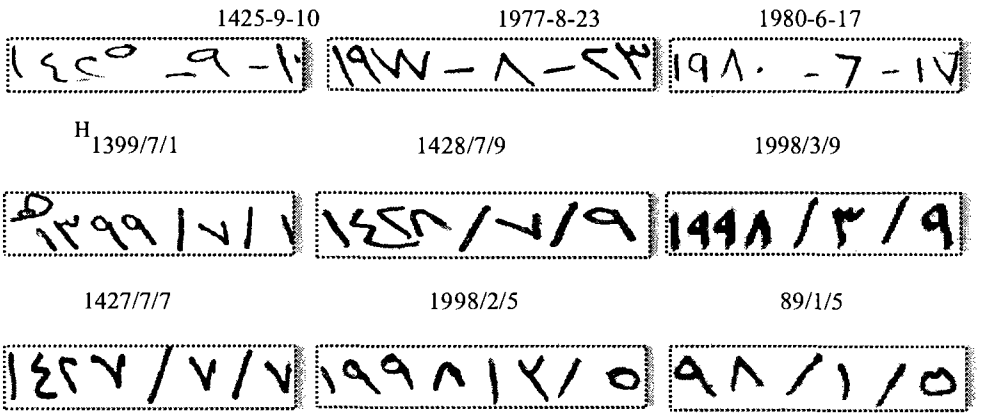


Figure 6.3: Handwritten Arabic dates from the CENPARMI Arabic database.

6.2.1 The Segmentation Module

In this recognition system, the segmentation module is very important and critical, because its performance influences the results of all the following modules. The goal of this segmentation module is to over-segment the date image into a set of primary constituents that can be individually preprocessed and recognized.

The first step in the segmentation is to apply a Connected Component Analysis (CCA) technique (detailed in Section 6.2.1.1, below) to segment all the connecting components (CCs) in the image. Afterwards, all the CCs are located and labeled

accordingly. Additional information about each CC, such as height and width, are registered to be used in the next step of the segmentation process. The final step of this module (detailed in Section 6.2.1.2, below) is to apply touching pair segmentation solely for the CCs that are candidates for touching pairs.

6.2.1.1 Connected Component Analysis Technique

Segmenting the date image into basic connected components is the first step in the segmentation module. A connected component (CC) is a group of pixels that share the same range of grayscale values. In the case of a binary image, a connected component is a group of black pixels.

Since we were working with grayscale images, we first needed to binarize the input image. For each input image, we computed a global threshold that could be used to convert a grayscale (intensity) image to a binary image. We used Otsu's method to calculate this threshold value [29]. This method chooses the threshold to minimize the intraclass variance of the black and white pixels. After the input image was binarized, it was scanned from top-to-bottom and left-to-right to find all the connected components (CCs). When a CC was found, we first calculated its Center of Gravity (CG), which could be found by averaging the corresponding x and y coordinates of all the black pixels that belonged to the CC. Then, the CC was labeled according to its appearance in the search, which was based on the horizontal position of its Center of Gravities (CG). Finally, the CC was converted back to its original grayscale values. The search continued until all the CCs were segmented. Examples of date images segmented into CCs are shown in Figure 6.4.

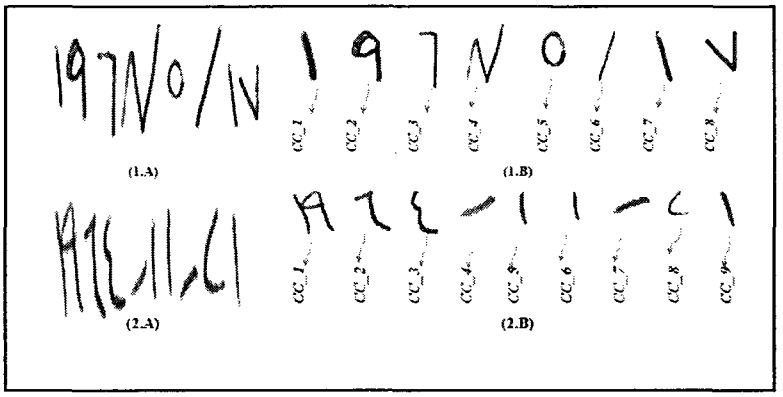


Figure 6.4: Two examples of Handwritten dates (1.A and 2.A), and the results of Basic Connected Component Analysis (1.B and 2.B). CC_4 In (1.B), and CC_4 In (2.B) are touched pairs.

6.2.1.2 Touched-pair segmentation module

Prior to this step, the input image has been segmented into CCs (connected components). A CC can be classified into one of the following types:

1. **A valid object:** This could be either an isolated numeral or separator. In this case, no further segmentation process is applied and the CC is passed to the next module (Classification Module).
2. **A touched pair:** This could comprise two connecting numerals, or a numeral connected with a separator (see Figure 6.5). In this case, the touched pair segmentation module, which was previously presented in Chapter 5, will be applied.
3. **Non-valid object**: Such as background noise. In this case, the object will be simply ignored.

The next step of the segmentation process involves the application of the touched-pair segmentation module only for the touching pair candidates. First, we

identified all CCs that could be candidates for touched-pairs. Different rules could be employed for this purpose. In our work, we employed the following rule as a measurement for the touching pair: Let W_{cc} be the CC's width. The rule is described as follows:

*If the ratio between a CC's width (W_{cc}) and the average widths of all the CCs is greater than **T**, then this CC is highly considered as a candidate for a touched pair. If the ratio is not greater than T, then the CC is more likely to be a single object.*

T is a threshold value derived empirically. When a CC is identified as a touched-pair candidate, it will be passed to the touched-pair segmentation module as presented in Chapter 5. The output of this module contains the recognition results for both of the two numerals in the touched-pair. Therefore, the touched-pair results will be passed to the post processing module rather than the classification module.

19 87 9/

(a) (b) (c)

Figure 6.5: Examples of touched pairs, segmented from date images. The touched pairs in (a) and (b) consist of two connected digits. In (c) It consists of a digit connected to a separator.

6.2.2 The Classification Module

After segmenting all the CCs from the input image, all the valid CCs were sent to the classification module. In this module, the CC is first preprocessed using the Preprocessing procedures as presented in Chapter 4. Then, using the feature extraction procedure also presented in Chapter 4, gradient features are extracted from the CC image. The final step in this module is to pass the feature vector of the image to the classifiers. In this module, two classifiers are used:

1. The first classifier **(CL_1):** This is an SVM classifier that is trained to classify an input image into one of ten classes (0,1,..,9). This classifier is the same classifier that was previously presented in Chapter 4.
2. The second classifier **(CL_2):** This is also an SVM classifier but it is trained to classify an input image into one of eleven classes instead of ten. The first ten classes are the same as (1), which are the isolated numeral classes (0,1,..,9). The last class is the separator class (or backslash /).

There are two reasons for using these two classifiers. First, based on analyses of different Arabic handwritten date databases, we found that the backslash is the most used separator in Arabic handwritten dates. Therefore, the second classifier was trained on a dataset of handwritten backslashes. This dataset was presented in Chapter 3 and it is part of the CENPARMI Arabic Handwritten Database.

The other reason for using both classifiers is based on the knowledge that we have only two separators in a date image. Therefore, training the classifier to recognize 11 classes instead of 10 will definitely affect the recognition results. Therefore, we

trained a second classifier to recognize 11 classes and used it only for the constituents that could be separators.

The next module (post processing module) will use the classification results from CL_1 and CL_2, the probability estimations provided from each classifier, and the CC's information from the segmentation module to produce the final recognition results.

6.2.3 The Post Processing Model

The post processing module is a rule-based module that incorporates some contextual knowledge to evaluate the segmentation-recognition outcomes and produce the final results. The different types of contextual knowledge incorporated in this module include: the Arabic date format, and the range of the valid values for the date parts (day, month and year).

This module evaluates the segmentation and recognition outcomes on two different stages. On ***the date stage,*** a multi-hypotheses evaluation scheme is applied to assign a valid date format to the input image. On ***the sub-part stage,*** the validity of the month, day and year values is assured. A block diagram of this module is presented in Figure 6.6. In the following sections, we will describe each of these two stages of verification.

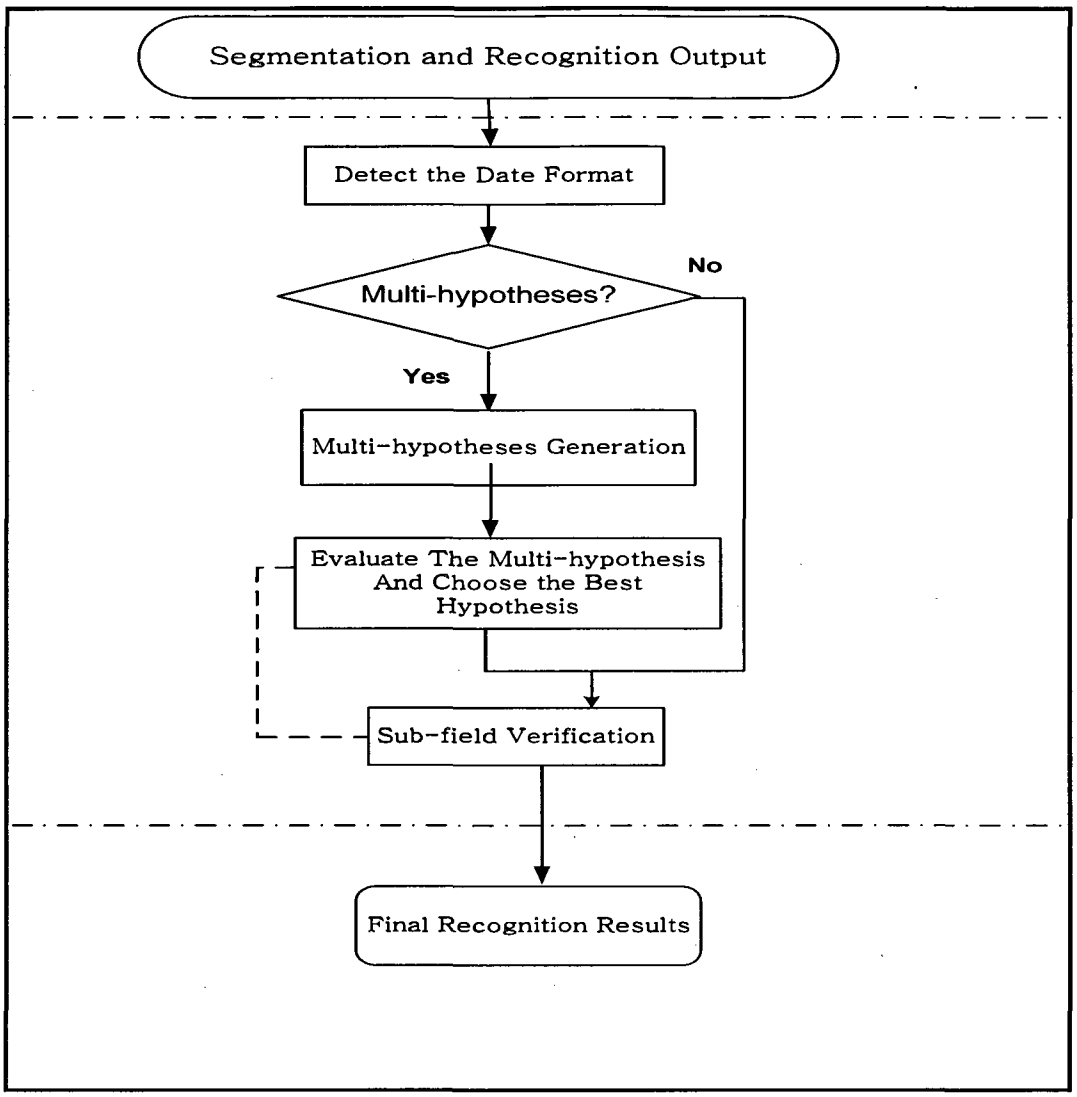


Figure 6.6: Diagram of the post processing module.

6.2.4 Verification of Date Stage

The goal of this verification stage is to assign a date format to the input image. In general, dates in Arabic are written from right-to-left and they can be described using this common pattern, with SP representing the separator as follows:

Year_Value SP Month_Value SP Day_Value

Different date formats can be introduced from this general pattern. The main difference between those formats is that each format has a different amount of numerals used to represent the day, month and year within the input image. Given the fact that in any date, the day and month can each be presented by no more than two numerals, and the year can always be presented by either two or four numerals, we can thus introduce all the different formats that could be used in writing an Arabic date (see Figure 6.7). We have categorized these formats according to the number of objects (No_Of_Obj) in the date image.

	No_Of_Obj = 10	*No_Of_Obj = 9*	*No_Of_Obj = 8*	*No_Of_Obj =7*	*No_Of_Obj = 6*
Date Format	f1:YYYYspMMspDD	f2:YYYYspMMspD	f5:YYYYspMspD	f9:YYspMMspD	f11:YYspMspD
		f3:YYYYspMspDD	f8:YYspMMspDD		

Figure 6.7: The different formats that can be used in writing Arabic dates, they are categorized according to the number of objects (No_Of_Obj).

From Figure 6.7, we can see that there are three cases where we can assign only one date format to the input image using only No_Of_Obj. These cases occur when No_Of_Obj = 10, 7 or 6. For the other cases of No_Of_Obj, there is more than one possible format can be assigned to the input image. A multi-hypotheses evaluation module will be activated in this case. Each hypothesis represents one possible format for the corresponding No_Of_Obj. These hypotheses agree on some other parts of the

dates (common parts), and conflicts on some parts (ambiguous parts). For example, in Figure 6.8, the No_Of_Obj is 9 and therefore the formats that could be assigned to this date image are:

f^2: YYYY sp MM sp D

f^3: YYYY sp M sp DD

In this example we can see that both formats agree on some parts (CC1, CC2, CC3, CC4, CC5, CC6, and CC9) and disagree on others (CC7 and CC8). The multi-hypotheses evaluation module ranks all the different hypotheses based on the confidence values for the conflict parts.

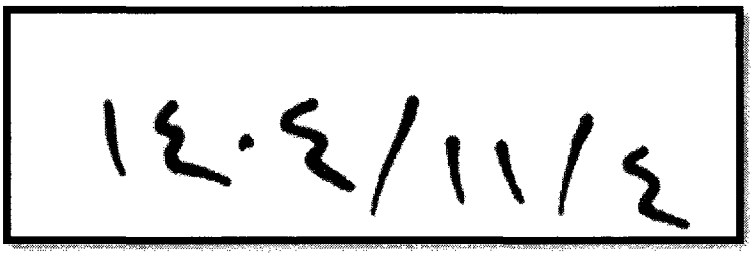


Figure 6.8: A date image with No_Of_Obj = 9.

As mentioned earlier, each classifier provides a list of probabilities that each CC belongs to any of the ten classes, where the highest probability is for the output class. We used only two probabilities. The first one is the highest probability (from Cl_1), which is the output class's probability. We will call it (digitPro). The second one is the probability for the class "separator" (from Cl_2). As mentioned earlier, the second classifier was trained to classify the input into one of 11 classes. The last class is the backslash class. We will call this probability (support). Using the latter two probabilities, and the classifier's output class value, this verification module assigns a

date format for a given input image by applying the decision tree as shown in Figure 6.9.

To illustrate this approach, we use the example given in Figure 6.9 where No_Of_Obj is 9. Therefore, the possible formats that can be assigned to this image include f^2 and f^3, which can be presented in the following hypotheses:

hyp^1: 1404 – 11 – 4

hyp^2: 1404 – 1 – 14

In this example, the ambiguous parts are CC_7 and CC_8. By using the decision tree in Figure 6.9, and the classification outputs (see Figure 6.10), the chosen format thus is hyp^1.

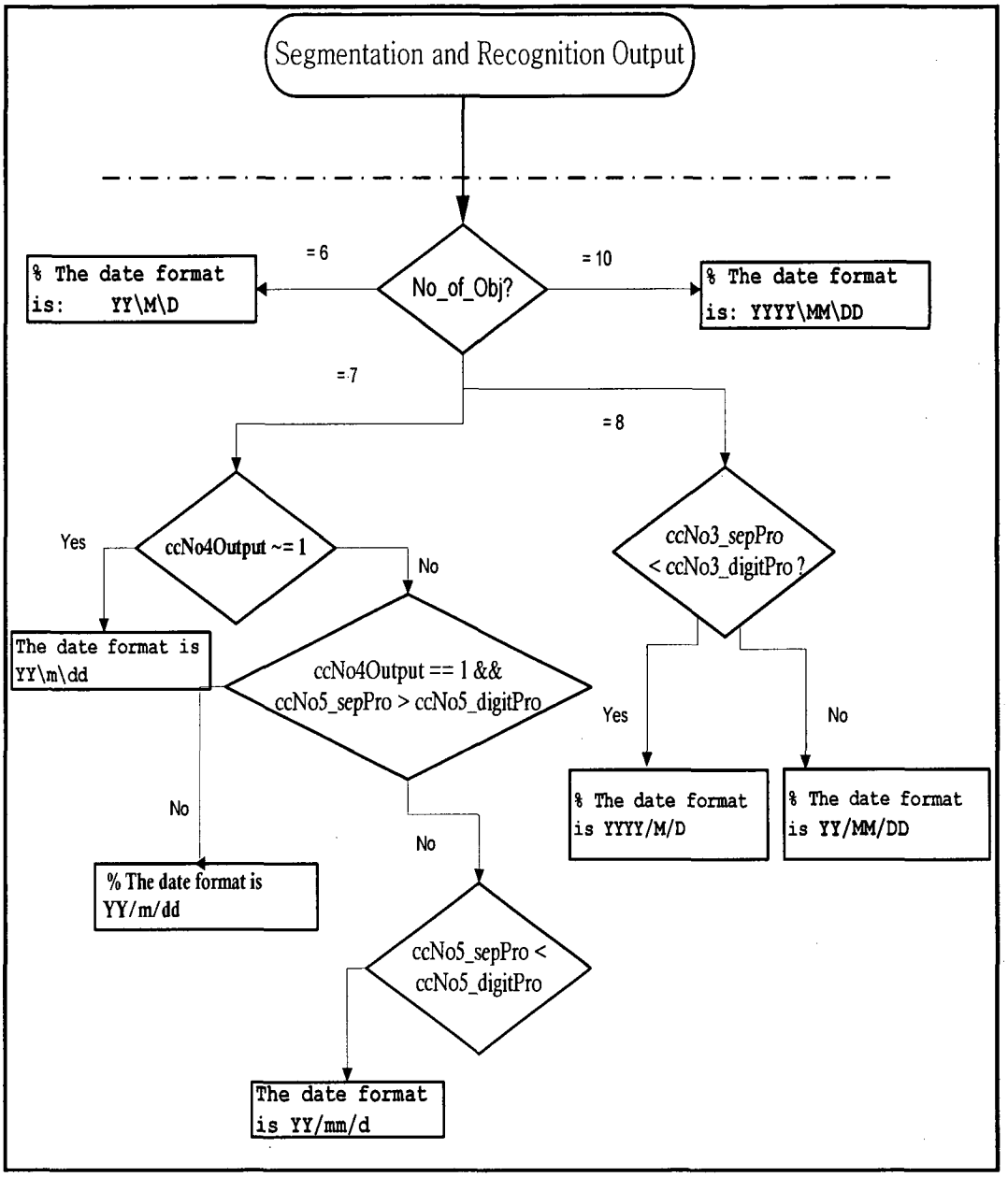


Figure 6.9: The decision tree for the date verification stage.

	digitPro	support
CC_No7	0.8459	0.0017
CC_No8	0.5586	0.4413

Figure 6.10: Digit probability and the separation probability of the ambiguous parts.

6.2.5 Verification of Sub-field Stage

The second part of the post processing module is to check the validity of each sub-field of the date. Since the lexicon sizes of the day and month parts are known and limited, we can use this lexicon information to check the validity of each sub-field. This can help to increase the recognition rate, since very often confusions between some classes of digits can be avoided. In order to do that, we used different verifications as defined by Takahashi and Griffin in [39], which include:

1. An absolute verification for a certain class (e.g., is it a "1"?).
2. One-to-one verification between two classes (e.g., is it an "8" or "9"?).
3. Clustered verification (e.g., is it a "9","8" or "5" ?).

By using these levels of verification, we have introduced a rule-base module to verify each sub-field of the date. For example, for the day part, there are two cases: 1-digit day and 2-digit day (denoted by D^1 and D^1D^2, respectively). For the D^1 day, we apply a clustered verification (is it "1", "2", "3", "4", "5", "6", "7", "8", or "9" ?). For the D^1D^2 day, we apply a clustered verification for D^2 and an absolute verification for D^1 (Is it "1" ?). For any type of verification, if the response for the verification question is positive, then the recognition result is accepted and considered the final result. But, if the response is negative, then the recognition result is changed to the second ranked output (based on the probability estimation provided by the classifier), and this output is tested. If the response is positive, it is chosen in the final result. This process is repeated until the test response becomes "Yes". The complete rule-based verification module for the date sub-field stage is illustrated in Figure 6.11. For example, if the

output of the date verification level is = 0998/3/5, then the following verification rules will apply:

1. The day part: Ver3(D1) = Ver3(5) → Positive
2. The month part: Ver3(M1) = Ver3(3) → Positive
3. The year part: Ver3(Y4)= Ver3(8) → Positive

   Ver3(Y3)= Ver3(9) → Positive

   Ver3(Y2)= Ver3(9) → Positive

   Ver1 (Y1)= Ver1(0) → Negative

   2nd ranked class = 1, Ver1(Y1) = Ver1(0) → Positive

In the next chapter, we will discuss the experimental results.

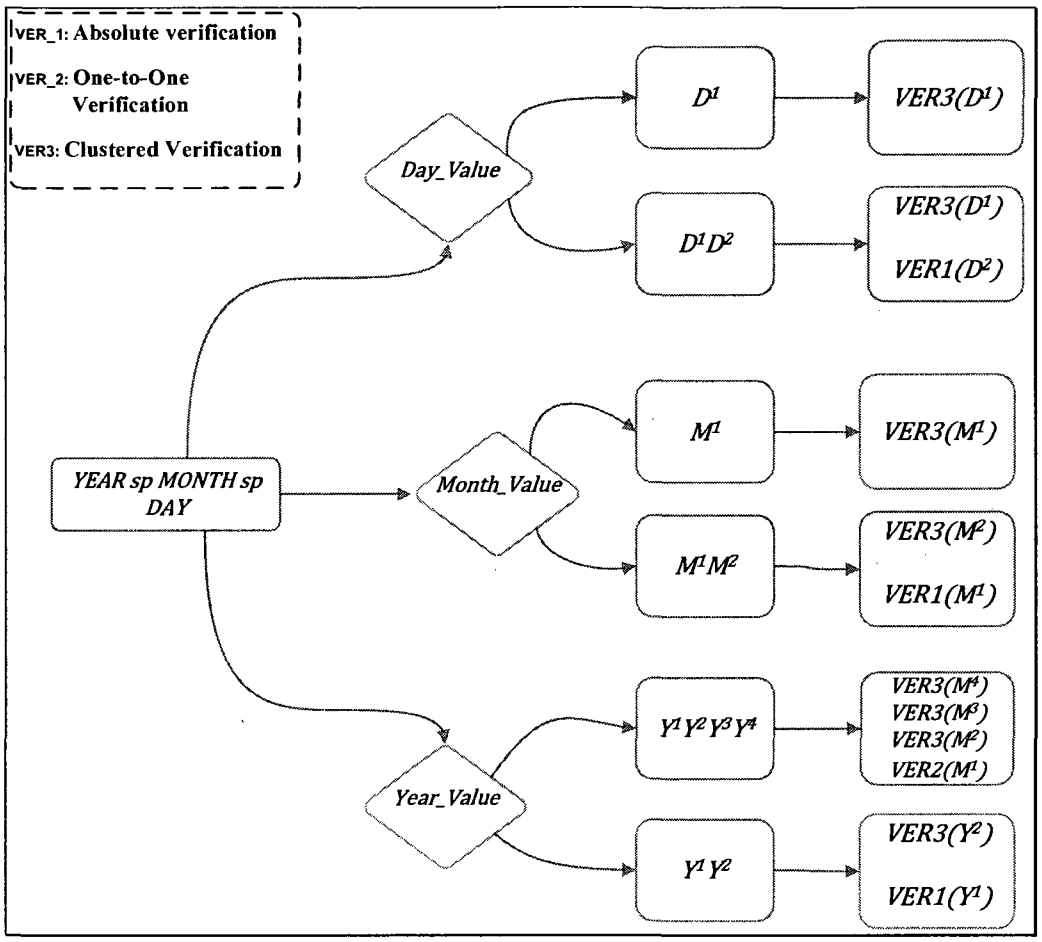


Figure 6.11: Rule-based verification module for sub-field stage.

Chapter 7

Experiments and Results

7.1 Introduction

This chapter describes the experiments that we have performed. First, in Section 7.2, experiments and analyses on the Indian off-line handwritten isolated numeral recognition system are described as well as the recognition results on different databases. Section 7.3 presents experiments and analysis of Arabic handwritten touching numeral segmentation and recognition. In addition, experiments and analyses of the same segmentation method on other two Arabic script databases are described: Urdu, and Dari. Section 7.4 includes a description of experiments and results on Arabic handwritten date recognition. These results are presented along with a description of the data set preparation.

7.2 Recognition of Off-line Isolated Indian Handwritten Numerals

For the problem of Indian off-line handwritten isolated numeral recognition, we implemented an off-line handwritten isolated numeral recognition system. Given an input image of a handwritten numeral, the system first removes the noise of the input image, extracts the bounding box around the numeral, and scales the image to a

standard size of 64-by-64 pixels. Gradient features are then extracted by convolving each pixel neighborhood with Roberts' operators to calculate the gradient strength and gradient direction, respectively. After that, the feature vector is generated as a composition of the gradient strengths accumulated in different directions, as described in Chapter 4. As a result, a feature vector of size 400 (5 vertical, 5 horizontal and 16 directional resolutions) is produced. Finally, the feature vector is passed to an SVM classifier to classify the input image to one of the ten isolated numeral classes {0,1,..,9}. Experiments were conducted on five different databases: The CENPARMI Arabic [6], Urdu, Dari [24], Farsi, and Pashto Isolated Handwritten Numeral Databases. In all of these languages (Arabic, Urdu, Dari, Farsi, and Pashto), Indian numerals are used.

Figure 7.1 shows the Indian numerals used in each language. As illustrated, some numerals are written differently in each language while others are the same in all five languages. The total number of samples in each CENPARMI database was collected as follows: 51,393 samples in the Arabic database, 60,329 in the Urdu database, 32,180 in the Dari database, 24,109 in the Farsi database, and 33.467 in the Pashto database. Each database was divided into training and testing sets. The total numbers of samples for each digit in both sets (training and testing) in Arabic, Urdu, Dari, Farsi and Pashto databases are presented in Table 7.1, Table 7.2, Table 7.3, Table 7.4, and Table 7.5, respectively.

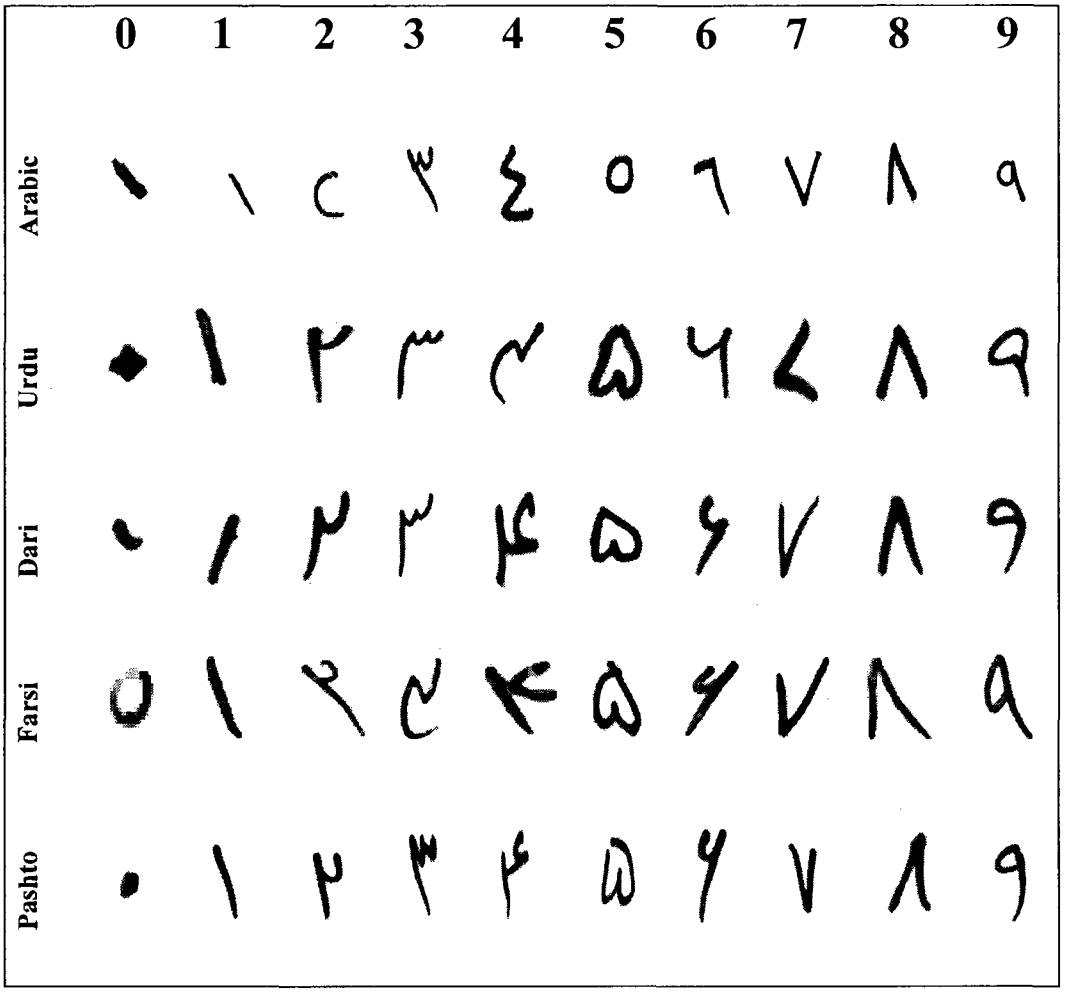


Figure 7.1: Samples of handwritten isolated numerals in Arabic, Urdu, Dari, Farsi, and Pashto and the equivalent English numerals.

Table 7.1: Total number of samples of each isolated numeral in the CENPARMI Arabic database.

Digit	0	1	2	3	4	5	6	7	8	9	Total
Training Set	4551	4049	4228	4106	3894	3923	4088	3847	3815	4611	**41112**
Testing Set	1136	1011	1058	1027	1014	941	1023	962	955	1154	**10281**

Table 7.2: Total number of samples of each isolated numeral in the CENPARMI Urdu database.

Digit	0	1	2	3	4	5	6	7	8	9	Total
Training Set	6898	5276	6370	4122	5196	3861	3980	3078	3635	4737	**47151**
Testing Set	2018	1559	1955	1065	1554	963	996	976	908	1184	**13178**

Table 7.3: Total number of samples of each isolated numeral in the CENPARMI Dari database.

Digit	0	1	2	3	4	5	6	7	8	9	Total
Training Set	3171	2621	2629	2571	2416	2326	2329	2286	2459	2934	**25741**
Testing Set	794	654	656	643	605	582	582	572	615	734	**6439**

Table 7.4: Total number of samples of each isolated numeral in the CENPARMI Farsi database.

Digit	0	1	2	3	4	5	6	7	8	9	Total
Training Set	2424	1711	1677	2020	1863	1659	2052	1813	1937	2129	**19285**
Testing Set	606	428	420	505	466	415	514	454	484	532	**4824**

Table 7.5 Total number of samples of each isolated numeral in the CENPARMI Pashto database.

Digit	0	1	2	3	4	5	6	7	8	9	Total
Training Set	2766	2726	2817	2694	2610	2457	2648	2515	2515	3022	**26770**
Testing Set	606	428	420	505	466	415	514	454	484	532	**6697**

For the isolated handwritten numeral classification, we have chosen SVM as a classifier. The kernel function chosen for this classifier is a Radial Basic Function (RBF). Before training the SVM classifier, the optimal values for the two parameters (λ) and (c) had to be chosen, where (c) is a penalty parameter of the error term and (λ) is a parameter in RBF. We used cross-validation to find the optimal values for λ

and c. Table 7.6, Table 7.7, Table 7.8, Table 7.9, and Table 7.10 show the 5-fold cross-validation rates using different parameter settings for Arabic, Urdu, Dari, Farsi and Pashto numeral databases, respectively. We then used these parameters to train the classifier using the training sets. Finally, we tested the classifier using the testing sets.

The recognition results in the Arabic, Urdu, Dari, Farsi, and Pashto testing sets were: 97.29%, 97.75%, 97.75%, 97.95% and 98.36% respectively (see Table 7.11). Confusion matrixes for classification results on the Arabic, Dari, Urdu, Dari and Farsi testing sets are shown in Figure 7.7, Figure 7.8, and Figure 7.9, Figure 7.10 and Figure 7.11, respectively. As we can see from the Arabic testing set confusion matrix (Figure 7.7), digit 7 has the highest recognition rate (99.17%) and digits 2 and 3 have the lowest recognition rates (92.82% for digit 2 and 93.79% for digit 3). Sixty-seven samples of the numeral 2 were misclassified as numeral 3, and 61 samples of the numeral 3 were misclassified as numeral 2. Our observation on this high misclassification between digits 2 and 3 is that both numerals have very similar handwritten shapes. In Figure 7.12, the first row is a set of samples of the numeral 2 and the second row is a set of samples of the numeral 3. As we can see, the first three samples from the right in both rows are almost the same. Even an Arabic native speaker cannot distinguish between them. Another common misclassification is between the Arabic digits 0 and 5 (see Figure 7.13) which is also because of the similarity between the two handwritten digits. For Urdu and Dari recognition results, we can see that some of the common misclassifications are between numerals 6 and 9 in Urdu (see Figure 7.14) and between digits 2 and 3 in Dari (Figure 7.15). The reason

for these common recognition errors is because of the similarity of shapes between these handwritten numerals.

Table 7.6: 5-fold cross-validation rates using several settings for the optimization parameters in Arabic isolated numeral database.

Log2(C)	Log2(gamma)	Recognition Rate	Error Rate	Rejection Rate
5	-7	96.87%	3.13%	0.00%
15	-11	95.98%	4.02%	0.00%
5	-15	94.43%	5.57%	0.00%
-1	-11	93.90%	6.1%	0.00%
-3	-13	90.05	9.95%	0.00%

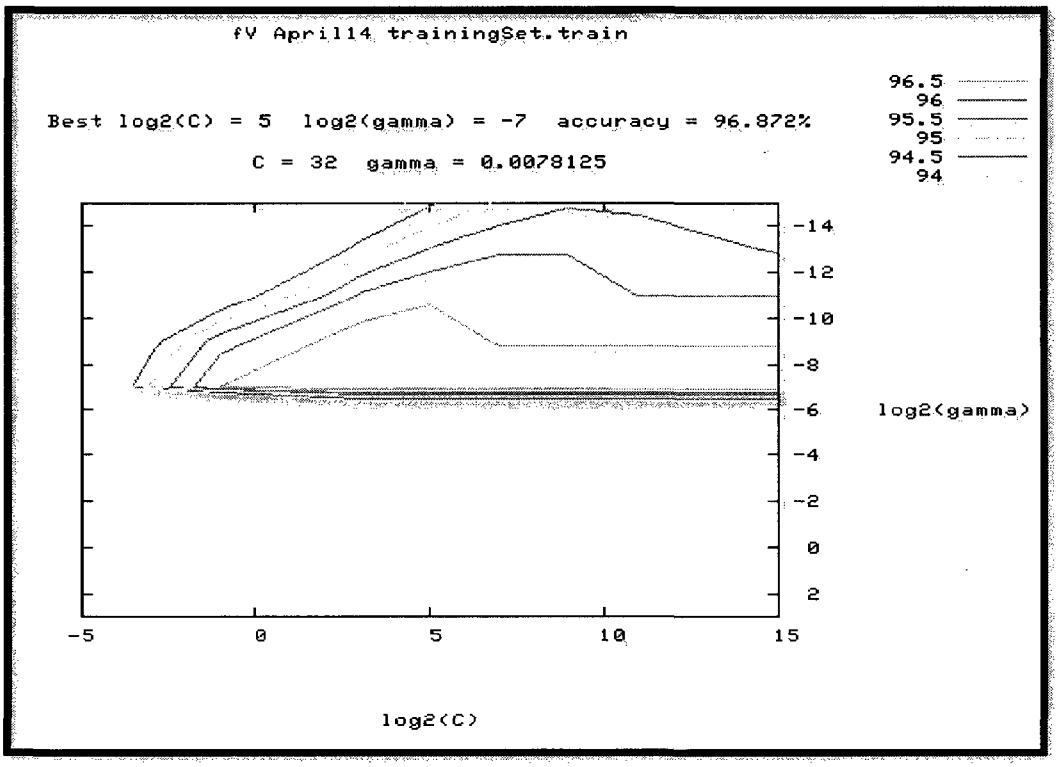


Figure 7.2: Cross-validation (Arabic).

Table 7.7: 5-fold cross-validation rates using several settings for the optimization parameters in Urdu isolated numeral database.

Log2(C)	Log2(gamma)	Rec. (%)	Err. (%)	Rej. (%)
5	-7	96.54%	3.46%	0.00%
-1	-7	95.54%	4.46%	0.00%
-3	-7	94.19%	5.81%	0.00%
5	-13	93.91%	6.09%	0.00%
-1	-13	90.63%	9.37%	0.00%

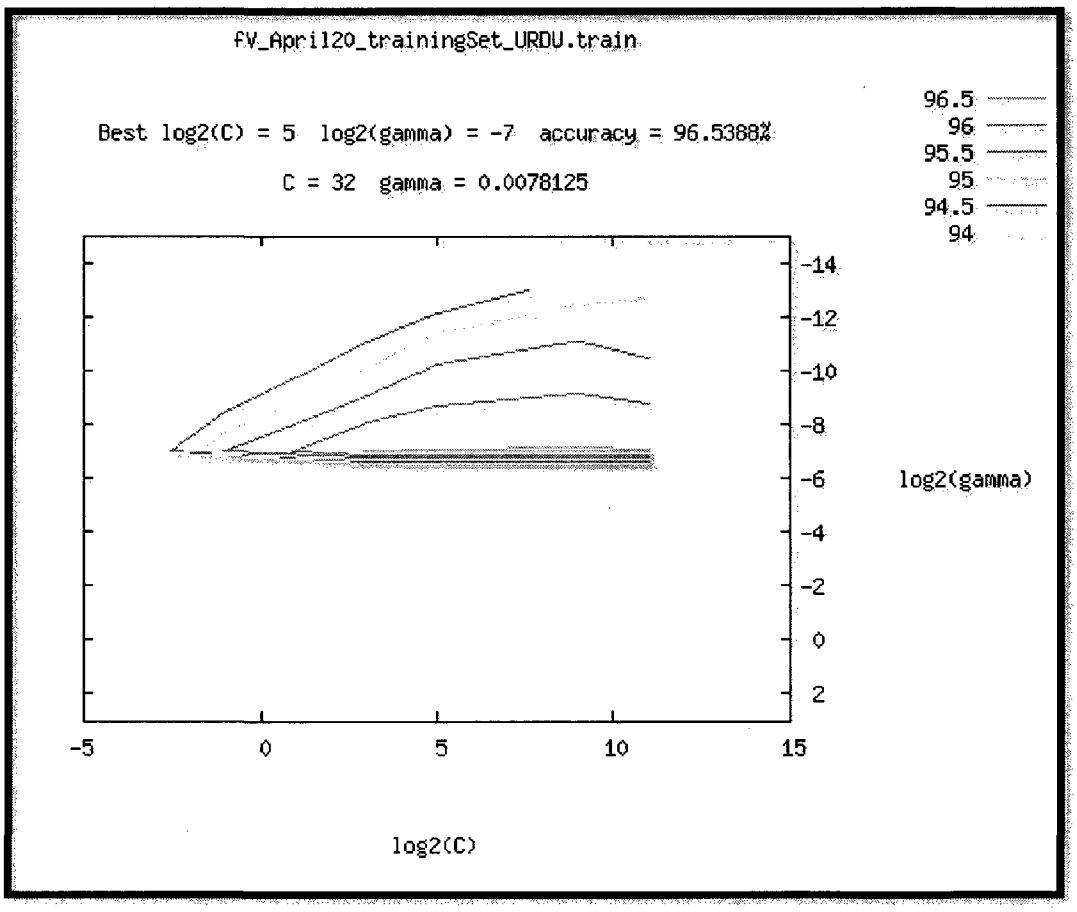


Figure 7.3: Cross-validation (URDU).

Table 7.8: 5-fold cross-validation rates using several settings for the optimization parameters in Dari isolated numeral database.

Log2(C)	Log2(gamma)	Recognition Rate	Error Rate	Rejection Rate
5	-7	97.41%	2.59%	0.00%
-1	-7	96.84%	3.16%	0.00%
5	-13	95.94%	4.06%	0.00%
-5	-7	92.2%	7.80%	0.00%
1	-7	97.49%	2.51%	0.00%

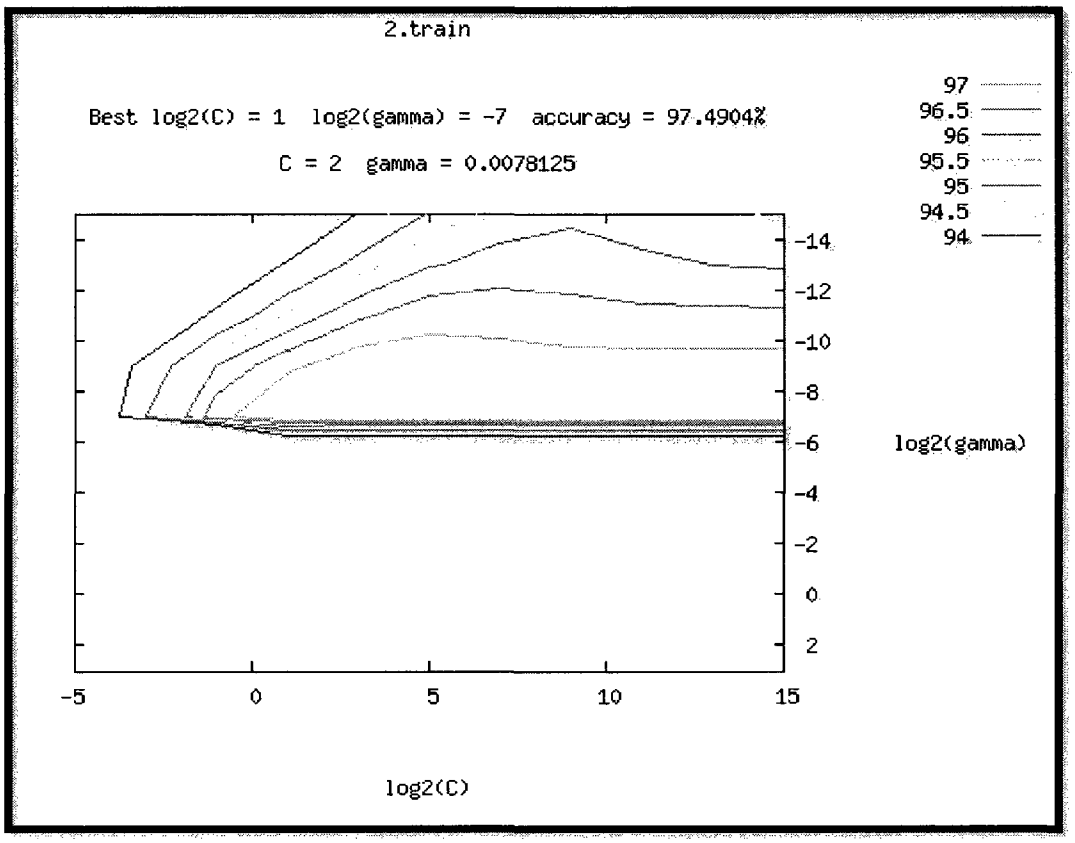


Figure 7.4: Cross-validation (Dari).

Table 7.9: 5-fold cross-validation rates using several settings for the optimization parameters in Farsi isolated numeral database.

Log2(C)	Log2(gamma)	Rec. (%)	Err. (%)	Rej. (%)
5	-7	96.54%	3.46%	0.00%
-1	-7	95.54%	4.46%	0.00%
-3	-7	94.19%	5.81%	0.00%
5	-13	93.91%	6.09%	0.00%
-1	-13	90.63%	9.37%	0.00%

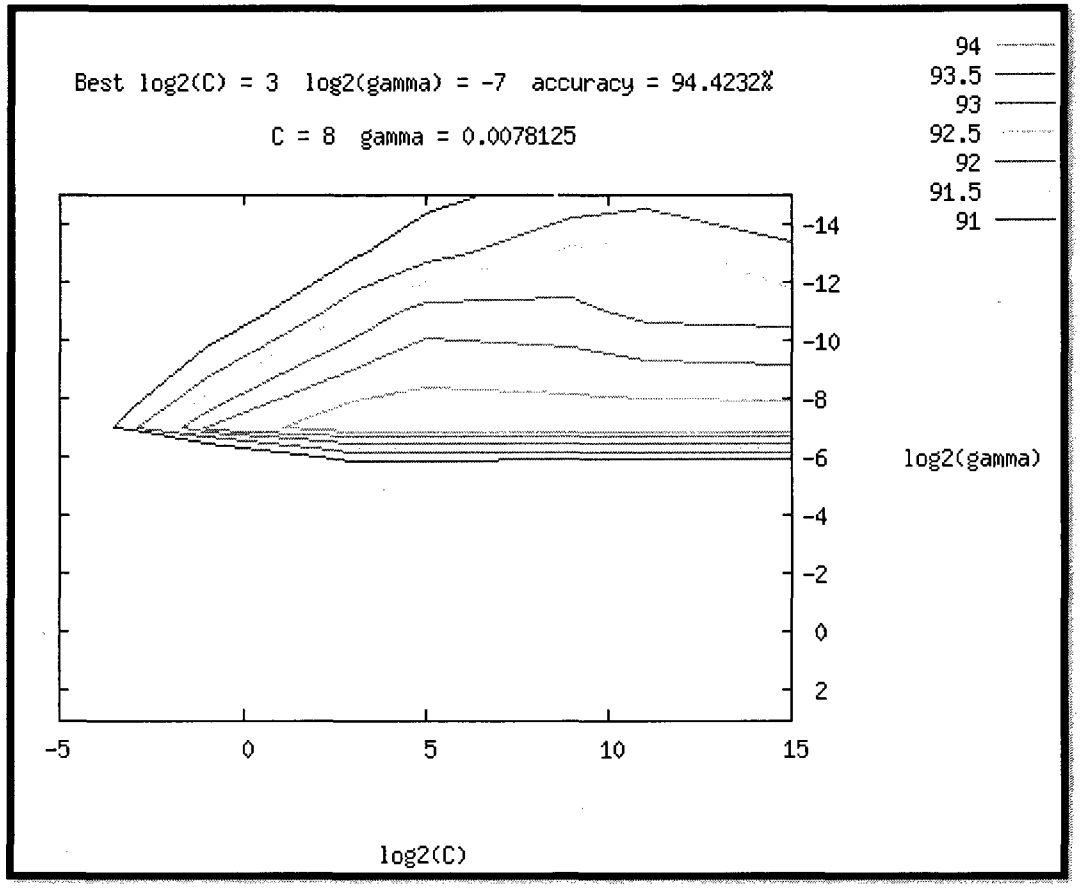


Figure 7.5: Cross-validation (Farsi).

Table 7.10: 5-fold cross-validation rates using several settings for the optimization parameters in Pashto isolated numeral database.

Log2(C)	Log2(gamma)	Rec. (%)	Err. (%)	Rej. (%)
5	-7	97.29%	2.71%	0.00%
-1	-7	96.41%	3.59%	0.00%
5	-13	95.47%	4.53%	0.00%
-3	-7	94.57%	5.43%	0.00%
-1	-13	90.84%	9.16%	0.00%

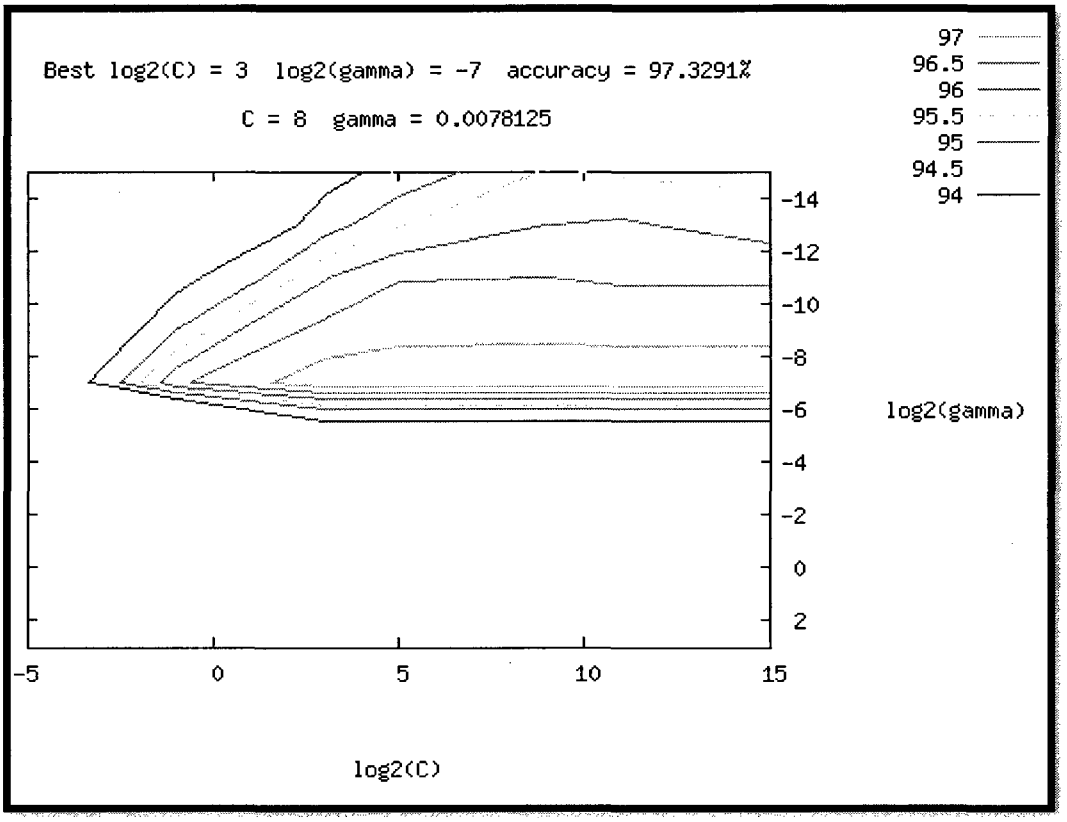


Figure 7.6: Cross-validation (Pashto).

Table 7.11: Recognition results on the CENPARMI isolated numeral databases.

	Cor.	Inc.	Total	Rec. (%)	Err. (%)	Rej. (%)
CENPARMI Arabic [6]	10281	279	10002	97.29%	2.71%	0%
CENPARMI Urdu	12882	296	13178	97.75%	2.25%	0%
CENPARMI Dari [24]	6294	145	6439	97.75%	2.25%	0%
CENPARMI Farsi	4725	99	4824	97.95%	2.05%	0%
CENPARMI Pashto	6687	10	6697	98.36%	1.64%	0%

		Arabic_Testing_Set_Results									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
True Label	**0**	**1113**	10		1		6	1	1	1	3
	1	2	**994**	1		4		1	2	2	5
	2	2	3	**982**	61	2	4		3		1
	3	1	1	67	**953**	3		1	1		
	4	1		7		**1004**			1		1
	5	11		2	1		**924**			2	1
	6	1	4	1		1		**1005**			11
	7	1	4		3				**954**		
	8		3	1			2	1		**945**	3
	9	1	5	2		2	1	13	1	1	**1128**
Accuracy	**Cor.**	1113	994	982	953	1004	924	1005	954	945	1128
	Inc.	23	17	76	74	10	17	18	8	10	26
	Total	1136	1011	1058	1027	1014	941	1023	962	955	1154
	%	**97.98**	**98.32**	**92.82**	**92.79**	**99.01**	**98.19**	**98.24**	**99.17**	**98.95**	**97.75**
	Rank	**7**	**4**	**9**	**10**	**2**	**6**	**5**	**1**	**3**	**8**

Figure 7.7: The Confusion matrix for recognition results on Arabic isolated numeral testing Set.

		Urdu_Testing_Set_Results									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
True Label	**0**	**2001**	2		1		11		2		1
	1	1	**1549**	1		1		1	2	2	2
	2			**1927**	12	16					
	3			53	**998**	13		1			
	4		1	7	1	**1544**	1				
	5	10	1				**939**		2	11	
	6	2	7	3			2	**941**	2		39
	7	2		3		1			**969**		1
	8	4	3			2	15	1	1	**881**	1
	9	1	14	2		2	2	30			**1133**
Accuracy	**Cor.**	2001	1549	1927	998	1544	939	941	969	881	1133
	Inc.	17	10	28	67	10	24	55	7	27	51
	Total	2018	1559	1955	1065	1554	963	996	976	908	1184
	%	**99.16**	**99.36**	**98.57**	**93.71**	**99.36**	**97.51**	**94.48**	**99.28**	**97.03**	**95.69**
	Rank	**4**	**1**	**5**	**10**	**2**	**6**	**9**	**3**	**7**	**8**

Figure 7.8: The confusion matrix for recognition results on Urdu isolated numeral testing set.

		Dari_Testing_Set_Results									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
True Label	**0**	**787**	1				3	1			2
	1		**650**					1			5
	2			**632**	13	6		1	2		2
	3			16	**617**	10					
	4		2	9	8	**577**		7	2		
	5	3					**575**			4	
	6			2		5		**564**	1		10
	7			2			1	1	**567**		1
	8						4	2		**607**	2
	9		6	3				7			**718**
Accuracy	**Cor.**	**787**	**650**	**632**	**617**	**577**	**575**	**564**	**567**	**607**	**718**
	Inc.	**7**	**6**	**24**	**26**	**28**	**7**	**18**	**5**	**8**	**16**
	Total	**794**	**656**	**656**	**643**	**605**	**582**	**582**	**572**	**615**	**734**
	%	**99.12**	**99.09**	**96.34**	**95.96**	**95.37**	**98.80**	**96.91**	**99.13**	**99.70**	**97.82**
	Rank	**1**	**4**	**10**	**7**	**9**	**5**	**8**	**3**	**2**	**6**

Figure 7.9: The confusion matrix for recognition results on Dari isolated numeral testing set.

		Farsi_Testing_Set_Results									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
True Label	**0**	**596**					7	3			
	1		**427**						1		
	2			**409**	7	1		3			
	3	2		13	**483**	7					
	4			1	8	**452**		4	1		
	5	4		2		2	**403**			4	
	6	1		1		2	1	**506**			3
	7					3			**451**		
	8					1	3			**479**	1
	9	1	2			2	2	6			**519**
Accuracy	**Cor.**	**596**	**427**	**409**	**483**	**452**	**403**	**506**	**451**	**479**	**519**
	Inc.	**10**	**1**	**11**	**22**	**14**	**12**	**8**	**3**	**5**	**13**
	Total	**606**	**428**	**420**	**505**	**466**	**415**	**514**	**454**	**484**	**532**
	%	**98.35**	**99.77**	**97.38**	**95.64**	**97.00**	**97.11**	**98.44**	**99.34**	**98.79**	**97.56**
	Rank	**5**	**1**	**7**	**10**	**9**	**8**	**4**	**2**	**3**	**6**

Figure 7.10: The confusion matrix for recognition results on Farsi isolated numeral testing set.

		Pashto_Testing_Set_Results									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
True Label	**0**	**685**	2				5				
	1	1	**679**								2
	2		1	**693**	3	7		1			
	3			27	**643**	5					
	4			13	9	**629**		1			
	5						**614**				
	6		2					**653**	3		5
	7			1	1			3	**624**		
	8		2				3			**624**	
	9	1	6					6			**743**
Accuracy	**Cor.**	685	679	693	643	629	614	653	624	624	743
	Inc.	7	3	12	32	23	0	10	5	5	13
	Total	692	682	705	675	652	614	663	629	629	756
	%	**98.99**	**99.56**	**98.30**	**95.26**	**96.47**	**100.0**	**98.49**	**99.21**	**99.21**	**98.28**
	Rank	**5**	**2**	**7**	**10**	**9**	**1**	**6**	**3**	**3**	**8**

Figure 7.11: The confusion matrix for recognition results on Pashto isolated numeral testing set.

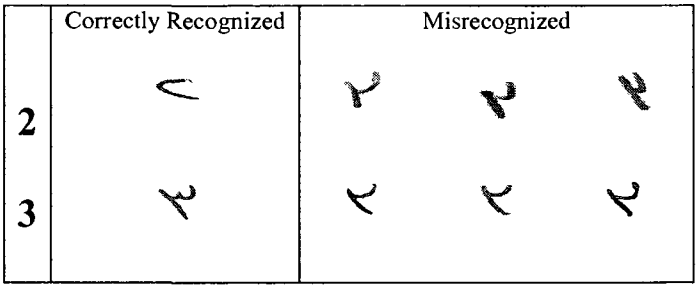


Figure 7.12: Samples of Arabic handwritten digit 2 that were misclassified as digit 3 (first row), and samples of Arabic handwritten digit 3 misclassified as digit 2 (second row).

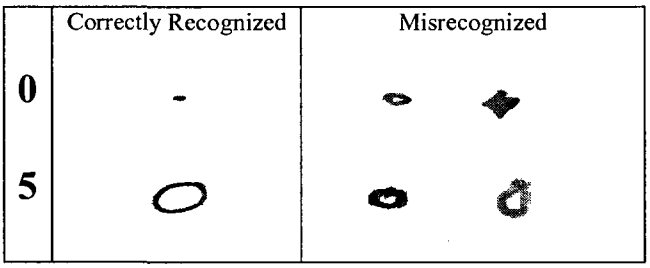


Figure 7.13: Samples of Arabic handwritten digit 0 that were misclassified as digit 5 (first row), and samples of Arabic handwritten digit 5 misclassified as digit 5 (second row).

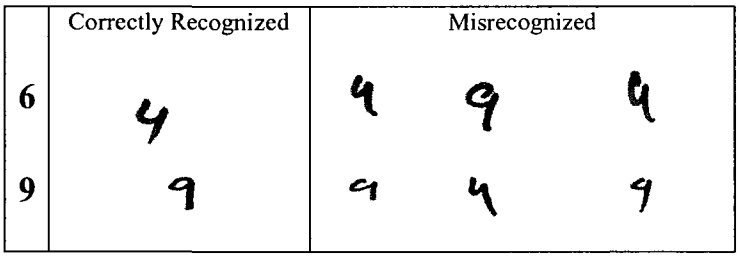


Figure 7.14: Samples of Urdu handwritten digit 6 that were misclassified as digit 4 (first row), and samples of Urdu handwritten digit 9 misclassified as digit 4 (second row).

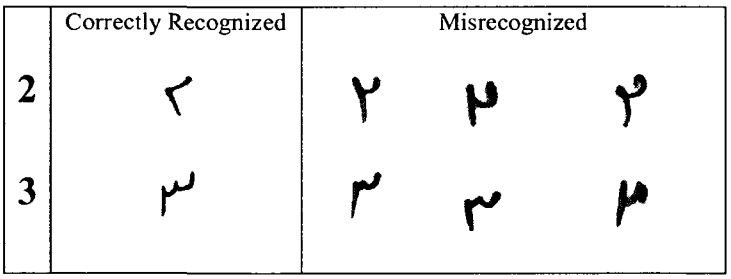


Figure 7.15: Samples of Dari handwritten digit 2 that were misclassified as digit 3 (first row), and samples of Dari handwritten digit 3 misclassified as digit 2 (second row).

7.3 Segmentation and Recognition of Arabic Off-line Handwritten Touching Numerals

7.3.1 Segmentation and Recognition of Numeral Touching Pairs

The first step in segmentation was choosing a set of values for the parameters α and β. For feasible computation, a set of five values was chosen for each parameter. The chosen sets were: [0.3, 0.4, 0.5, 0.6, and 0.7] for α, and [-6, -3, 0, 3, and 6] for β. We searched each image using all the different combinations of the chosen values (25 different combinations). As a result, 25 different models were detected and extracted to individually represent each numeral. Each model was then passed to the isolated digit classifier for classification. The highest recognition rates were: Arabic 85.4962% for model 12 ($\alpha = 0.5$, $\beta = 0.0$), Urdu 84.5745% for model 12 ($\alpha = 0.5$, $\beta = 0.0$), and Dari 79.0727% for model 12 ($\alpha = 0.5$, $\beta = -3$). Table 7.12 illustrates the recognition results of six different models for each language.

Table 7.12: The recognition results of six different models for each of the three languages.

	M_5	M_{10}	M_{12}	M_{13}	M_{16}	M_{21}
	$\alpha = 0.3$ $\beta = -6$	$\alpha = 0.4$ $\beta = -3$	$\alpha = 0.5$ $\beta = 0.0$	$\alpha = 0.5$ $\beta = -3$	$\alpha = 0.6$ $\beta = -6$	$\alpha = 0.7$ $\beta = -6$
ARABIC	60.31%	77.48%	85.50%	84.73%	83.21%	62.21%
URDU	68.09%	80.31%	84.57%	84.04%	72.87%	60.90%
DARI	56.64%	72.81%	79.07%	78.82%	75.70%	62.91%

For post processing, as described in Chapter 5, the model that provides the highest recognition rate is chosen as the default final output if its estimated probability

is higher than the threshold *Th*. Otherwise, the post processing module will be activated and a scoring scheme will be applied to choose the final recognition output. We applied three different scoring schemes. First, we applied *(M_P)* and *(H_P)* using the outputs from all 25 models. Both scoring schemes were able to improve the overall recognition rate to 88% in the Arabic and Dari datasets, and to 89% in the Urdu dataset. We then applied a scoring scheme based on both *(M_V)* and *(H_P)*, using only the outputs for the following five models: M^5, M^{10}, M^{12}, M^{16}, and M^{21}.

The choice of the best model, the choice of the different values of *(Th)*, and the choice of five models, were all based on the experimental results of the Arabic dataset. The Urdu and Dari datasets were used to test the final settings. The final recognition results are shown in Table 7.13.

We can see that the best recognition rates are as follows: 92.22%, 90.43%, and 86.10% in the Arabic, Urdu and Dari databases, respectively. They were achieved by applying the scoring scheme *(H_P+ M_V)* on the five chosen models and $Th = 0.95$.

Table 7.13: Recognition rates on three different numeral datasets of touching pairs.

Language	Th	The Best Result (M^{12})	25 Models		Combination of 5 Models
			H_P	M_V	H_P+ M_V
ARABIC	0.500	84.732%	86.6412%	87.7863%	88.1679%
	0.8755	84.732%	88.1679%	88.1679%	91.6031%
	0.9000	84.732%	88.5496%	88.5496%	91.6031%
	0.95	84.732%	88.5496%	87.4046%	92.2214%
URDU	0.500	84.5745%	85.6383%	85.1064%	86.7021%
	0.9000	84.5745%	88.0319%	86.4362%	87.2340%
	0.9044	84.5745%	88.2979%	86.4362%	89.2340%
	0.95	84.5745%	89.8936%	86.4362%	90.4255%
DARI	0.500	79.0727%	82.3308%	80.0752%	82.2055%
	0.8795	79.0727%	85.7143%	81.3283%	86.0902%
	0.9000	79.0727%	88.0319%	88.0319%	85.5890%
	0.95	79.0727%	85.4637%	80.9524%	84.7118%

7.4 Recognition of Arabic Off-line Handwritten Dates

7.4.1 Preparation of Datasets

We conducted our experiments for Arabic handwritten date recognition using two datasets. The first dataset is The Arabic Handwritten Date Dataset, which is a part of The CENPARMI Arabic Handwritten Database as presented in Chapter 3 [6]. Samples

of this dataset are shown in Figure 7.16. The other dataset was extracted from the CENPARMI Arabic Cheque Database [38]. This database includes grayscale scanned images of real-life handwritten Arabic bank cheques. Samples of this database are shown in Figure 7.17. Each sample basically consists of a set of main parts (legal amount, courtesy amount, and date, etc). In this work we are only interested on the handwritten date part of the cheque image.

The CENPARMI Arabic cheque database includes images in different sizes, which have been saved in different resolutions. Therefore, we needed to create a dataset of the handwritten date images. This was done in three steps: date-part extraction, Preprocessing, and ground truth labeling.

For the date extraction step, we first chose a small set of images from the database, and tried to estimate the ratio of the date part size to the whole image size (α_i). We calculated (α_i) for each image in this small set. We then used the average value $\alpha = \text{avg}(\alpha_i)$ to extract the date part from all the images within the dataset. Using the avarge ration, we were able to extract most of the dates from the cheque images, for some of the images were the ration is quit bigger than the avarge, we had to extract them manully.

The next step was the Preprocessing. In the date images extracted from the CENPARMI Arabic cheque database, the background of the date part consisted of light texture information or pre-printed characters (Arabic and English) (see Figure 7.18). The aim of the Preprocessing step was to remove the preprinted characters while preserving the connection of object strokes. First, we calculated a certain threshold

value (t) using Otsu's method [29]. Then, the background was removed by converting any pixel with a value less than (t) to 255. Examples of extracted and preprocessed dates from the CENPARMI Arabic Cheque Database are shown in Figure 7.19.

After that, we manually verified the output of the extraction and the Preprocessing steps. We found that some images presented different problems and therefore we had to exclude those images from the extracted set. For example, within the database (some dates were written using Arabic numerals (1,2,...,9) instead of Indian numerals (١,٢,...٩). In other images, the date parts were fully connected to other objects not related to the date, such as signatures, and handwritten or pre-printed letters. These specific images were excluded from the final set. Examples of these images are shown in Figure 7.20.

After selecting the final set from the extracted images, the last step was to generate ground truth labels, which included important information about each image: image name, date format, and content.

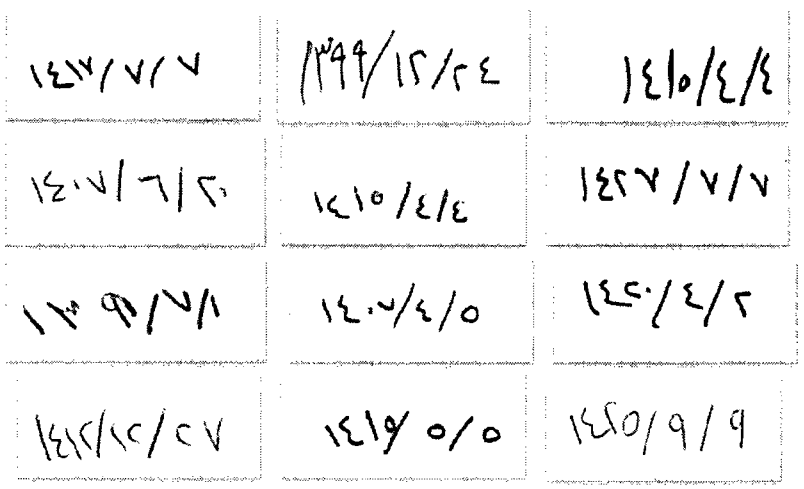


Figure 7.16: Examples from CENPARMI Arabic handwritten dates dataset [6].

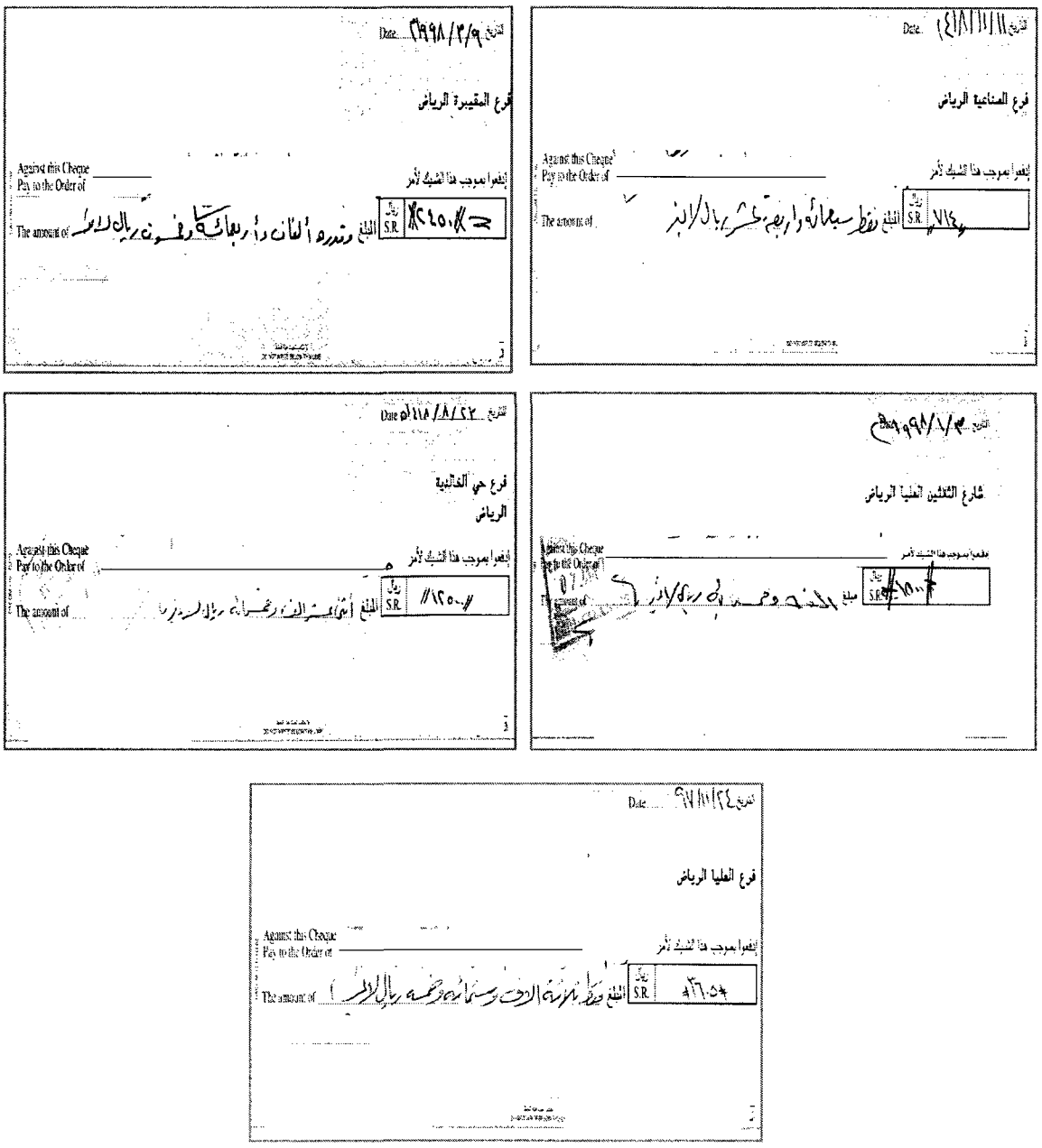


Figure 7.17: Samples of CENPARMI Arabic cheque database.

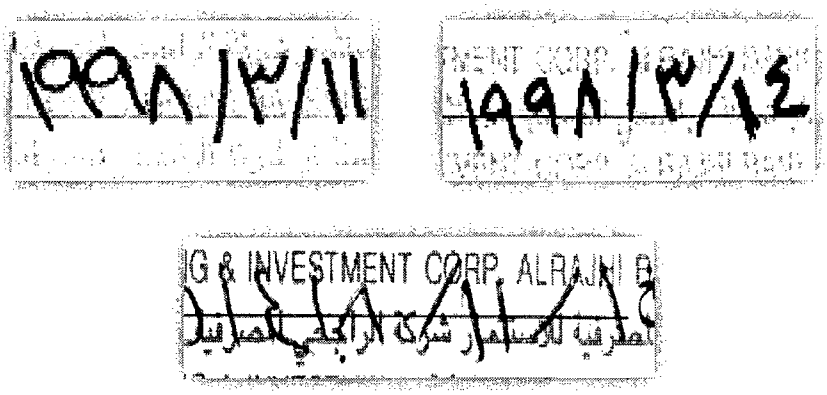


Figure 7.18: Examples of extracted dates with light texture backgrounds.

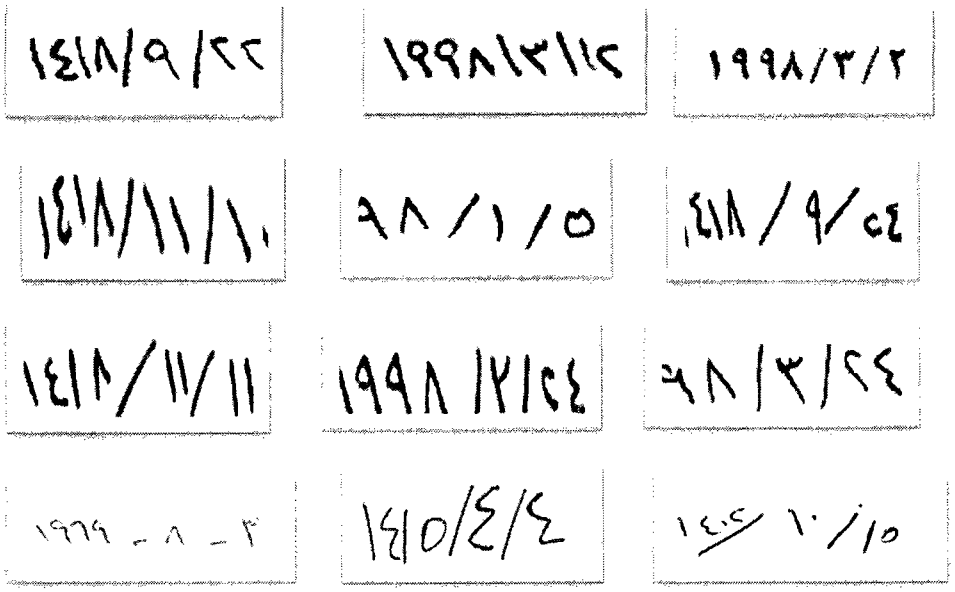


Figure 7.19: Examples of Arabic handwritten dates extracted from CENPARMI cheque database after preprocessing.

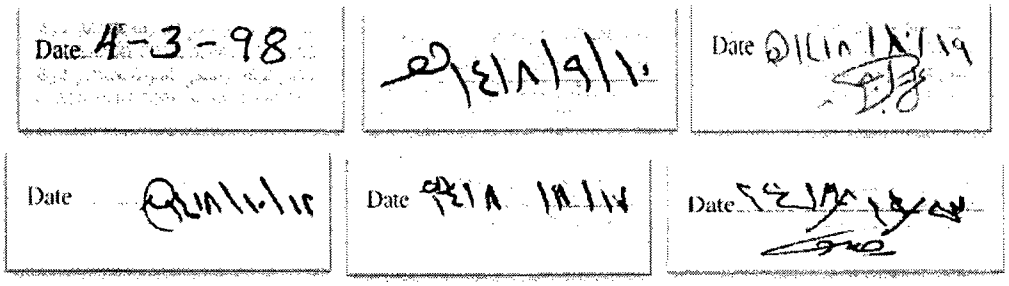


Figure 7.20: Examples of problems in some dates extracted from CENPARMI Arabic cheque database.

As a result, a total of 749 images of Arabic handwritten dates were extracted from the CENPARMI Arabic Cheque Database. By combining the samples from both databases we had a total of: 1032 samples.

7.4.2 Recognition Results

In order to discover the recognition rate at the date string level as well as at the isolated character level, we used one of the dynamic string matching algorithms, Levenshtein distance algorithm. This algorithm was developed by Russian scientist Vladimir Levenshtein in 1965 [59]. It was designed to measure the similarity between two strings. The distance measurement is based on the number of insertions, deletions, or substitutions required transforming one string into another. This algorithm is used in different applications, such as spell checking, and DNA analysis. In our experiments, we used this algorithm to compare our Arabic handwritten date recognition system output (for a certain date image) with the corresponding label.

The proposed system was tested on two different databases and the recognition results are shown in Table 7.14. As we can see from the table, the first experiment was conducted using the CENPARMI Arabic Date Dataset [6]. In this experiment, the

system was able to achieve an 85.05% recognition rate at the string level – which is the date level in this case - and a 93.80% recognition rate at the character level- which is the isolated numeral level in this case. The second experiment was conducted using the date images extracted from the CENPARMI Bank Cheque Database [7]. In this experiment, the system achieved a 66.49% recognition rate at the date level and an 88.42% recognition rate at the isolated numeral level.

After we verified the recognition outcomes, we found that there were two main types of recognition errors. First type occurred in the segmentation module, where the system failed to segment some parts of the image because there were more than two completely connected digits (see Figure 7.21). In such cases, the system failed to segment the image correctly, which affected the recognition results.

The second type of error comes from the recognition module. In these cases, the system was able to segment the date image correctly but the classifier misclassified the segmented components. The main type of classification error was the misclassification between the digit 1 and the separator (/) (see Figure 7.22). Due to the diversity in handwriting styles, the separator could look very similar to the numeral 1.

Some of the recognition errors could be corrected by the post processing module. In other cases, however, the post processing module failed to fix this problem and a wrong format would be assigned to this input image. Examples of classification errors that could be corrected in the post processing stage are shown in Figure 7.24. For example, for the first date image (first row), we can see that the recognition module produces this recognition output: (0998/3/5) and the correct label is (1998/3/5). Using

the post processing module as presented in Chapter 6, this recognition error was corrected and the final recognition output was 1998/3/5. Examples of classification errors that could not be fixed by the post processing module are shown in Figure 7.25.

Table 7.14: Arabic handwritten date recognition results.

	ARABIC Handwritten Date Recognition		
		CENPARMI ARABIC DATE DATASET [6]	CENPARMI BANK CHEQUE DATASET[7]
	Total No. of Images	283	749
	Total No. of Characters	2418	11321
Before Post Processing	Recognition Results (String-level)	85.05% (239/283)	66.49% (528/749)
	Recognition Results (Character level)	93.80% (2268/2418)	88.42% (10010/11321)
After Post Processing	Recognition Results (String-level)	85.05% (239/283)	66.49% (528/749)
	Recognition Results (Character level)	93.92% (2271/2418)	91.78% (6295/6859)

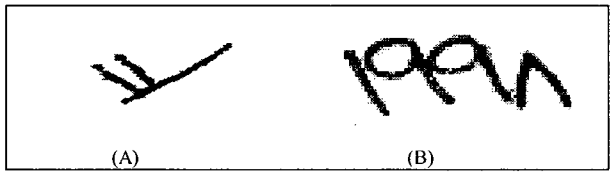


Figure 7.21: Examples of misrecognized Arabic images due to the complete connectivity between more than two digits.

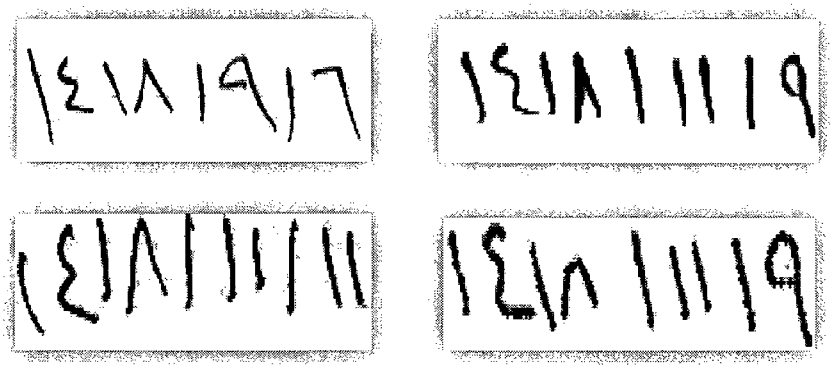


Figure 7.22: Examples of misrecognition between digit Indian 1 and separator.

Date Image	Ground Truth Label	Recognition Result (Before Post Processing)	Recognition Result (After Post Processing)
۱۹۹۸/۴/۵	**1998/3/5**	**0998/3/5**	**1998/3/5**
۱۴۱۸/۹/۱۵	**1418/9/15**	**0418/9/15**	**1418/9/15**

Figure 7.23: Examples of recognition errors improved by post processing.

Input Image	Ground Truth Label	Recognition Result (Before Post Processing)	Recognition Result (After Post Processing)
۱۴۱۸/۱۱/۹	**1418/11/9**	**9408/11/9**	**1408/11/9**
۱۹۹۸/۱/۱	**1998/1/1**	**0098/1/0**	**1089/1/1**

Figure 7.24: Examples of recognition errors could not be improved by post processing.

Chapter 8

Conclusions and Future Work

8.1 Summary

The main focus of this thesis was the recognition of Arabic handwritten dates. In this context, we have developed a recognition system for off-line isolated handwritten Indian numerals. The system has achieved an average recognition rate of 98% for three different languages: Arabic, Dari and Urdu.

Within this recognition system, a special segmentation module was also developed to segment and recognize an input image of two completely connected numerals. An overall recognition rate of 93% was achieved on touching pairs from three different databases: Arabic, Dari and Urdu.

The recognition system for the Arabic handwritten dates consists of a set of modules. The first module is the segmentation module, which is responsible for explicitly segmenting the input image into a set of isolated constituents. A special rule is then employed to measure if the segmented constituent consists of a touched pair. If it is true, then it will be passed to the touching pair segmentation module, where it will be further segmented into two isolated constituents.

For the off-line isolated handwritten Indian numeral recognition system, the segmented constituents will be preprocessed, and a set of gradient features will be extracted and passed on to two different classifiers. Each classifier will produce a recognition result and a list of probability estimations for each output class. One of the classifiers will be trained to recognize the input image as one of the ten isolated numerals. The other classifier will take into account eleven classes instead of ten, where the last class represents the backslash (/), which is frequently used in Arabic handwritten dates as a separator.

The final module in this recognition system is the post processing module. This module uses the outcomes from the segmentation and recognition modules to verify and improve the final recognition results. First, it assigns the input image a certain format based on the number of segmented objects. If more than one format can be assigned to this input image, then a multi-hypotheses generation will be activated and all the different hypotheses will be evaluated to choose the best hypothesis as the final date format. After the date format is chosen, the recognition outcomes for each part of the date (day, month, and year) will be evaluated using the sub-field verification module. Based on this verification module, if a sub-field value (based on the recognition results) is valid, it will be chosen as the final recognition result. Otherwise, the value will be changed to the closest valid value based on the recognition rank for all possible valid values.

The Arabic handwritten date recognition system was tested on two databases: the CENPARMI Arabic Handwriting Database [6] and the CENPARMI Arabic Cheque Database [7]. The experiments on the second database involved extracting the date

images from the cheque image, Preprocessing, and labeling. For both of the databases, the system respectively achieved 85.05% and 66.49% recognition rates at the date-level and 93.92% and 91.78% recognition rates at the isolated numeral-level.

8.2 Future Work

For this thesis, we developed a recognition system for Arabic handwritten dates. For this system, we developed a set of different modules using different state-of-the-art techniques in handwritten recognition. However, the performance of the recognition system can still be improved in some places in the future which we could not address in this thesis due to time constraints.

For the isolated handwritten Indian numeral recognition system, the recognition rates could be improved by adding more complementary structural features to the extracted directional features. Some structural features may complement the directional features and have low dimensionalities, such as structural and concavity features [2]. These additions could also help to improve the discrimination between numerals 2 and 3.

Another place for improvement is in the touching pair segmentation module. Although the module achieved a good recognition result for two completely connected numerals, it can only be applied to two connected numerals. Therefore, the module could be improved by considering the segmentation of more than two connecting numerals.

Finally, for the Arabic handwritten date recognition system, improvement could be made by generalizing the system to segment and recognize dates mixed with handwritten words. This could involve the extraction of more features and the consideration of different approaches for segmentation. Also, for the extraction of the date parts from the handwritten bank cheques, more Preprocessing procedures could be applied to remove the texture background and enhance the character strokes, such as pixel-contour analysis [61].

References

1. R. Plamondon, and S.N. Srihari, "On-line and Off-line Handwriting Recognition: A Comprehensive Survey," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 1. pp. 63–84, 2000.
2. C.L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten Digit Recognition: Benchmarking of State-of-The-Art Techniques," *Pattern Recognition,* Vol. 36, pp. 2271–2285, 2003.
3. Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Milller, E. Sackinger. P. Simard, and V. Vapnik, "Comparison of Learning Algorithms for Handwritten Digit Recognition," In *Proc. of the International Conference on Artificial Neural Networks*, pp. 53–60, Paris, 1995.
4. C.-L. Liu, Y.-J. Liu, and R.-W. Dai, "Preprocessing and Statistical/Structural Feature Extraction for Handwritten Numeral Recognition," *Progress of Handwriting Recognition*, pp. 161–168, Singapore, 1997.
5. F. Kimura, and M. Shridhar, "Handwritten Numeral Recognition Based on Multiple Algorithms," *Pattern Recognition*, Vol. 24, No. 10, pp. 969–983, 1991.
6. H. Alamri, J. Sadri, N. Nobile, and C. Y. Suen, "A Novel Comprehensive Database for Arabic Off-Line Handwriting Recognition," In *Proc. of the 11th International Conference on Frontiers in Handwriting Recognition*, pp. 664–669, Montreal, Canada, 2008.

7. Y. Al-Ohali, M. Cheriet, and C. Y. Suen, "Databases for Recognition of Handwritten Arabic Cheques," *Pattern Recognition*, Vol. 36, No. 1, pp. 111–121, 2003.

8. M. Shi, Y. Fujisawa, T. Wakabayashi, and F. Kimura, "Handwritten Numeral Recognition Using Gradient and Curvature of Grayscale Image," *Pattern Recognition*, Vol. 35, pp. 2051–2059, 2002.

9. J.T. Favata, G. Srikantan, and S.N. Srihari, "Handprinted Character/Digit Recognition Using a Multiple Feature/Resolution Philosophy," In *Proc. of the 4th International Workshop on Frontiers of Handwriting Recognition*, pp. 57–66, Taipei, Taiwan, 1994.

10. C.-L. Liu and C. Y. Suen, "A New Benchmark on The Recognition of Handwritten Bangla and Farsi Recognition," In *Proc. of the 11th International Conference on Frontiers in Handwriting Recognition*, pp. 278–283, Montreal, Canada, 2008.

11. L.G. Roberts, "Machine Perception of Three-Dimensional Solids", *Optical Electro-Optical Processing of Information*, MIT Press, Cambridge, Massachusetts, pp. 159–197, 1965.

12. J.C. Russ, *The Image Processing Handbook*, 2nd Edition, CRC Press, Boca Raton, USA, 1995.

13. T. Wakabayashi, S. Tsuruoka, F. Kimura, and Y. Miyake, "Increasing the Feature Size in Handwritten Numeral Recognition to Improve Accuracy," *Systems and Computers in Japan*, Vol. 26, No. 8, pp. 35–44, 1995.

14. K. Fukunaga, *Introduction to statistical pattern recognition*, 2nd Edition, Academic Press, New York, 1990.

15. I.-S. Oh, and C. Y. Suen, "A Class-Modular Feed Forward Neural Network for Handwriting Recognition," *Pattern Recognition*, Vol. 35, No. 1, pp. 229–244, 2002.

16. C.J.C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Knowledge Discovery Data Mining*, Vol. 2, No.2, pp. 1–43, 1998.

17. S. Huang, M. Ahmadi, and M. Sid-Ahmed, "A Hidden Markov Model-Based Character Extraction Method," *Pattern Recognition,* Vol. 41, No. 9, pp. 2890–2900, 2008.

18. N. Cristianini, and J. Shawe-Taylor, *Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, 2000.

19. C.-C. Chang, and C.-J. Lin, "LIBSVM: a Library for Support Vector Machine," 2001. Software available at: http://www.csie.ntu.edu.tw/~cjlin/libsvm

20. D. S. Britto JR., R. Sabourin, E. Lethelier, F. Bortolozzi, and C. Y. Suen, "Improvement in Handwritten Numeral String Recognition by Slant Correction and Contextual Information," In *Proc. of International Workshop on Frontiers in Handwriting Recognition*, pp. 323–332, Amsterdam, 2000.

21. L. Liu, H. Sako, and H. Fujisawa, "Effects of Classifier Structure and Training Regimes on Integrated Segmentation and Recognition of Handwritten Numerals Strings," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 11, pp. 1395–1407, 2004.

22. L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "Automatic segmentation of handwritten numerical strings: A Recognition and Verification Strategy," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 11, pp. 1438–1454, 2002.

23. J. Sadri, C. Y. Suen, and T. D. Bui, "Automatic Segmentation of Unconstrained Handwritten Numeral Strings," In *Proc. of International Workshop on Frontiers in Handwriting Recognition*, pp. 317–322, Tokyo, 2004.

24. M. I. Shah, J. Sadri, N. Nobile, and C. Y. Suen, "A New Multipurpose Comprehensive Database for Handwritten Dari recognition," In *Proc. of the 11th International Conference on Frontiers in Handwriting Recognition*, pp. 635–640, Montreal, Canada, 2008.

25. Y. Wang, X. Liu, and Y. Jia, "A Holistic Approach to Handwritten Numeral Pair Recognition Based on Generative Models of Numeral Pairs", In *Proc. of 11th International Conference on Frontiers in Handwriting Recognition*, pp. 54–58, Montreal, Canada, 2008.

26. L. Lam, C. Y. Suen, D. Guillevic, N. W. Strathy, M. Cheriet, K. Liu, and J. N. Said. "Automatic Processing of Information on Cheques," In *International Conference on Systems, Man and Cybernetics*, Vancouver, 1995, pp. 2353–2358.

27. Q. Xu, L. Lam, and C. Y. Suen, "Automatic Segmentation and Recognition System for Handwritten Dates on Canadian Bank Cheques," In *Proc. of 7th*

International Conference on Document Analysis and Recognition, pp. 704–708, Edinburgh, 2003.

28. M. E. Morita, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "Segmentation and Recognition of Handwritten Dates: An HMM-MLP Hybrid Approach," *International Journal on Document Analysis and Recognition,* Vol. 6, No. 4, pp. 248–262, 2003.
29. N. Otsu, "A Threshold Selection Method from Gray-Level Histograms, IEEE Transactions on Systems," *IEEE Trans. on Man, and Cybernetics*, Vol. 9, No. 1, pp. 62–6, 1979.
30. A. Sagheer, N. Tsuruta, and R.-I. Taniguchi, "Arabic Lip-reading System: A Combination of Hypercolumn Neural Network Model with Hidden Markov Model," In *Proc. of International Conference on Artificial Intelligence and Soft Computing*, pp. 311–316, Marbella, Spain, 2004.
31. F. Barbara, *Ethnologue, Languages of the World*, 14th Ed, SIL International, Dallas, USA, 2000.
32. L.M. Lorigo, and V. Govindaraju, "Offline Arabic Handwriting Recognition: A Survey," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 28, pp. 712–724, 2006.
33. U. Marti, and H. Bunke, "A Full English Sentence Database for Off-line Handwriting Recognition," In *Proc. of the 5th International Conference on Document Analysis and Recognition*, pp. 705–708, Bangalore, India, 1999.

34. J.J. Hull, "A Database for Handwritten Text Recognition Research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, pp. 550–554, 1994.

35. G. Dimauro, S. Impedovo, R. Modugno, and G. Pirlo, "A New Database for Research on Bank-Check Processing," In *Proc. of the 8th International Workshop on Frontiers in Handwriting Recognition*, pp. 524–528, Niagara-on-the-Lake, Canada, 2002.

36. M. Pechwitz, S. S. Maddouri, V. Maergner, N. Ellouze, and H. Amiri, "IFN/ENIT- Database of Handwritten Arabic Words", In *Proc. of the Colloque International Francophone sur l'Écrit et le Document*, pp. 129–136, Hammamet, Tunisia, 2002.

37. S. Alma'adeed, D. Elliman, and C.A. Higgins, "A Database for Arabic Handwritten Text Recognition Research," *In Proc. of the 8th International Workshop of Frontiers in Handwriting Recognition*, pp. 485–589, Niagara-on-the-Lake, Canada, 2002.

38. Y. Al-Ohali, M. Cheriet, and C. Suen, "Database for Recognition of Handwritten Arabic Cheques," In *Proc. of the 7th International Workshop on Frontiers in Handwriting Recognition*, PP. 601-606, Amsterdam, 2000.

39. H. Takahashi and T.D. Griffin, "Recognition Enhancement by Linear Tournament Verification," In *Proc. of the 2nd International Conference on Document Analysis and Recognition*, pp. 585–588, Tsukuba, Japan, 1993.

40. L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "A Modular System to Recognize Numerical Amounts on Brazilian Bank Cheques," In *Proc. of the 6th International Conference on Document Analysis and Recognition*, pp. 389–394, Seattle, USA, 2001.

41. L. M. Kleper, *The Handbook of Digital Publishing*, Prentice Hall PTR, Upper Saddel River, USA.

42. E. Borovikov, I. Zavorin, and M. Turner, "A Filter Based Post-OCR Accuracy Boost System," In *Proc. of the 1st ACM workshop on Hardcopy document processing*, pp. 23–28, Washington, DC, USA, 2004.

43. D. Das, and R. Yasmin, "Segmentation and Recognition of Unconstrained Bangla Handwritten Numeral," *Asian Journal of Information Technology,* Vol. 5, No. 2, pp. 155–159, 2006.

44. H. Al-Rashaideh, "Preprocessing Phase for Arabic Word Handwritten Recognition," *Information Transmissions in Computer Networks*, Vol. 6, pp. 11–19, 2006.

45. C. He, P. Zhang, J. Dong, C. Y. Suen, and T. D. Bui, "The Role of Size Normalization on the Recognition Rate of Handwritten Numerals," In *Proc. of Workshop of 8th International Conference on Document Analysis and Recognition Neural Networks and Learning in Document Analysis and Recognition*, pp. 8 – 12, Seoul, South Korea, 2005.

46. A. Harifi and A. Aghagolzadeh, "A New Pattern for Handwritten Persian/Arabic Digit Recognition", In *Proc. of World Academy of Science, Engineering and Technology*, Vol. 3, pp. 293-296, 2005.

47. O. D. Trier, A. K. Jain, and T. Taxt, "Feature Extraction Methods for Character Recognition—A Survey," *Pattern Recognition*, Vol. 29, No. 4, pp. 641–662, 1996.

48. P. Phuong, N. Tao, and L. Mai, "Speeding Up Isolated Vietnamese Handwritten Recognition by Combining SVM and Statistical Features," *IJCSES International Journal of Computer Sciences and Engineering Systems*, Vol. 2, No. 4, pp. 257-261, 2008.

49. N. Sharma, U. Pal, and F. Kimura, "Recognition of Handwritten Kannada Numerals," In *Proc. of the 9th International Conference on Information Technology*, pp. 133–136, Bhubaneswar, India, 2006.

50. C. V. Lakshmi, R. Jain, and C. Patvardhan , "Handwritten Devnagari Numerals Recognition with Higher Accuracy," *In Proc. of the International Conference on Computational Intelligence and Multimedia Applications*, Vol. 3, pp. 255–259, Sivakasi, India, 2007.

51. W. Zhang, Y. Y. Tang, and Y. Xue, "Handwritten Character Recognition Using Combined Gradient and Wavelet Feature," In *Proc. of International Conference on Computational Intelligence and Security*, Vol. 1, pp. 662–667, Guangzhou, China, 2006.

52. S. Behnke, M. Pfister, and R. Rojas, "A Study on the Combination of Classifiers for Handwritten Digit Recognition," In *Proc. of Neural Networks in Applications, 3rd International Workshop (NN' 98)*, pp. 39–46, Magdeburg, Germany, 1998.

53. D. Gorgevik, and D. Cakmakov, "An Efficient Three-Stage Classifier for Handwritten Digit Recognition," In *Proc. of 17th International Conference on Pattern Recognition*, Vol. 4, pp. 507–510, Cambridge, England, 2004.

54. J. Cao, M. Ahmadi, and M. Shridhar, "A Hierarchical Neural Network Architecture for Handwritten Numeral Recognition," *Pattern Recognition*, Vol. 30, No. 2, pp. 289–294, 1997.

55. G.Y. Chen, T.D. Bui, and A. Krzyzak, "Contour-based Handwritten Numeral Recognition Using Multiwavelets and Neural Networks," *Pattern Recognition*, Vol. 36, pp. 1597–1604, 2003.

56. S. M. Awaidah, and S. A. Mahmoud, "A Multiple Feature/Resolution Scheme to Arabic (Indian) Numerals Recognition Using Hidden Markov models," *Signal Processing,* Vol. 89, pp. 1176–1184, 2009.

57. A. Jr, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "The recognition of Handwritten Numeral Strings Using a Two-stage HMM-based Method," *International Journal on Document Analysis and Recognition*, Vol. 5, pp. 102–117, 2003.

58. J. Cai and Z.-Q. Liu, "Integration of Structural and Statistical Information for Unconstrained Handwritten Numeral Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 3, pp. 263-270, 1999.

59. V. I. Levenshtein. "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals," *Soviet Physics Doklady,* Vol. 6, pp. 707–710, 1966.

60. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-Based Learning Applied to Document Recognition," *Proc. of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

61. Ch. Zhang, J. Yang, and Z. Lou, "Preprocessing of Handwritten Date Images on Chinese Cheque," In *Proc. of the 18th International Conference on Pattern Recognition*, Vol. 2, pp. 1010-1013, Hong Kong, 2006.

Appendix

Databases		Training	Testing	Validation	Total
Dates_Free_Fromat		170	57	57	284
Isolated _Characters	ALIF	194	65	66	325
	Alif_2	194	65	66	325
	Alif_3	194	65	66	325
	AYN	194	65	66	325
	BAA	194	65	66	325
	DAAD	194	65	66	325
	DALL	194	65	66	325
	FAA	194	65	66	325
	GAAF	194	65	66	325
	GHAYN	194	65	66	325
	HA	194	65	66	325
	Ha_2	194	65	66	325
	Ha_3	194	65	66	325
	HAA	194	65	66	325
	HAMZA	194	65	66	325
	JEEM	194	65	66	325
	KAAF	194	65	66	325

	KHA	194	65	66	325
	LAAM	194	65	66	325
	MEEM	194	65	66	325
	NUUN	194	65	66	325
	RAA	194	65	66	325
	SAAD	194	65	66	325
	SEEN	194	65	66	325
	SHEEN	194	65	66	325
	TA	194	65	66	325
	TAA	194	65	66	325
	THA	194	65	66	325
	THAA	194	65	66	325
	THAAL	194	65	66	325
	WAAW	194	65	66	325
	Waaw_2	194	65	66	325
	YAA	194	65	66	325
	Yaa_2	194	65	66	325
	ZAAY	194	65	66	325
Isolated Digits	**Isolated_0**	2800	940	940	4680
	Isolated _1	2800	940	940	4680
	Isolated _2	2800	940	940	4680
	Isolated _3	2800	940	940	4680
	Isolated _4	2800	940	940	4680
	Isolated _5	2800	940	940	4680

			Isolated _6	2800	940	940	4680
			Isolated _7	2800	940	940	4680
			Isolated _8	2800	940	940	4680
			Isolated _9	2800	940	940	4680
			Length_2	2522	840	840	4202
			Length_3	1499	500	500	2499
Numeral Strings		**Integer_Strings**	**Length_4**	1299	434	435	2168
			Length_6	966	322	323	1611
			Length_7	969	324	324	1617
		Real_Strings	**Length_3**	195	66	66	327
			Length_4	194	84	84	362
			Float_Dot	389	132	132	653
	Special_Symbols		**At_Symbol**	196	66	66	328
			Comma_Symbol	196	66	66	328
			Number_Symbol	196	66	66	328
			Colon_Symbol	196	66	66	328
			Dash_Symbol	196	66	66	328
	Words		**Debit**	191	66	64	321
			Amount	191	66	66	323
			Article	191	66	66	323
			Balance	191	65	66	322
			Barrel	191	66	66	323
			Carton	191	66	66	323
			Cash	191	66	66	323

	Centimeter	191	65	65	321
	Cost	191	66	66	323
	Credit	191	66	66	323
	Decrease	191	66	66	323
	Delivery	191	66	66	323
	Dozen	191	66	66	323
	Due	191	66	66	323
	Duty	191	66	66	323
	Eight	191	64	64	319
	Five	191	65	65	321
	Expire	191	66	66	323
	Four	191	66	66	323
	Gallon	191	66	66	323
	Gram	191	66	66	323
	Hundred	191	66	66	323
	Inch	191	66	66	323
	Increase	191	66	66	323
	Interest	191	66	66	323
	Inventory	191	66	66	323
	Item	191	66	66	323
	Kilo	191	66	66	323
	Transfer	191	66	66	323
	Length	191	66	66	323
	Liter	191	66	66	323

	Meter	191	66	66	323
	Milligram	191	65	65	321
	Milliliter	191	66	66	323
	Nine	191	65	65	321
	Plus	191	66	66	323
	Hallah	191	66	66	323
	Number	191	66	66	323
	One	191	66	66	323
	Period	191	66	66	323
	Price	191	66	66	323
	Rent	191	66	66	323
	Sale	191	66	66	323
	Seven	191	66	66	323
	Payment	191	66	66	323
	Six	191	66	66	323
	Part	191	66	66	323
	Stock	191	66	66	323
	Tax	191	66	66	323
	Ten	191	66	66	323
	Riyal	191	66	66	323
	thousand	191	66	66	323
	Three	191	66	66	323
	Ton	191	66	66	323
	Total	191	66	66	323

Product	191	66	66	323
Two	191	66	66	323
Volume	191	66	66	323
Weight	191	66	66	323
Width	191	66	66	323
Bank	137	46	46	229
Date	137	46	46	229
Frequency	137	46	46	229
Law	137	46	46	229
Insurance	137	46	46	229
Order	137	46	46	229
Plan	137	46	46	229
Person	137	46	46	229
System	137	46	46	229

Appendix A: The complete sets of classes and their statistics.

ARA0003

لا تقوم بالكتابة هنا مطلقا

عربي

Arabic Handwritten Collection Form
Concordia University (Montreal, Canada)
Email: hu_alam@cenparmi.concordia.ca

Address:
1455 de Maisonneuve W - EV3-403,
Montreal QC H3G 1M8, Canada
http://www.cenparmi.concordia.ca

أود أن أقدم جزيل الشكر على تفضلك بتعبئة هذا النموذج
فضلا يرجى مراعاة الآتي عند تعبئة هذا النموذج:
لكل خانة من الخانات التالية، قم بكتابة الفحص المرضي أعلاه، مع مراعاة وضوح الخط وعدم تقطع الكتابة، وفي حالة حدوث أي خطأ، يمكنك استخدام المصحح لإزالة الخطأ ومن ثم الكتابة مرة أخرى، أو بامكانك طلب نموذج آخر إذا تعذر تصحيحه. كذلك عند الكتابة يرجى عدم الخروج عن مربع الكتابة تجنبا لذلك.
وتعبيرا عن امتناني لتعاونك ، فضلا قم بكتابة بريدك الإلكتروني لعد تتمكن من الفوز بأحد هذا الأجهزة.

IPOD

تربح

البريد الأكتروني :

تاريخ الميلاد مجرد أرقام . ١ ٢ ٣ ٤ ٥ ٦ ٧ ٨ ٩ ٨ ٣ ٩ ٨ ٧ ٦ ٥ ٤ ٣ ٢ ١ . ٧ ٥ ٨ ٩ ٨ ٧ ٦ ٥ ٤ ٣ ٢ ١ ٠ ١٩٨١/٧/١٤

٢١ ٧٥ ٦٢ ٤٨ ٣٣ ٩٥ ٦٩ ٤٧ ٦١ ٩٩ ٧٨ ٥٠ ٢٣

١,٥٠ ٤٢٣ ٩٠٨ ٥٦٤ ٧٩٢ ١٣٦ ٣٩٤ ٠٨٦

٧١,٣٥ ٨٢٠٣ ٦٨٣٩ ٥٥٨٤ ٩٧٠١ ٢٢٥٦ ٧١٠٨

٨١,٢٥ ٤٠٧ ٦٨٣٩ ٥٥٨٤ ٩٩١ ٢٤٥٦ ٧١٠٨

٠٤٨٦٨٣ ٢٩٣٧١٢ ٤٥٠٩٦١ ٣٦٥٢٩٧ ٨١٠٦٧٤

١٤٨٦٨٣ ٢٩٣٧١٢ ٤٥٠٩٦١ ٣٦٥٢٩٧ ٨١٠٦٧٤

٨٢٤٩٠٠١ ٠٥٨١٢٩٤ ٢٤٦٠٢٥٧ ١٢٧٩٣٤٠ ١٢٧٩٤٠٠

٨٢٤٩٠٠١ ٠٥٨١٢٩٤ ٢٤٦٠٢٥٧ ١٢٧٩٣٤٠ ١٢٧٩٤٠٠

أ ب ت ث ح ج خ د ذ ر ز س

ش ص ض ط ظ ع غ ف ق ك ل م

ن هـ و ي ى آ إ ة ؤ ء ه

طول عرض حجم وزن ذرزن انش كميه

متر سنتيمتر لتر ملليلتر جرام ملجرام كيلو

Appendix B: Sample of the Form, page 1.

ARA0003

Appendix C: Sample of the Form, page 2.