

Automatic Arabic Handwritten Check
Recognition

BY

SAMEH ABDULLAH BELLEGDI

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In
COMPUTER SCIENCE

January 2013

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **SAMEH ABDULLAH BELLEGDI** under the direction his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER SCIENCE**.



Dr. Sabri A. Mahmoud
(Advisor)



Dr. Adel F. Ahmed
Department Chairman



Dr. Radwan E. Abdel-Aal
(Member)



Dr. Salam A. Zummo
Dean of Graduate Studies



Dr. Salahadin Mohammed
(Member)

5/1/13

Date

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

© Sameh A. Bellegdi

2013

*This work is dedicated to all my family members, especially my parents, my wife
"Marwah," and my son "Abdullah." |*

ACKNOWLEDGMENT

In the name of Allah, the Most Beneficent in Mercy, the Most Merciful.

All praise is due to Allah, *Rubb* of the worlds, and may Allah praise Prophet Muhammad, and render him and his household safe and secure from all evil.

I would like to express my deep gratitude and appreciation, for the inspiration, encouragement, valuable time, wonderful patience, and continuous guidance given to me by my thesis advisor Prof. Sabri Mahmoud. I am grateful to him for continuously reminding me to keep the intention pure for Allah and do what benefits our Arabic language, the language of the Quran. I am also grateful to the other members of my thesis committee Prof. Radwan Abdel-Aal and Dr. Salahadin Adam for their constructive guidance, encouragement, and support.

My heartfelt thanks are due to my father and mother for their prayers, guidance, and moral support throughout my life. Special deep thanks to my beloved wife “Marwah” for her sacrifice, encouragement, and support. I do not forget to thank my son “Abdullah.” I would like to extend my thanks to all my family members especially my brothers Khaled and Saeed for their encouragements, helpful advices, and support.

I would like to take this opportunity to thank Hadhramout Establishment for Human Development, Yemen who gave me the scholarship to pursue my graduate study in King Fahd University of Petroleum and Minerals (KFUPM). Special thanks go to the Chairman of the Board of Trustees, Eng. Abdullah Bugshan, for his valuable support, constructive guidance and precious advices. I thank the General Manager Dr. Saleh Aram and all members of the Board of Trustees and staffs for all that they do for our

community, particularly in support of education. I am also grateful to all professors from Hadhramout in Yemen, who work in Saudi Arabia, for their guidance and precious advices especially Prof. Omar Al-Amoudi.

I would like to acknowledge the Information and Computer Science Department for providing an excellent research environment. I am also grateful to KFUPM for providing me the opportunity to pursue graduate studies and for funding this work through project number RG 1009-1 and RG 1009-2.

I am grateful to Mr. Abdulrazzaq Al-Mashjari for his efforts during my study. I will not forget to thank Mr. Irfan Ahmed and Mr. Mohammed Yahaya for their encouragement and help.

Finally, I extend my thanks to my friends and everyone who helped to get this work done.

TABLE OF CONTENTS

ACKNOWLEDGMENT	V
TABLE OF CONTENTS	VII
LIST OF TABLES	X
LIST OF FIGURES	XI
ABSTRACT.....	XIII
1 CHAPTER 1 INTRODUCTION.....	1
2 CHAPTER 2 LITERATURE REVIEW	3
2.1 Introduction.....	3
2.2 Arabic Handwriting recognition	4
2.3 CENPARMI Arabic Checks' Database	7
2.4 Check Analysis and Regions' Extraction.....	8
2.5 Handwritten Digits Recognition	11
2.6 Courtesy Amount Recognition.....	16
2.7 Legal Amount Recognition.....	19
2.8 Summary.....	23
3 CHAPTER 3 CHECK ANALYSIS AND REGIONS' EXTRACTION.....	25
3.1 Introduction.....	25
3.2 Check Analysis	25
3.3 Region's Extraction	26
3.3.1 Courtesy Amount Extraction	26
3.3.2 Legal Amount Extraction.....	29

3.4	Results and Discussion.....	31
4	CHAPTER 4 HANDWRITTEN DIGITS RECOGNITION.....	34
4.1	Introduction.....	34
4.2	Feature Extraction	34
4.2.1	Segment-Based Features.....	35
4.2.2	Concavity Features	37
4.2.3	Statistics of Structural Features	40
4.2.4	Other Features.....	49
4.3	Classification Methods.....	52
4.4	Experimental Results.....	54
5	CHAPTER 5 COURTESY AMOUNT RECOGNITION	60
5.1	Introduction.....	60
5.2	Preprocessing.....	60
5.3	Courtesy Amount Delimiters Detection	61
5.3.1	Features.....	62
5.3.2	Delimiters Detection Rule-Based Classifier	64
5.4	Comma Detection.....	65
5.5	Courtesy Amount Digits Recognition.....	66
5.6	Experimental Results.....	66
6	CHAPTER 6 CONCLUSION AND FUTURE DIRECTIONS	69
6.1	Conclusion	69
6.2	Future Directions	71
	REFERENCES.....	72
	APPENDIX 1: RULE-BASED CLASSIFIER.....	76

VITAE..... 81

LIST OF TABLES

Table 1 Distribution of CENPARMI Data between Training and Testing Sets	8
Table 2 Comparison between Different Check Processing Techniques	23
Table 3 Sample Feature Vector Values for Digit '3'	36
Table 4 The Confusion Matrix of the Tested Samples with Classifier's Fusion	55
Table 5 Recognition Rates of Svm, Logitboost, Randomforest, Rule-Based and the Two Types of Classifiers' Fusion	56
Table 6 Misclassified Samples.....	58
Table 7 Comparing the Proposed Technique with the Literature	59
Table 8 Delimiters Confusion Matrix of the Tested Samples.....	67
Table 9 Comma or digit Confusion Matrix of the Tested Samples	67
Table 10 Confusion Matrix of the Extracted Digits.....	68

LIST OF FIGURES

Figure 1 Overview of [17] Technique	5
Figure 2 Overview of the System Proposed by [27].....	12
Figure 3 Illustration of Turning Function of [27]: (a) Arabic character Daal (ﺩ), (b) Contour of (a) after Smoothing, (c) Polygon Defined by the Dominant Points of (b), and (d) the Turning Function Representation of (c)	13
Figure 4 Decomposition of Digit 3: (a) Feature Points, (b) Primitives	14
Figure 5 Decomposition and Feature Extraction of [31]	15
Figure 6 Courtesy Amount's Feature Vector Representation of [11].....	17
Figure 7 General Scheme of the Recognition Module of [5].....	18
Figure 8 Parallel Classifiers' Combination [25]	22
Figure 9 Original Check Image.....	26
Figure 10 Check Image after Preprocessing	26
Figure 11 (a) Right Half of Check Image, (b) The Projection of (a) before Dilation.....	27
Figure 12 (a) Horizontal Image Dilation of the Right Half, (b) Horizontal Projection of the Check	27
Figure 13 (a) Courtesy Amount Box after Determining the Horizontal Indices, (b) Vertical Projection of (a)	28
Figure 14 (a) Courtesy Amount Box after Dilation, (b) Vertical Projection of (a)	29
Figure 15 The Extracted Courtesy Amount Box	29
Figure 16 The Extracted Courtesy Amount after Removing its Boundaries	29
Figure 17 (a) The Legal Amount in the Check Image, (b) Horizontal Projection of (a), (c) The legal Amount after Removing the Noise	30
Figure 18 (a) Legal Amount Region for Blank Check, (b) Vertical Projection of (a).....	31
Figure 19 The Extracted Legal Amount	31
Figure 20 Bad Data	32
Figure 21 (a) Check Form, (b) Extracted Legal Amount with Unremoved Printed Text.	32
Figure 22 (a, c) Noisy Check Forms, (b, d) Extracted Legal Amounts	33
Figure 23 Location of Horizontal Scan Lines with One and Three Segments	36
Figure 24 Convex Shape from Top.....	38
Figure 25 Concave Shape from Top	39
Figure 26 Digit "8" has N_{mhs} in the Middle Third $> THK$	41
Figure 27 leftDistIncrease Feature Justification	43
Figure 28 rightDist Feature Justification	43
Figure 29 TopDist1 Feature Illustration: (a) Satisfied Condition, (b) not Satisfied Condition.....	44
Figure 30 TopDist2 Feature Illustration: (a) Satisfied Condition, (b) not Satisfied Condition.....	45
Figure 31 topDistCount Feature Illustration Example.....	46

Figure 32 bottomDist1 Feature Illustration: (a) Satisfied Condition, (b) not Satisfied Condition.....	47
Figure 33 SizeVSeg Feature Justification: (a) Satisfied Condition, (b) not Satisfied Condition.....	49
Figure 34 LeftPeaks Feature Illustration.....	50
Figure 35 Digit '7' Divided into 3 x 4 Divisions.....	51
Figure 36 Illustrating Upper, Lower Rightmost and Upper Leftmost Points	52
Figure 37 Pseudo Code for the Rules of Digit Zero	53
Figure 38 Comparing the Average Recognition Results with SVM, Logitboost, Randomforest and the Two Types of Classifiers' Fusion.....	56
Figure 39 Courtesy Amount Recognition Phases	60
Figure 40 Mean Upper Horizontal Location.....	61
Figure 41 Courtesy Amount with Special Noisy Component	61
Figure 42 Courtesy Amounts with Different Delimiter Types	62
Figure 43 Inner-loop Feature Illustration.....	63
Figure 44 Not-one Feature Illustration	63
Figure 45 Terminals' Center of Gravity Illustration.....	64
Figure 46 Pseudo Code for the Rules of Delimiters	65
Figure 47 Mean Lower Horizontal Location	66

ABSTRACT

Full Name : [Sameh Abdullah Bellegdi]
Thesis Title : [Automatic Arabic Handwritten Check Recognition]
Major Field : [Computer Science]
Date of Degree : [January 2013]

[In this thesis, we address the problem of automatic processing of Arabic handwritten checks. We proposed techniques for automatic extraction of courtesy and legal amounts and Arabic handwritten digits and courtesy amount recognition. Real data from CENPARMI Arabic check database is used. For automatic extraction of courtesy and legal amount fields, we used image dilation, horizontal and vertical projections, and image registration. We achieved extraction rates of 100% and more than 91% for courtesy and legal amounts, respectively. More emphasis is given to Arabic handwritten digits recognition. Novel structural features that are based on the structure of Arabic digits are proposed for automatic recognition of Arabic (Indian) bank check digits. In addition, a rule-based classifier is implemented. Support Vector Machine (SVM), LogitBoost, and RandomForest classifiers are used in this work. Classifiers' fusion with majority voting is used with and without the rule-based classifier. Recognition rates of 98.95% and 99.08% are achieved using fusion of the statistical classifiers alone and with the rule-based classifier, respectively. The achieved recognition rates outperform the published work using the same database. The experimental results indicate the effectiveness of the structural features and the rule-based classifier for recognizing Arabic digits. The proposed technique for automatic recognition of courtesy amounts has four main phases: preprocessing, delimiters detection, commas detection, and digits

recognition. Through experimentations, the proposed technique has demonstrated promising performance. In addition, the proposed technique can be further extended. To the best of our knowledge, our work is the first to tackle this problem on the given database.

|

ملخص الرسالة

الاسم الكامل: سامح عبدالله بلقدي

عنوان الرسالة: التعرف الآلي على الصكوك العربية المكتوبة يدوياً

التخصص: علوم الحاسب الآلي

تاريخ الدرجة العلمية: صفر ١٤٣٤

تطرقنا في هذه الأطروحة لموضوع التعرف الآلي على الصكوك العربية المكتوبة يدوياً. اقترحنا أساليب لاستخراج الأجزاء الخاصة بالمبالغ الرقمية والحرفية، والتعرف الآلي على الأرقام العربية المكتوبة يدوياً، والتعرف الآلي على المبالغ العربية الرقمية. وقد استخدمنا قاعدة بيانات حقيقية من "قاعدة بيانات سينبارمي (CENPARMI) للصكوك العربية" للتحقق من الطرق المقترحة. استخدمنا طرق تمدد الصورة والإسقاطات الرأسية والأفقية ومقابلة الصور لاستخراج الأجزاء الخاصة بالمبالغ الرقمية والحرفية من الصكوك. حققنا معدل استخراج بلغ ١٠٠٪ لاستخراج أجزاء المبالغ الرقمية، وأكثر من ٩١٪ لاستخراج أجزاء المبالغ الحرفية. قمنا بالتركيز على موضوع التعرف الآلي على الأرقام العربية المكتوبة يدوياً واستحدثنا سمات تعتمد على بنية الأرقام العربية للتعرف الآلي على هذه الأرقام. كما طورنا قواعد للأرقام العربية للتعرف على هذه الأرقام تلقائياً. وقد استخدمنا بعض تقنيات التعلم الآلي كآلات الدعم الإشعاعي وغيرها، مفردة ومع القواعد المطورة. وقد حصلنا باستخدام طريقة أغلبية التصويت لدمج نتائج تقنيات التعلم الآلي مفردة على نسبة تعرف تبلغ ٩٨,٩٥٪، وفي المقابل، بدمج تقنيات التعلم الآلي مع القواعد المطورة كانت نسبة التعرف ٩٩,٠٨٪. وبذلك فقد فاقت نتائجنا كل نتائج الأبحاث المنشورة باستخدام نفس قاعدة البيانات، كما بينت النتائج التجريبية فعالية السمات البنائية وقواعد التعرف على الأرقام العربية. يعتمد أسلوبنا المقترح للتعرف الآلي على المبالغ الرقمية على أربع مراحل رئيسية: مرحلة ما قبل المعالجة، والكشف عن محددات الأرقام، والكشف عن الفواصل، والتعرف على الأرقام. إضافة إلى أن مشكلة التعرف على المبالغ الرقمية لم يسبق معالجتها من قبل على قاعدة البيانات المستخدمة، فقد أظهرت التقنية المقترحة أداءً واعداً كما أنها تمتاز بإمكانية تطويرها وزيادة أدائها.

CHAPTER 1

INTRODUCTION

Handwriting recognition systems can contribute to the advancement of the automation process and can improve the interaction between human and machine in many applications, such as office automation, mail sorting, signature verification, and check recognition.

Many countries around the globe have implemented check truncation systems (CTS) [or image-based clearing system (ICS)] to process bank checks automatically [1]. However, bank checks in Arabic countries are processed manually. There are market demands for recognizing bank checks automatically. A large number of research papers have already been published that address the automatic recognition of courtesy and/or legal amounts of bank checks. However, comparatively few of these papers addressed the problem of automatically recognizing Arabic handwritten bank checks.

Arabic is written from right to left. The Arabic Alphabet has 28 letters. Additional marks (hamza, shadda, dots, diacritical marks, etc.) may change the letter and the word meaning. Each Arabic letter has two, three, or four shapes. The shape of a letter is determined by its position, initial, medial, final, or isolated, within the word. Most of the letters can be connected from both, the right and left, sides. However, there are six letters which can be connected from the right side only. Letters of a word may overlap vertically (even without touching). Arabic Letters are joined together along the baseline to form

words or sub-words. The baseline is an important characteristic of Arabic writing. It is a medium line in the Arabic word in which connections between the successive characters take place.

This work is addressing the analysis of the check image and extracting its courtesy and legal amount fields. We proposed the use of image dilation, horizontal and vertical projections, and image registration to extract courtesy and legal amount regions from real bank checks of the CENPARMI database [2].

Furthermore, this work presents a technique for automatic recognition of Arabic (Indian) bank check digits. A rule-based classifier is developed using novel structural features. In addition, three classifiers are used (viz. Support Vector Machine (SVM), LogitBoost, and RandomForest). The classifiers output are combined using majority voting.

Moreover, we present a technique for automatic recognition of courtesy amounts extracted from the CENPARMI database [2]. Four main phases are proposed: preprocessing, delimiters detection, commas detection, and digits recognition.

The rest of this thesis is organized as follows: Chapter 2 presents a literature review of Arabic handwritten recognition, including digits recognition, and check recognition. Check analysis and regions' extraction is discussed in Chapter 3. Chapter 4 describes the proposed technique for automatic recognition of Arabic (Indian) bank check digits. Courtesy amount recognition is presented in Chapter 5. Chapter 6 concludes this thesis and provides some future directions.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Automatic bank check recognition problem has been addressed by many researchers. There are many techniques developed in the literature to tackle this problem [3–8]. Comparatively limited number of papers addressed the problem of automatic recognition of Arabic checks [9–13], hence it needs to be explored more. A comprehensive survey on this topic can be cited in [1].

Arabic handwriting recognition is challenging because Arabic text (both machine printed and handwritten) is cursive in general, letter shapes are context sensitive, the problem of diacritical marks, and dots, to name a few. Legal amount recognition is a limited vocabulary problem; hence a dictionary and holistic techniques may be used. Holistic techniques are not applicable to the general Arabic handwritten text recognition due to the huge number of unique words. Even with very large dictionary, there will be words that are in the language and not in the dictionary. Section 2 presents a review of Arabic handwriting recognition. An overview of the used CENPARMI Arabic checks' database [2] is described in Section 3. The problem of automatic recognition of Arabic handwritten bank checks involves check analysis and regions' extraction and digits, courtesy and legal amount's recognition. Sections 4 – 7 provide a review of some research work related to these tasks.

2.2 Arabic Handwriting recognition

Line segmentation has been studied by Li et. al. [14] as an essential stage in handwritten document image analysis since it contributes to the document structure extraction and to text recognition. The authors claimed that their paper is the first to study the text line segmentation problem from the density estimation perspective. By using an anisotropic Gaussian filtering [15], the system starts by estimating a probability map for a given image, where each element represents the probability that a considered pixel belongs to a text line. For visualization, the probability map is rescaled to a gray-scale image. Then the authors adopted the level set method to determine the boundary of neighboring text lines [16]. Finally, geometrical constrains are imposed to group isolated connected components or segments to their closest major text lines. The system was tested on a data set of 7,528 heterogeneous handwritten documents from 439 writers in nine scripts, such as Hebrew, Hindi, Japanese, and Persian. In addition, the authors evaluated their method on a large data set of Arabic documents containing 166,071 images. In terms of the pixel-level hit rate, they reported 94.7 percent accuracy.

Al-Hajj et. al. [17] developed an HMM-based Arabic handwritten recognition system without pre-segmentation. They claimed that their system is robust for a wide range of orientation angles. As illustrated by Figure 1, the system starts by baseline estimation at a preprocessing stage. A set of 28-features are then extracted within three sliding windows of different orientations, 11 of them are baseline dependent. Each orientation is associated to one of three HMM-based classifiers. One of these classifiers uses a vertical window (frame), while the other two observe the image through slanted frames, one

slanted to the left and the other slanted to the right. The authors used two types of features viz. features based on foreground (black) pixel densities and concavity features that reflect local concavity and stroke directions. The extracted features are then fed into the recognition system, and Viterbi decoding algorithm is used for simultaneous classification and implicit segmentation. Finally, each classifier produces a list of word candidates with their scores. Those candidate lists are then fused at a decision level in which the three individual classifiers are combined by using three combination strategies: multilayer-perceptron (MLP) neural network-based combination, sum rule and majority rule. The best combination scheme is obtained with the MLP based combination. Higher than 90 percent recognition rate was reported on the benchmark IFN/ENIT database [18]. However, this database is for city names and hence not a natural Arabic handwritten text.

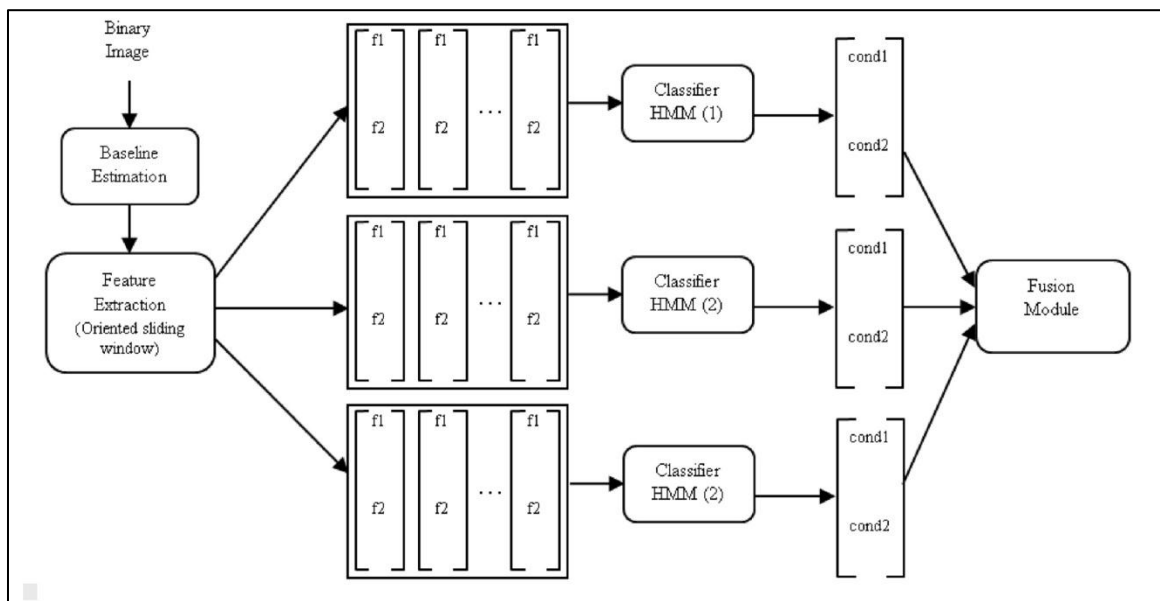


Figure 1 Overview of [17] Technique

Pechwitz and Maergner used 160 semi-continuous HMM-models representing the Arabic characters or shapes [19]. The system has four main steps preprocessing, normalization,

feature extraction, and recognition. Preprocessing is applied to perform connected component analysis, extract a contour representation of the image, perform noise reduction filtering, and skeletonization. In order to reduce writing style variability, the system accomplishes some normalization steps, such as writing line skew normalization, and horizontal word width scaling. The features are collected using a sliding window approach. Karhunen-Loeve transformation [20] is performed in order to reduce the number of features. Due to the fact that Arabic characters might have several shapes depending on their position in a word, a semi-continuous HMM (SCHMM) is generated for each character shape. This SCHMM has 7 states, in which each state has 3 transitions a self-transition, a transition to the next state, and one allowing skipping a single state. The training process is performed by a segmental k-means algorithm. The recognition is carried out by applying a frame synchronous network Viterbi search algorithm together with a tree-structured lexicon representing the valid words. The system was tested on IFN/ENIT database [18] with a reported 89 percent word-level recognition rate. However, the database is limited to city names, not a general Arabic handwritten text database.

ElBaati et. al. [21] developed a recognition system based on the automatic reconstruction of the trajectories of a handwritten word and restoration of their temporal order. For a given connected component having n segments, there are $(n!)$ possible permutations of these segments to traverse. A genetic algorithm (GA) [22] was used to obtain the best permutation. A module of trajectory signal sampling considering trajectory curvatures was calculated using the correlation between the angular velocity and the curve. Finally, the trajectory is sampled at fixed time intervals by traversing it with a curvilinear

velocity. The beta-elliptical modeling was adopted to calculate the characteristics of the rebuilt trajectory. The system was validated by HMM Tool Kit (HTK) recognition system [23] using IFN/ENIT database [18]. They reported more than 83 percent recognition rate. Similar to the work of [17] and [19], the same comments apply to the use of IFN/ENIT database.

2.3 CENPARMI Arabic Checks' Database

Al-Ohali et al. [2] of the Centre for Pattern Recognition and Machine Intelligence (CENPARMI) in Montreal developed an Arabic check database extracted from 3000 out of 7000 real bank checks provided by AlRajhi Bank, Saudi Arabia. The database composed of images of 3000 samples of Arabic checks, 2499 samples of courtesy and legal amounts, 15,175 samples of Arabic (Indian) digits, and 29,498 samples of Arabic sub-words. The data is divided randomly into training and testing sets such that the training set includes 66–75% of the available data. Training and testing data are further divided into touching and non-touching data. Table 1 shows the sizes of training and testing sets.

Each courtesy amount is tagged so that it contains a sequence of coordinates and tags of objects. Objects may include Indian digits, delimiters, commas, decimal points or noise. However, within our study we found that some tags are incorrect and some objects, specifically noisy and broken data, are not tagged. Similarly, each legal amount contains a sequence of coordinates and tags of objects. Objects may include sub-words, or noise. Arabic digits and sub-words have references to the original courtesy and legal amounts, respectively.

Table 1 Distribution of CENPARMI Data between Training and Testing Sets

		Number of Samples			
		Arabic courtesy amounts	Arabic digits	Arabic legal amounts	Arabic sub-words
Training	Touching	266	243	838	1,066
	Not touching	1,513	10,536	941	19,813
	Total	1,779	10,779	1,779	20,879
Testing	Touching	94	72	321	447
	Not touching	626	4,324	399	8,172
	Total	720	4,396	720	8,619

The authors proposed grammars and algorithms as a validation process to verify the correctness of the tagging process. They compared numerical values of the tags of legal and courtesy amounts of each check. If the numerical values obtained from the tags of legal and courtesy amounts match, the tagged check is approved. Otherwise, the check is tagged manually. The validation process approved about 83% of the 3,000 tagged checks.

2.4 Check Analysis and Regions' Extraction

Analyzing and segmentation of the check image into different regions is an important task in check recognition systems. Cheriet et. al. [11] applied dynamic thresholding for legal amount region's binarization. In addition, other preprocessing techniques are applied like baseline detection and correction, noise removal, and thickening of digits' strokes.

Wang [3] designed an automatic recognition system for Chinese bank checks. He used Ostu's image binarization algorithm [24]. Median filtering and connected component

analysis are employed to remove image noise. Skew correction is performed using Hough transformation. Layout analysis is used to locate the different regions using an adaptive template searching algorithm and utilizing Hough transformation and vertical and horizontal projection histograms.

Palacios et. al. [5] developed a system to automatically recognize courtesy amounts. Initially, the input image is binarized based on dynamic thresholding. The courtesy amount string is extracted utilizing the concept of minimum bounding rectangle (MBR). After organizing information into component blocks, the strings are built based on horizontal collinearity and proximity. Then, the courtesy amount is identified by using some rules to identify the handwritten text and to locate the courtesy amount string.

Farah et. al. [25] proposed a system for recognizing handwritten Arabic legal amounts. Ostu's algorithm image binarization [24] was employed. They used vertical projection method to extract words from the literal amount and horizontal projection to extract the baseline.

El-Melegy et. al. [9] proposed an approach for recognizing offline handwritten Arabic literal amounts. Their system considered a word as a unique entity. Morphological closing is applied to connect objects that are close to each other. The baseline is detected using the horizontal projection. Image contours of the word is extracted and labeled to determine the parts that constitute the word. Each piece of Arabic word (PAW) is then classified as primary or secondary depending on its size.

Samoud et. al. [13] proposed the use of two methods, viz. Mathematical Morphology (MM) and Hough Transform (HT), for automatic extraction of handwritten regions of

Arabic bank checks. They used CENPARMI Arabic checks' database [2]. Using MM method, they applied MM filter to an image binarized using Ostu's image binarization algorithm [24]. They extracted the courtesy amount region applying horizontal and vertical closing filters. The date and literal amount regions are extracted by constructing connected components. They applied horizontal and vertical structuring element to connect horizontally and vertically neighboring zones. Using HT, the authors extracted courtesy amount by extracting the rectangle shape in which the amount is localized. After eliminating the courtesy amount zone, they extracted literal amount zone by detecting all horizontal lines. Then, the accumulator with maximum value is found. After that, an estimation of the height of the handwritten text to extract the handwritten literal amount is used. In their work of extracting literal amount, the authors depend on two wrong hypotheses viz. the literal amount is written only on one line and that line is the longest one on the check. Another limitation of this work is the used database which is not of real checks.

In their approach of automatically reading bank checks, Kaufmann and Bunke [6] addressed skew, slant, and size variations. The skew angle is extracted using horizontal projection. The slant angle is computed using the histogram of different directions of contours. A shear transformation is then applied. The scale factor in the x-direction is calculated by dividing the width of a word by the estimated number of characters. The scale factor in the y-direction is calculated after detecting upper and lower baselines. The scale factor is represented as "the ratio of the middle area (bounded by these two baselines) to a standard middle area size".

2.5 Handwritten Digits Recognition

Mahmoud and Al-Khatib proposed a technique based on Log Gabor filters for the automatic recognition of Arabic (Indian) bank check digits [26]. The authors used Log Gabor-based features with different numbers of scales, orientations and image segments. The authors applied their technique to CENPARMI Arabic checks database [2]. They used a feature vector of $6 \times 4 \times 3 \times 3 \times 2 = 432$ features using 6 orientations (0, 30, 60, 90, 120 and 150), 4 scales (wavelengths of 3, 6, 12 and 24), 3×3 digit image segments and using the mean and variance of each segment. K-Nearest Neighbor (K-NN), Hidden Markov Models (HMM), Support Vector Machines (SVM) and Nearest Mean (NM) classifiers were used. A reported average recognition rates of 98.75%, 98.62%, 94.43%, 97.21% and 98.95% with 1-NN, 3-NN, NM, HMM and SVM classifiers, respectively. The authors reported that their technique outperforms the previously published work using the same database.

Parvez and Mahmoud proposed a method that utilizes polygonal approximations and fuzzy directional edges for recognition of Arabic handwritten alphanumeric [27]. Figure 2 shows a general overview of the proposed algorithm. After binarizing and smoothing a character image, the character contour is extracted and polygonal approximation i.e. dominant points schemes are constructed. Given a contour $C = P_i (x_i, y_i)$, $i = 1, 2, \dots, n$, the proposed algorithm selects an initial set of dominant points. Let $\{a_1, a_2, \dots, a_n\}$ be the Freeman chain code for n points of C . A point P_i is called a break point if a_i and a_{i+1} are different. The initial set of dominant points is represented by the set of all break points in C . The directions and length features are extracted from the polygonal approximation.

Character models are built in the training phase using the extracted features. A classifier based on fuzzy logic and turning angle functions is utilized in the recognition phase. Figure 3 shows the turning function for the Arabic character Daal (ﺀ). The possibility of high computational complexity of the proposed method is considered a drawback. The authors used a database of handwritten Arabic characters and another one of handwritten Arabic numerals. They used ADBase [28] to test their technique on Arabic digits. The reported average recognition rate is 97.18%.

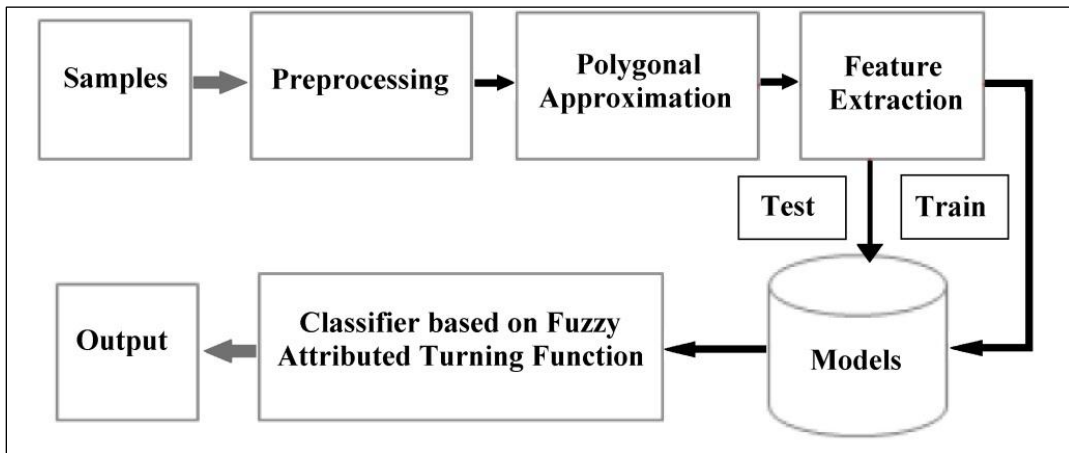


Figure 2 Overview of the System Proposed by [27]

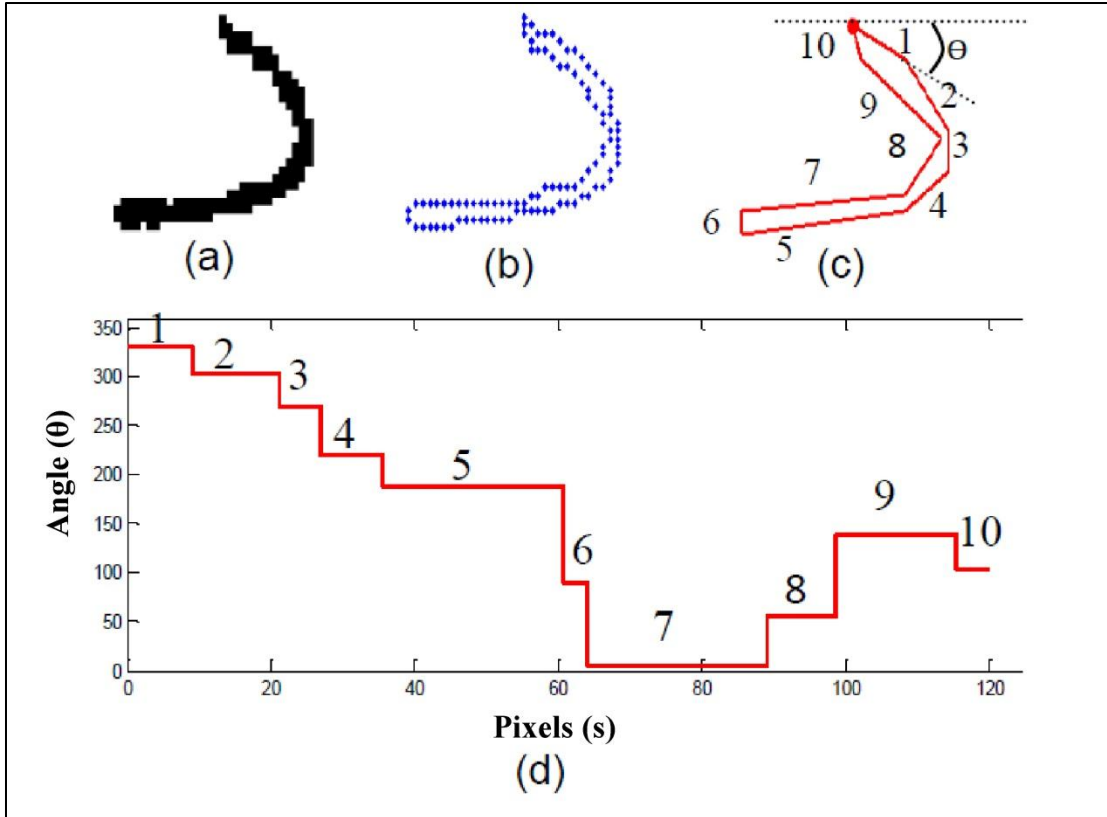


Figure 3 Illustration of Turning Function of [27]: (a) Arabic character Daal (د), (b) Contour of (a) after Smoothing, (c) Polygon Defined by the Dominant Points of (b), and (d) the Turning Function Representation of (c)

Salehpour and Behrad [29] presented a technique for classification and recognition of Farsi handwritten digits. Some preprocessing steps were applied (viz. image binarization, noisy points' removal using morphology based algorithms, and size normalization to 40×40). All images in the database are represented as one dimensional vector. Then, the features are extracted using principle component analysis (PCA) and PCA with linear discrimination analysis (PCA-LDA). The authors proposed the use of cluster-based weighted SVM (CBWSVM) in which cluster-based weighted support vector machine is used. They tested their technique using 2000 samples from Farsi handwritten database with 7600 handwritten digits with and without rotation [30]. They reported recognition

rates for different number of PCA features, viz. 10, 20, 30, 40, and 50. The best reported recognition rate is 96.5% with 30 PCA-LDA features.

Mozaffari et. al. [31] developed structural and statistical features to recognize handwritten Farsi/Arabic digits. Three standard feature points were detected; viz. "T" point that has only one black pixel in its 8-neighbors, "Y" point that has three black pixels, and "X" point that has four black pixels. These points were used to decompose the skeleton into primitives. A primitive was defined as the skeleton segment which starts from a feature point and ends at another feature point as shown in Figure 4. If a character has single primitive, it is divided into four equal parts and two equal parts otherwise. Each part is traced and changes in X and Y directions are recorded. The average and variance of X and Y coordinates changes in each primitive are used as features totaling eight features. The features for all primitives were concatenated and PCA is used to normalize their size. Figure 5 shows a flowchart of the decomposition process and feature extraction. Nearest-neighbor was used as a classifier. The authors tested eight digits with 280 samples of each were used for training and 200 for testing. The reported recognition rate is 94.44%.

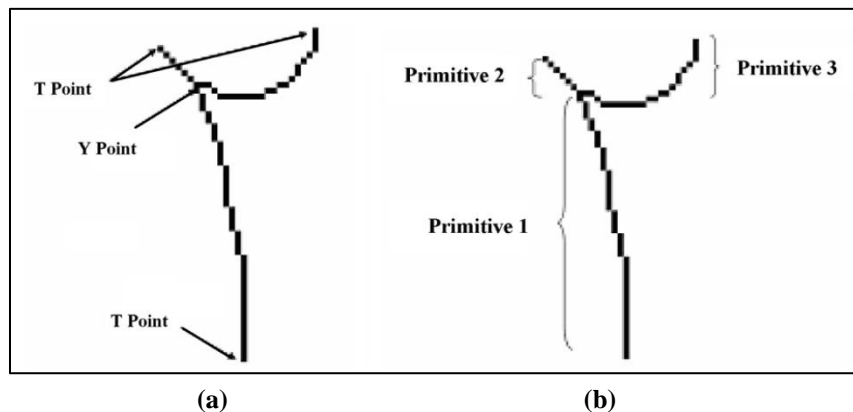


Figure 4 Decomposition of Digit 3: (a) Feature Points, (b) Primitives

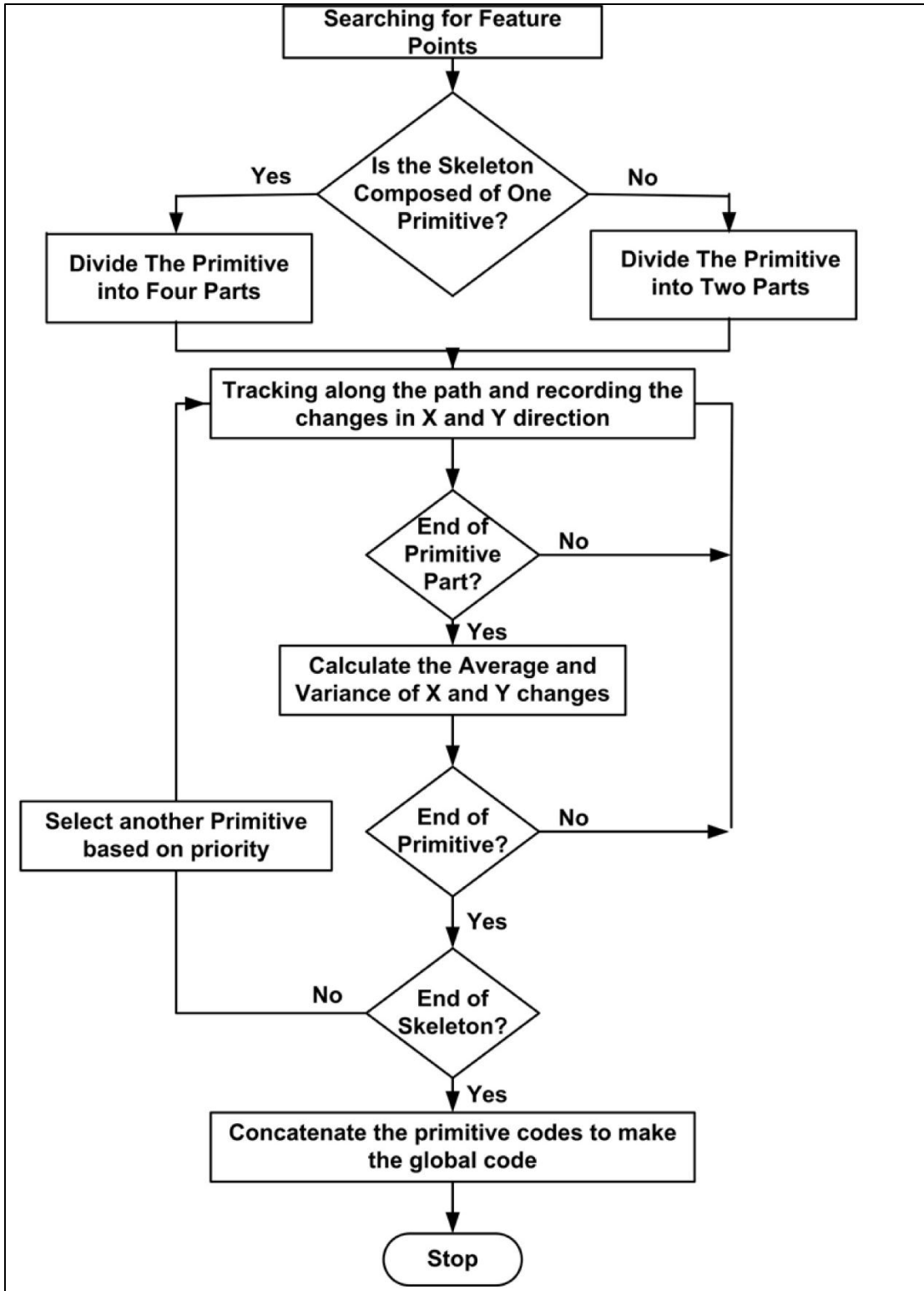


Figure 5 Decomposition and Feature Extraction of [31]

Ba-Karait and Shamsuddin [32] proposed a handwritten digits recognition technique based on Particle Swarm Optimization (PSO) [33]. They used two types of features, viz. local and global features. Information about image pixels and their neighbors is extracted as local features. As global features, the authors calculated the number of horizontal crossings (HC) and the total number of times the number of horizontal crossings change in the image (HC_i) from j to i ($j \neq i$) crossings. Similarly, VC and VC_i features are calculated as HC and HC_i, respectively, in the vertical direction. Furthermore, they divided the digit horizontally into three parts and used the pixels distributions in sub-images as global features. Each class of ten digits is encoded as a centroid in multidimensional feature space. PSO is used to probe the optimal position for each centroid. A single encoded particle was used. The authors tested their technique on MINST database [34] with a reported 73.10% recognition rate.

2.6 Courtesy Amount Recognition

There are many approaches that are proposed in the literature to recognize courtesy amount. As shown in Figure 6, Cheriet et. al. [11] used a feature vector of eight standard freeman directions, five values for the curvature of the contour strokes to automatically recognize Arabic handwritten digit. The feature vector contains a mesh grid representation that accounts for counting the number of pixels pertaining to morphological patterns like hole, mountain, and buckle patterns. Two classifiers are used; (viz. a neural network and an SVM). The used neural network classifier is a multi-layer perceptron with one hidden layer. Hyperbolic tangent is used as an activation function for the hidden and output neurons. The SVM-based classifier is trained with respect to one-

against-one strategy. The best error rate is reported with the SVM-based system that yields 1.92% with no rejection.

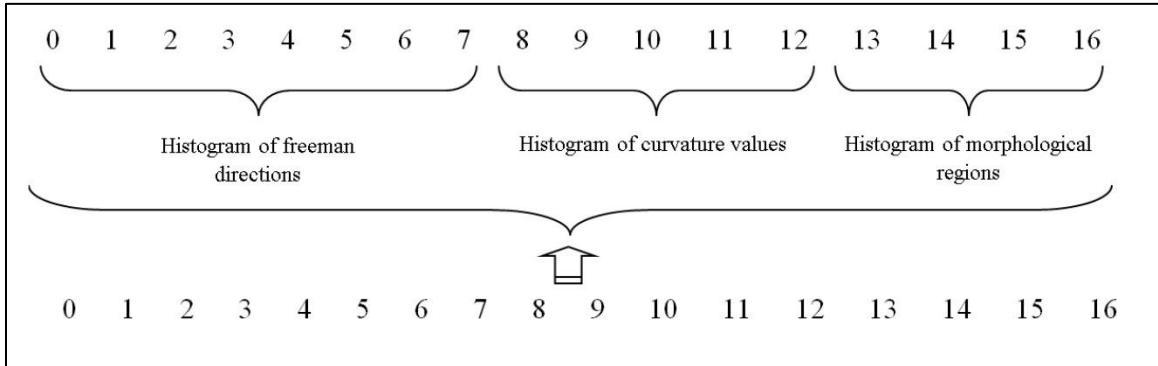


Figure 6 Courtesy Amount's Feature Vector Representation of [11]

Palacios et. al. [5] proposed a feedback system that classifies each segment as being digit, multiple-segment, or punctuation. Digits are sent directly to the recognition module while multiple-segments are divided and then classified again. Any segment classified as being a digit is recognized as a number with a given level of confidence or it can be rejected. As shown in Figure 7, the used classification model is implemented as an array of three or four neural networks working in parallel. Their results are analyzed by an arbiter function. The networks are three-layered, fully connected, feed-forward MLP, with 204 nodes as input layer, 50 nodes in the hidden layer and 10 outputs (for the 10 digits). As a post processing step, the resulting string is analyzed to determine whether the recognized value is a valid amount for a bank check or not. The proposed technique may fail to recognize some digits correctly and this failure could not be validated because the system does not recognize the legal amount.

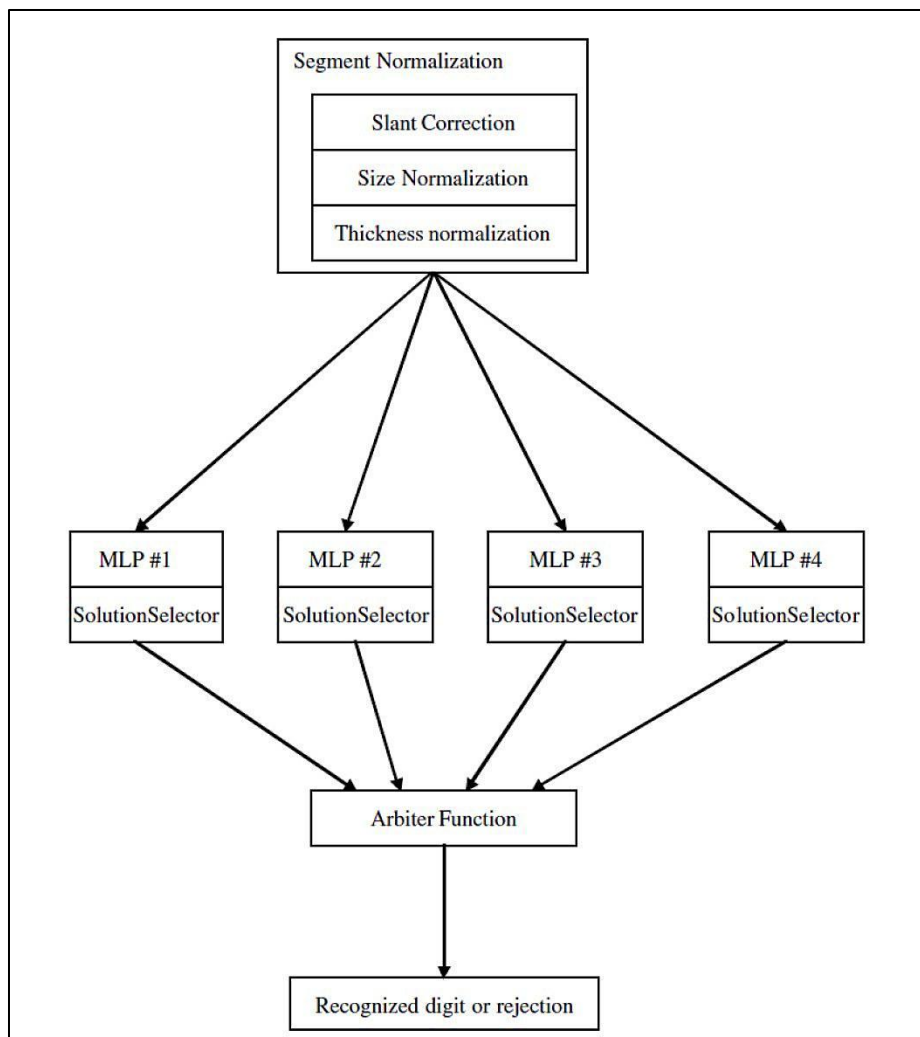


Figure 7 General Scheme of the Recognition Module of [5]

Kaufmann and Bunke [6] developed a system for reading handwritten German bank check amounts. Their system for reading courtesy amount begins by dividing the input numeral string into independent groups of one or more digits. Each group is then processed by a cascade of two recognition modules; a digit detection module which recognizes only isolated digits, and a segmentation-free module which recognizes the remaining digit groups. The digit detection module uses projection-based features. It uses four histograms of digit black pixels, four histograms of black-to-white transition pixels,

and eight contour profiles. The segmentation-free module computes its features from contour information (the direction at each contour point and the number of contour pixels). Both sub-recognizers use a fully connected feed-forward multi-layer perceptron for classification with back-propagation algorithm. The networks have one hidden layer consisting of 60 neurons. The authors applied the winner-takes-all principle to decide the identity of an input pattern. Finally, global decision module merges all results to a digit string. The system was tested on a database with real checks from Swiss postal services. The reported rate is 79.3% with zero rejection.

Wang [3] designed an automatic recognition system for courtesy amount. It can be simply segmented based on grid region analysis. The features for recognition include black pixel distribution, stroke line elements and frequency coefficients. Artificial neural network is used to recognize the courtesy amount. The author applied his experiments on two sample sets of Chinese bank checks containing 900 and 600 check images respectively. The reported recognition rates for the two sample sets are 92.86% and 93.44%, respectively. The author did not describe the chosen samples and their characteristics.

2.7 Legal Amount Recognition

Several techniques are proposed in the literature to address the problem of recognizing legal amounts. Cheriet et. al. [11] applied skeletonization to the input image to facilitate easier extraction of analytical features of legal amount. Then, a graph representation of the skeleton is built and transformed to a tree. Pen-trajectory is then estimated. After that, vector quantization is applied to facilitate efficient use of HMM. Clustering is performed

to reduce within-class variations in human handwriting. A left-to-right HMM is used to model each cluster. The model of a cluster ω_i , is noted $\lambda_i (S_i, \pi_i, A_i, B_i)$ where S_i represents the number of states, π_i is the initial probability of each state, A_i is the matrix of transition probabilities and B_i is the emission probabilities. The number of states in each model is defined in relation to the number of letters within each sub-word, and therefore is unique for all models of the same class. Models are trained using the Baum–Welch algorithm with maximum likelihood (ML) learning criterion. The reported sub-words recognition rate is 73.53%. Legal amount interpretation is then performed which intends to translate proper sequences of sub-word codes into their equivalent numerical values. One limitation of this system is the use of skeletonization as it is time consuming and adds some ambiguities to the skeleton of words.

Kaufmann and Bunke [6] proposed an HMM based approach for automatic legal amount recognition. They used bitmap-based features that are the positions of the pixels set within a sliding window. The features are extracted by using a sliding window (216×8 pixels) which is further divided into squares (4×4 pixels). The resulting vectors from counting the black pixels in those squares are then reduced by a principal axis transformation to a dimension of 18. The authors defined a grammar to construct all valid legal amounts up to one million. Based on that grammar, they organized the HMMs in a hierarchical network. Baum–Welch algorithm is used in the training phase while a modified Viterbi algorithm is used in the classification phase. The reported recognition rate is 71.9%.

Wang [3] used a feature vector generated from stroke line elements to recognize Chinese legal amount by using HMM. He used 366 states. Directional element features of strokes

in sub-regions are employed to determine the observation state sequence. Each normalized character image is divided into 8×8 sub-regions. The number of contour points belonging to four types of line elements is then calculated. Four types of line elements are used; vertical, horizontal and two oblique lines slanted at ± 45 degree. The normalized numbers are used to generate the feature vector. The recognition rates are 90.32% and 91.68% for testing on the two sets of check samples. However, the authors did not provide details of the used sets and their characteristics.

Farah et. al. [25] used a set of structural features; the number of descenders, ascenders, loops, dots, and sub-words. They proposed a multi-classification system composed of three classifiers working in parallel as shown in Figure 8. These classifiers are multilayer neural network (MLP), k-nearest neighbor (k-NN), and fuzzy k-nearest neighbor (fuzzy k-NN). The neural network is formed using 21 neurons for each of the input and hidden layers, and 48 neurons for the output layer. Sigmoid is used as an activation function for the hidden layer. The training set of k-NN consists of M samples for each of the 48 lexicon words with their features vector. Fuzzy k-NN uses the information of crisp nearest neighbor to calculate the membership degree of each neighbor. The authors used their own database of 4800 words, where 1200 words were used for training the classifiers. They reported 89%, 91%, and 92% recognition rates for k-NN, MLP, and fuzzy k-NN, respectively. The classification results are combined using score summation [35]. Each classifier produces a list of three candidate words with their confidence values. A new list is then produced by merging the three lists and the confidence values are summed. Syntactic analyzer uses the latter list to generate a syntactically correct literal amount. The reported words recognition rate after the combination stage is 96%. One

limitation of this work is the use of special database, so this work may not be comparable to other works.

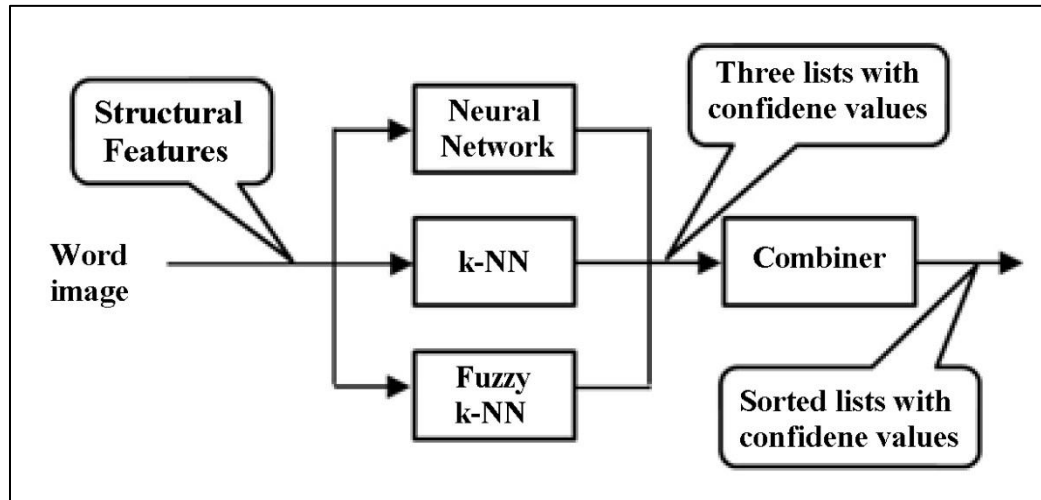


Figure 8 Parallel Classifiers' Combination [25]

El-Melegy et. al. [9] used a set of structural features to recognize Arabic literal amounts. They used number of PAWs, descenders, ascenders, loops, and dots. Four classifiers are used independently. The classifiers are k-nearest neighbor, Bayesian, decision tree, and neural network classifiers. The authors used the Euclidian distance with k-nearest neighbor. The multivariate normal density is chosen as a model for the Bayesian classifier. In decision tree classifier, c4.5 algorithm [36] is used to build a decision tree from a set of training samples. Back propagation feed forward neural network with three layers is used as a classifier. The hidden layer has 300 hidden units. The output layer consists of 50 units. The used database has 4970 words [37]. The training set has 50 samples for each of the 50 lexicon words. The reported recognition rates are 79%, 80%, 83%, and 86.5% for Bayesian, decision tree, k-nearest neighbor, and neural network classifiers, respectively. The used database is not of real checks and this is considered a limitation.

2.8 Summary

The following table summarizes and compares the discussed techniques.

Table 2 Comparison between Different Check Processing Techniques

Ref.	Check Analysis and Regions' Extraction	Digits and Courtesy Amount			Legal Amount		
		Features	Classifier	Recognition Rate	Features	Classifier	Recognition Rate
[3]	Utilized Median filtering and connected component analysis to remove image noise. Skew correction is performed by utilizing Hough transformation. Regions are extracted using adaptive template searching algorithm, Hough transformation and vertical and horizontal projection histograms.	Black pixel distribution, stroke line elements and frequency coefficients.	Neural network	92.86 and 93.44%	stroke line elements	HMM	91.68%
[5]	Image binarization based on dynamic thresholding. The courtesy amount string is extracted utilizing the concept of minimum bounding rectangle (MBR).	Used the image pixels.	MLP	92.70%	<i>Not addressed.</i>		
[6]	Skew angle is extracted using horizontal projection. Slant angle is computed using the histogram of different directions of contours. The scale factor in the x-direction is calculated by dividing the width of a word by the estimated number of characters. The scale factor in the y-direction is calculated after detecting upper and lower baselines. It is represented as the ratio of the middle area bounded by the baselines to a standard middle area size.	projection-based and contour-based features	MLP	79.30%	bitmap-based	HMM	71.9%
[9]	Morphological closing is applied to connect objects that are close to each other. The baseline is detected using the horizontal projection. Image contours of the word is extracted and labeled. Each PAW is classified as primary or secondary.	<i>Not addressed.</i>			number of PAWs, descenders, ascenders, loops, and dots.	k-NN, Bayesian, decision tree, and neural network	83, 79, 80, and 86.5%
[11]	Legal amount region binarization based on dynamic thresholding, thickening of digits stroke, base line detection and correction, and noise removal	Freeman directions, curvature values, and morphological regions.	MLP and SVM	98.08%	Pen-trajectory	HMM	73.53% per sub-words

Table 1 Cont'd Comparison between Different Check Processing Techniques

Ref.	Check Analysis and Regions' Extraction	Digits and Courtesy Amount			Legal Amount		
		Features	Classifier	Recognition Rate	Features	Classifier	Recognition Rate
[13]	MM filter is applied to the binary image. Courtesy amount region is extracted by applying horizontal and vertical closing filters. The date and literal amount regions are extracted by constructing connected components. Using HT, the authors extracted courtesy amount by extracting the rectangle shape. Literal amount zone is extracted by finding a maximum value of the accumulator of horizontal lines.	<i>Regions' extraction only.</i>			<i>Regions' extraction only.</i>		
[25]	Ostu's image binarization algorithm was employed. Words are extracted using vertical projection. Baseline is detected using horizontal projection.	<i>Not addressed.</i>			number of descenders, ascenders, loops, dots, and sub-words.	k-NN, MLP, and fuzzy k-NN	89, 91 and 92%
[26]	<i>Digits recognition only.</i>	Log Gabor-based features with different numbers of scales, orientations and image segments.	K-NN, NM, HMM, and SVM	98.75, 98.62, 94.43, 97.21 and 98.95%	<i>Digits recognition only.</i>		
[27]	<i>Digits recognition only.</i>	Directions and length of polygonal approximation	classifier based on fuzzy logic and turning angle functions	97.18%	<i>Digits recognition only.</i>		
[29]	<i>Digits recognition only.</i>	Principle component analysis (PCA) and linear discrimination analysis (LDA)	cluster-based weighted SVM (CBWSVM)	96.5% and 95.6% (with and without rotation, respectively)	<i>Digits recognition only.</i>		
[31]	<i>Digits recognition only.</i>	The average and variance of X and Y coordinates changes in each primitive	Nearest-neighbor	94.44%	<i>Digits recognition only.</i>		
[32]	<i>Digits recognition only.</i>	Information about images' pixels and their neighbors and overall images' properties.	Particle Swarm Optimization	73.10%	<i>Digits recognition only.</i>		

CHAPTER 3

CHECK ANALYSIS AND REGIONS' EXTRACTION

3.1 Introduction

This chapter addresses the analysis of the check image and extracting its courtesy and legal amount fields. We proposed the use of image dilation, horizontal and vertical projections, and image registration to extract courtesy and legal amount regions from real bank checks of the CENPARMI database [2]. Check analysis is presented in section 2. Section 3 describes an algorithm for region's extraction. Results are discussed in section 4.

3.2 Check Analysis

We applied image binarization and size normalization. For image binarization, we used Otsu's image binarization algorithm [24]. Check image's height is experimentally normalized to 430 rows while preserving the image's aspect ratio. Figure 9 and Figure 10 show original check image and the check image after applying the preprocessing steps, respectively.

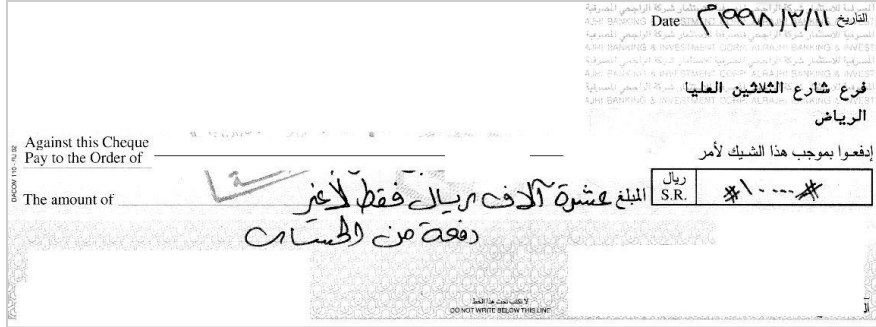


Figure 9 Original Check Image

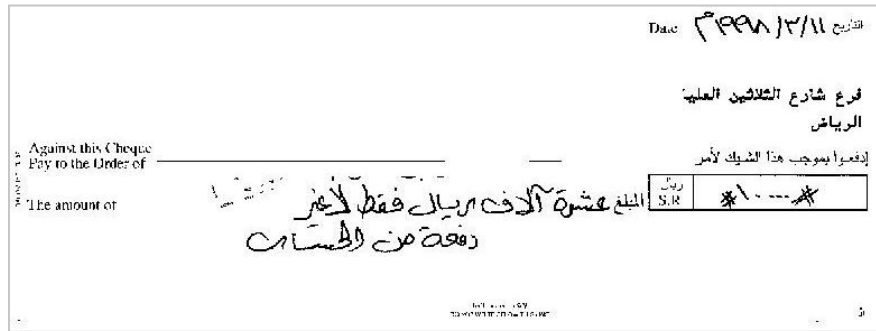


Figure 10 Check Image after Preprocessing

3.3 Region's Extraction

In the following sections we will describe the proposed techniques to extract courtesy and legal amount regions.

3.3.1 Courtesy Amount Extraction

We utilized check layout information in the sense that the courtesy amount is located on the right half of the check. In addition, since the courtesy amount is surrounded by a box, it is extracted by finding the indices for the box's boundaries. Morphological operations, used in [5], and vertical and horizontal projection histograms, used in [2 and 12], are used together to find horizontal and vertical indices. To extract horizontal indices, we created a flat linear structuring element with angle of 0° . Several lengths for the structuring

element were tested and 50 pixels produced the best result. Then image dilation is used with the created linear structuring element which produces an image with horizontal lines of length equal to or greater than the specified length. After image dilation, horizontal projection is used to extract the indices of the resulting horizontal lines. Figure 11 and Figure 12 show the right half of the check and its horizontal projection before and after dilation. Figure 13 shows the courtesy amount box after determining its horizontal indices and its vertical projection.

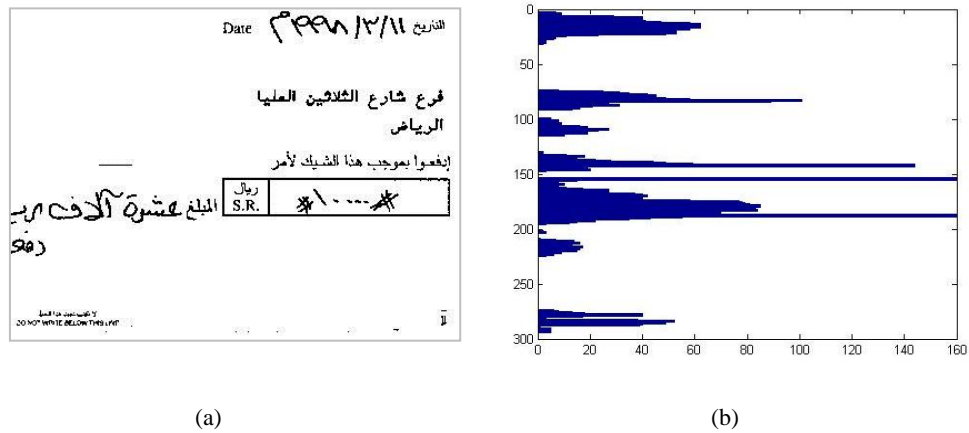


Figure 11 (a) Right Half of Check Image, (b) The Projection of (a) before Dilation

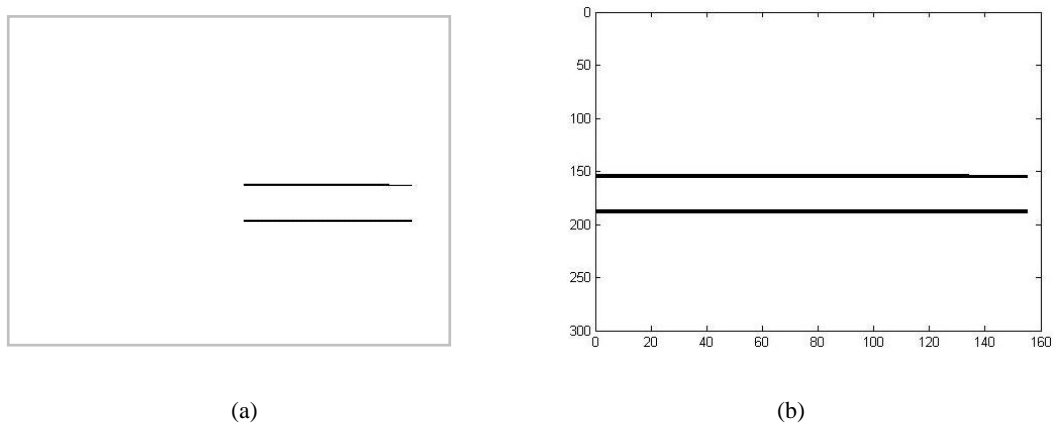
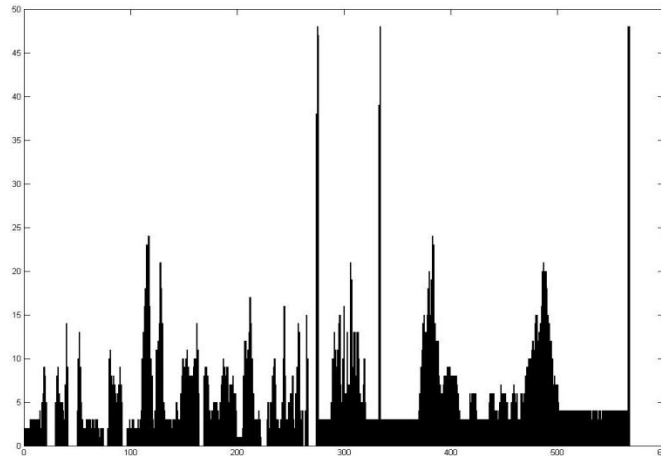


Figure 12 (a) Horizontal Image Dilation of the Right Half, (b) Horizontal Projection of the Check



(a)



(b)

Figure 13 (a) Courtesy Amount Box after Determining the Horizontal Indices, (b) Vertical Projection of (a)

In a similar manner, the vertical indices are found using image dilation and a flat linear structuring element with an angle of 90° and an experimentally specified length of 10 pixels. Then vertical projection is used to find the vertical indices. Figure 14 shows the image in Figure 13.a after dilation and its vertical projection. The extracted courtesy amount box is shown in Figure 15. Finally, horizontal and vertical projections are used to remove horizontal and vertical boundaries, respectively. Figure 16 shows the final extracted courtesy amount.



(a)

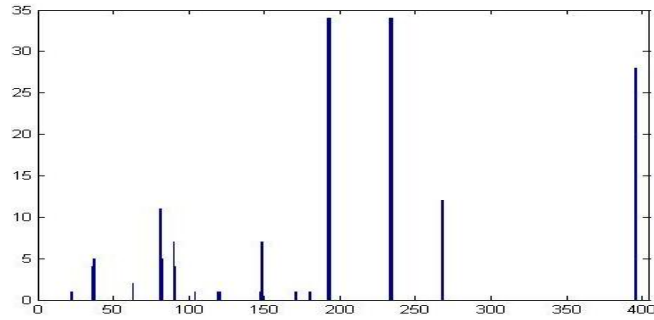


Figure 14 (a) Courtesy Amount Box after Dilation, (b) Vertical Projection of (a)

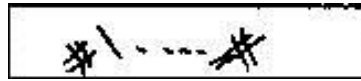


Figure 15 The Extracted Courtesy Amount Box



Figure 16 The Extracted Courtesy Amount after Removing its Boundaries

3.3.2 Legal Amount Extraction

We used the upper horizontal and the left vertical indices of the courtesy amount as the upper horizontal and right vertical indices, respectively, for legal amount. Figure 17.a shows the region that contains legal amount using the information of the courtesy amount.

The baseline is calculated using the horizontal projection and finding the maximum projection value in the upper part of the given sub image. The horizontal projection along with the calculated base line is used to remove white spaces below the legal amount.

Figure 17 shows a legal amount field before removing the noise from the bottom, its horizontal projection, and the legal amount field after removing noise.

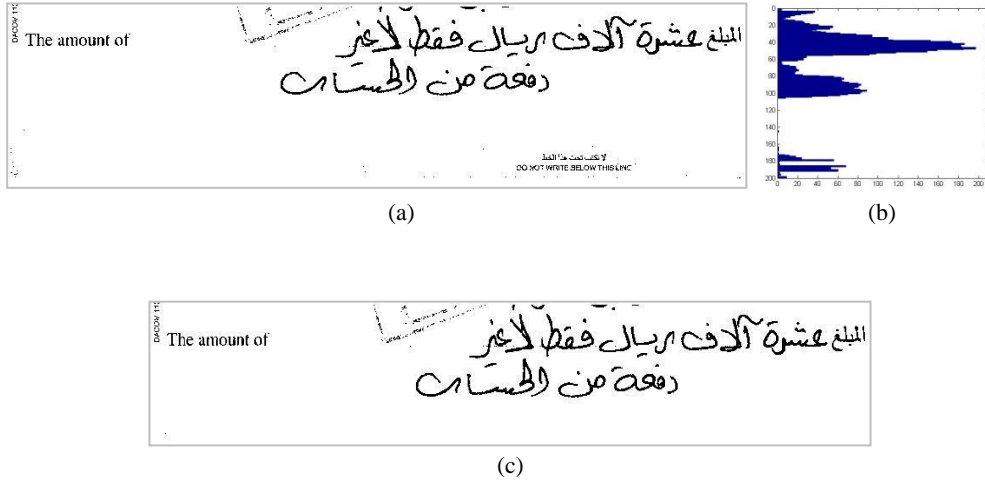
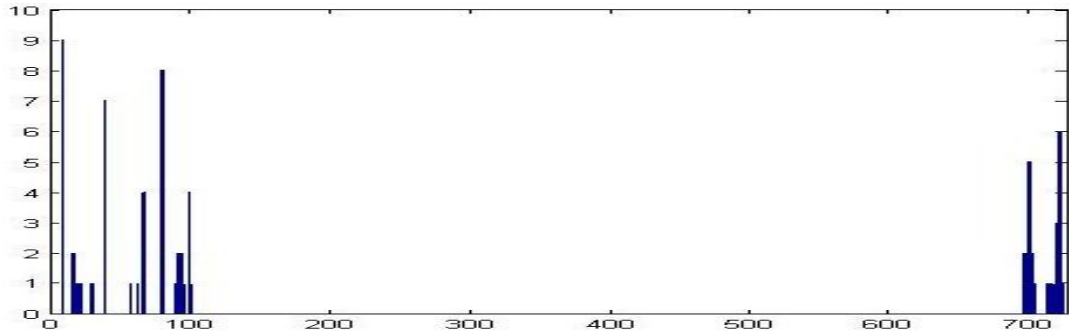


Figure 17 (a) The Legal Amount in the Check Image, (b) Horizontal Projection of (a), (c) The legal Amount after Removing the Noise

Image registration is used to find vertical indices of the handwritten part of legal amount. We used a legal amount region for a blank check, shown in Figure 18.a, as a reference image. To remove the pre-printed words, we use the blank check and extract its legal amount region. The blank legal amount image is resized so that it has the same width as the actual legal amount. Vertical indices of the blank legal amount are calculated using the vertical projection as shown in Figure 18. The obtained vertical indices are used for actual legal amount extraction. The extracted legal amount is shown in Figure 19.



(a)



(b)

Figure 18 (a) Legal Amount Region for Blank Check, (b) Vertical Projection of (a)

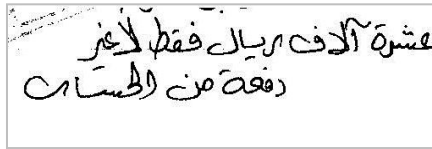


Figure 19 The Extracted Legal Amount

3.4 Results and Discussion

Our experiments are conducted using real check images from the CENPARMI database [2]. The training and testing sets have 1779 and 719 images, respectively. Figure 20 shows a sample of the testing set that is considered as bad data and is discarded since it does not have neither courtesy nor legal amount data. We evaluated the accuracy of our technique manually to assure that the required regions are correctly extracted. Our technique achieved a courtesy amount extraction rate of 100%. The obtained result shows the superiority of the used techniques. The number of correctly extracted legal amounts is

660 with an extraction rate of 91.71%. The extraction errors of the legal amount are due to unremoved printed text as shown in Figure 21 and to the overlap between printed and handwritten text and to noise as shown in Figure 22. All the experiments have been implemented using MATLAB 7.10.

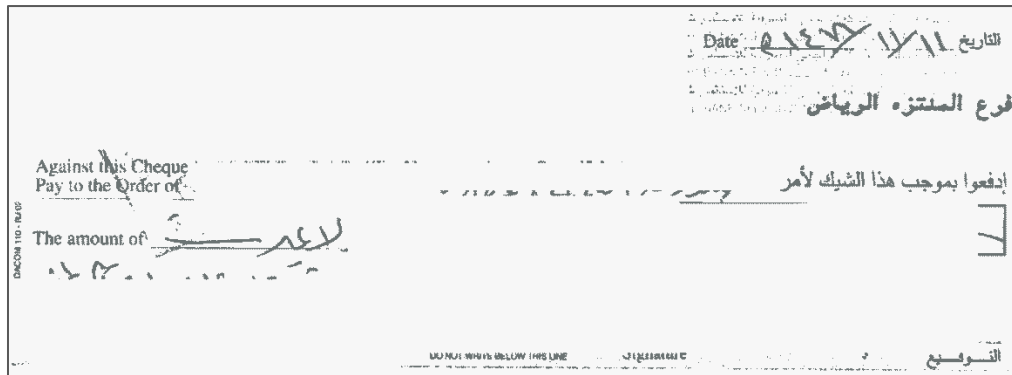
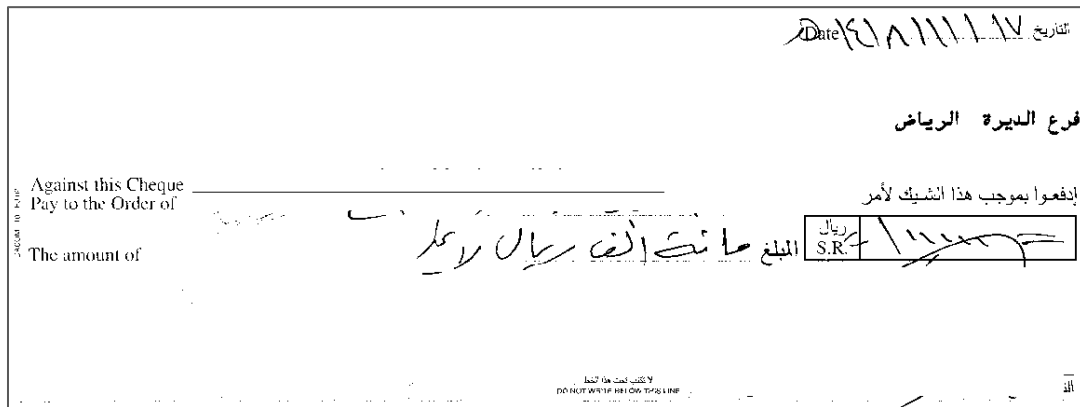


Figure 20 Bad Data



(a)

١٤٠٠٠ ريال

(b)

Figure 21 (a) Check Form, (b) Extracted Legal Amount with Unremoved Printed Text

Date ٢١٩٩/٢/١ التاريخ

فرع حي الورود الرياض

Against this Cheque
Pay to the Order of

إدفعوا بموجب هذا الشيك لأمر

The amount of ٣٠٠٠ ريال مبلغ ثلاثة آلاف ريال فقط

ريال S.R. ٣٠٠٠

٨

لا تكتب تحت هذا الخط
DO NOT WRITE BELOW THIS LINE

(a)

ثلاثة آلاف ريال فقط

(b)

Date ١٤٢٩/١/٢٩ التاريخ

فرع شارع الامير سلمان الرياض

Against this Cheque
Pay to the Order of

إدفعوا بموجب هذا الشيك لأمر

The amount of ٦١٠ ريال مبلغ مائة وأحد عشر ألف ريال فقط

ريال S.R. ٦١٠

٨

لا تكتب تحت هذا الخط
DO NOT WRITE BELOW THIS LINE

(c)

مائة وأحد عشر ألف ريال فقط

(d)

Figure 22 (a, c) Noisy Check Forms, (b, d) Extracted Legal Amounts

CHAPTER 4

HANDWRITTEN DIGITS RECOGNITION

4.1 Introduction

This chapter addresses the problem of automatic recognition of Arabic (Indian) bank check digits. Many researchers addressed the automatic recognition of handwritten digits. However, comparatively fewer ones used structural-based features or rule-based classifier. In this chapter, we are introducing a technique for automatic recognition of Arabic (Indian) bank check digits using novel structural features and a rule-based classifier. In addition, three classifiers are used (viz. Support Vector Machine (SVM), LogitBoost, and RandomForest). The classifiers output are combined using majority voting. Section 2 describes feature extraction. The used classification methods are addressed in Section 3. Experimental results are discussed in Section 4.

4.2 Feature Extraction

Automatic recognition of handwritten digits using structural-based features is not as common as using statistical-based ones. In addition, the structural-based features can model complex objects. Three main types of structural features are used; viz. segment-based, concavity, and statistics of the structural features. In addition, other features, like density and size features, are used. The feature vector consists of 27 (Segment-Based

Features), 67 (concavity features), 23 (statistics of the structural features), and 43 (other density and size features) which total 160 features. The features are described next.

4.2.1 Segment-Based Features

The average writing thickness of the digit under processing is estimated. This is done by scanning the image at different horizontal and vertical locations. The average digit's thickness is then calculated. The average thickness "THK" for all images is then used in the extraction of segment-based features. The digit's image is scanned again to extract the horizontal and vertical segments features. Segments with length less than $THK/2$ are treated as noise. Furthermore, segments with internal gap less than $THK/2$ are treated as one segment. The features under this category are characterized into digit's horizontal segments, digit's vertical segments, and double segments' distances. Ten horizontal segments, 13 vertical segments, and 4 distance features are extracted (a total of 27).

4.2.1.1 Digit's Horizontal Segments

The digit's image is scanned at $THK/3$ intervals horizontally. The number of scan lines that has one segment "HSeg1N" is estimated. The start and end horizontal locations "HSeg1L1 and HSeg1L2" of these scan lines are recorded. The three values "HSeg1N, HSeg1L1, and HSeg1L2" are normalized by the digit's height. Experimentally, we found that almost all digits have at most 3 horizontal segments which is the case of digit three as shown in Figure 23 and only limited number of these digits has four segments. Consequently, the same process is applied for the horizontal scan lines that have two and three segments. The normalized length of four segments is used as a feature. Figure 23 shows the Arabic digit three "3" which has scan lines with one and three segments. The start and end locations of these scan lines are shown. Table 3 shows the extracted features

for the shown image. Note that HSeg3N, HSeg3L1, and HSeg3L2 are used for the number of scan lines and start and end locations of horizontal scan lines with three segments after normalization. Three (lines that have 1, 2 or 3 segments) each with three (normalized number of segments and start and end locations) and one normalized number of lines with four segments, a total of 10 features, are extracted.

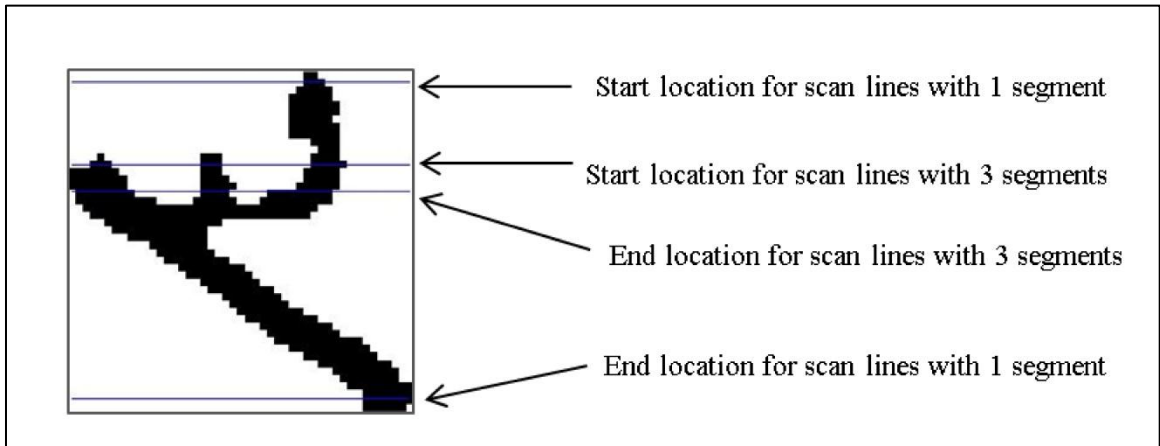


Figure 23 Location of Horizontal Scan Lines with One and Three Segments

Table 3 Sample Feature Vector Values for Digit '3'

feature	HSeg1N	HSeg1L1	HSeg1L2	HSeg3N	HSeg3L1	HSeg3L2
Value	0.875	0.04058	0.973913	0.125	0.284058	0.365217

4.2.1.2 Digit's Vertical Segments

Following a similar process, the digit's image is scanned at THK/3 intervals vertically. The number of scan lines that has one vertical segment "VSeg1N" is estimated. The start and end vertical locations "VSeg1L1 and VSeg1L2" of these scan lines are recorded. The three values "VSeg1N, VSeg1L1, and VSeg1L2" are normalized by the digit's Width. Experimentally, we found that almost all digits have at most 4 vertical segments which is the case of digit four and only limited number of these digits has five segments.

Consequently, the same process is applied for the vertical scan lines that have two, three, and four segments. The normalized length of the five segments is used as a feature. Four (lines that have 1, 2, 3, or 4 segments) each with 3 (normalized number of segments and start and end locations) and one normalized number of lines with five segments, a total of 13 features, are extracted.

4.2.1.3 Double Segments' Distances

The distance between top-most double horizontal segments is used as a feature. Similarly, the distances between bottom-most double horizontal segments, left-most and right-most double vertical segments are extracted. The number of this type of features is 4. We used top-most and bottom-most features mainly to help discriminating digit seven "٧" from digit eight "٨". The other two features are used mainly to discriminate digits two "٢" and three "٣" from other digits.

4.2.2 Concavity Features

Based on our study of the structure of Arabic digits, we defined four types of shapes, viz. convex shapes from top and right, and concave shapes from top and right. We extract the locations of the peaks for the convex ones and the holes for the concave ones. For every shape we extract the location of left and right ends, and height and width of the shape. For every digit, we extract four convex and four concave shapes. Other 3 feature values are extracted to specify the existence of special shapes. The total number of features is 4 (convex shapes) each with 8 (shape's properties) and 4 (concave shapes) each with 8 (shape's properties) and 3 (special shapes), a total of 67 features, are extracted.

4.2.2.1 Convex shape from the top

The digit's image is scanned from top to bottom. If there are two horizontal black segments with a peak in between, a convex shape from top is detected. The location of the peak and left and right ends are extracted. To extract the peak's location, the first horizontal line that has more than one segment is detected. The vertical locations for those segments are found. The peak is where the horizontal index is the minimum. The left and right ends location is at the base of the convex shape. Convex shape's width (d_1) is calculated as the distance between convex shape's left and right ends using the Euclidian distance. The height of the convex shape (d_2) is calculated as the distance between the peak and the line connecting left and right ends. Figure 24 shows digit eight (8) with the convex shape as features.

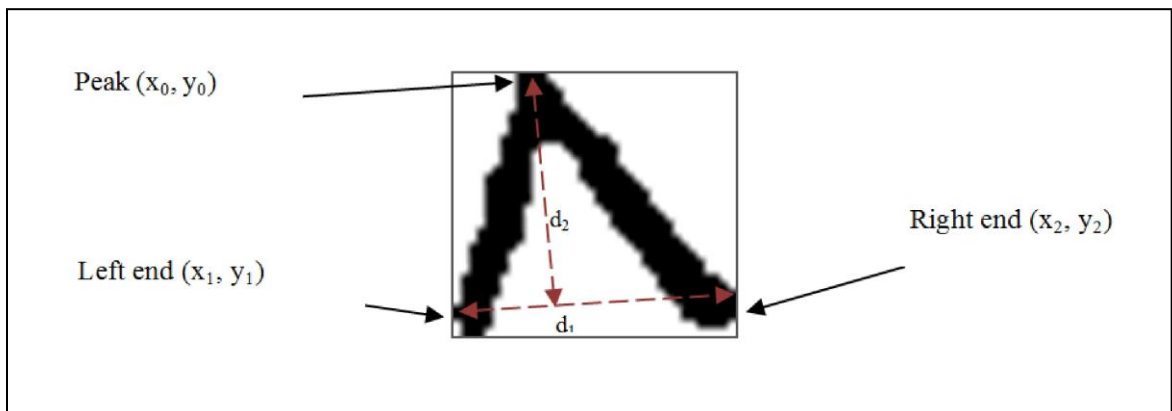


Figure 24 Convex Shape from Top

4.2.2.2 Convex shape from right

It is detected using the same procedure to find convex shape from top. To simplify processing, the image is rotated from right to left by 90 degrees and the same procedure is applied.

4.2.2.3 Concave shape from top

The digit's image is scanned from top to bottom. If there are two horizontal black segments with a hole in between a concave shape from top is detected. The location of the hole and left and right ends are extracted. To extract the hole's location, the horizontal line that has multiple segments is detected. The hole is between those horizontal segments at location where the horizontal index is the maximum. The left and right ends location is at the base of the concave shape. The concave shape's width is calculated as the distance between concave shape's left and right ends using the Euclidian distance. The height of the concave shape is calculated as the distance between the peak and the line connecting left and right ends. Figure 25 shows an example of the Arabic digit three "٣" that has two concave shapes from top.

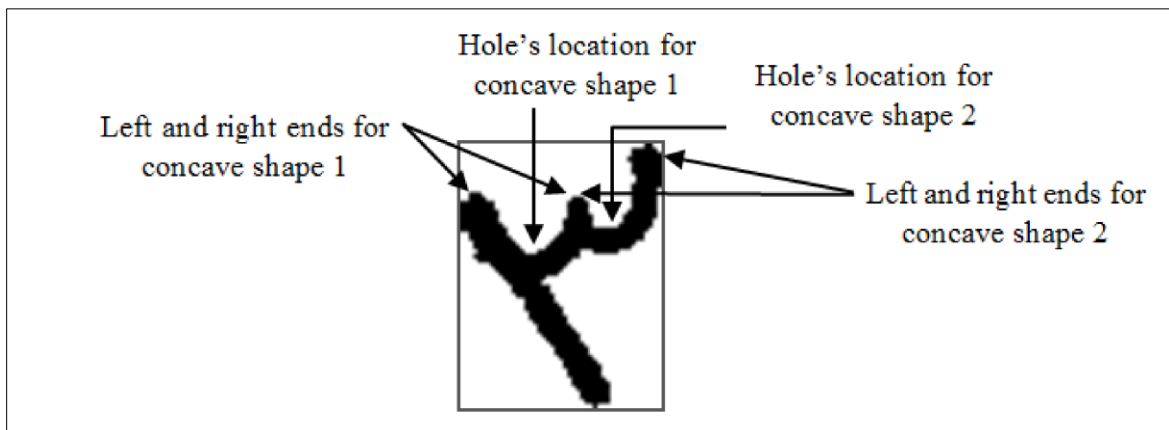


Figure 25 Concave Shape from Top

4.2.2.4 Concave shape from right

It is detected using the same procedure of the concave shape from top. To simplify processing the image is rotated from right to left by 90 degrees and then the same procedure is applied.

4.2.2.5 Special Convex and Concave shapes

Three feature values are set for special convex and concave shapes. If there exists a convex shape from top with width greater than $2*THK/3$, a feature is set to true. The first feature is mainly used to detect digits five "5", eight "8", and nine "9" as they mostly satisfies the feature's condition. Similarly, another feature is set to true if there exists a concave shape from top with width greater than both $THK/3$ and image's width normalized by 5. The feature is mainly used to detect digits three "3" and seven "7" as mostly both of them satisfy the condition. A third feature value is set to true if there exists a concave shape with width greater than image's width normalized by 5 and height greater than one third of image's length. This feature is utilized to differentiate between digits three "3" and seven "7" as both of them have concave shape from top but their concave shapes' width and height are different.

4.2.3 Statistics of Structural Features

In this feature type, if the feature is present it is set as 1 and 0 otherwise. They are used to indicate whether specific structural shapes are present or not (i.e. the processed digit's image satisfies a given condition or not). Let N_{mhs} be the number of times of getting multiple horizontal segments, N_{shs} be the number of single horizontal segments, and N_{mvs} be the number of times of getting multiple vertical segments. The arrows at top left corner of Figures (26-34) show the direction of processing the images. A total number of 23 features are extracted.

4.2.3.1 NMultiHSeg

N_{mhs} is calculated for the whole image and for the middle third of the image as shown in Figure 26. A feature is set to true if N_{mhs} in the middle third of the image is more than THK. Another feature is set to true if N_{mhs} along the whole image is greater than or equal to THK. In Figure 26, digit eight '8' has N_{mhs} in the middle third $>$ THK so both features are true. These features help discriminating digits five "o", seven "v", and eight "A" from other digits.

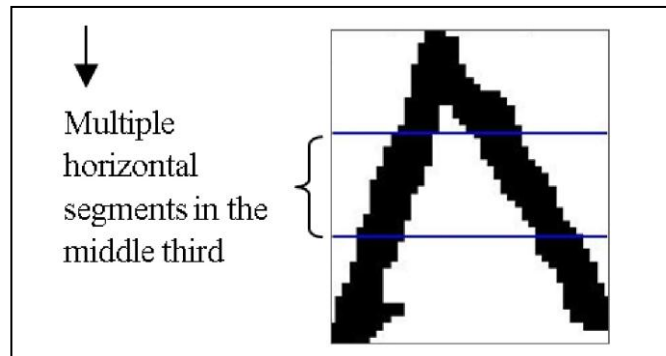


Figure 26 Digit "8" has N_{mhs} in the Middle Third $>$ THK

4.2.3.2 NSingleMultiHSeg

A feature is set to true if N_{mhs} is greater than $N_{shs}/4$ and N_{mhs} is greater than image's length normalized by $5 * THK/3$. By studying the structure of digits: zero "o", one "1", two "v", four "4", and six "6" we will find that these digits will not satisfy the feature's condition. Another feature is set to true if N_{mhs} in the bottom half of the image is greater than N_{shs} normalized by $2 * THK$. All samples of digits five "o" and eight "A" and 97% of samples of digit seven "v" satisfy the condition. Moreover, a third feature is set to true if there is no N_{mhs} in the bottom half of the image. The feature helps discriminating digits: five "o", seven "v", and eight "A" from other digits as mostly they has N_{mhs} in the bottom half.

4.2.3.3 IdxDoubleHSeg

We find the horizontal index of the last double horizontal segments with length larger than or equal to $2 * THK/3$. If the index is greater than $3/4$ of image's length, the feature is set to true. Normally, digits five "o", seven "v", and eight "^" satisfy this feature.

4.2.3.4 NLeftDistDec

The bottom half of the image is scanned horizontally and the number of times of getting decreased distance from left boundary is calculated. If that number is greater than 0, the feature is set to true. The feature is used mainly to detect digit four "4".

4.2.3.5 IdxLeastLeftDist

The image is scanned horizontally to find the least distance from left boundary to the writing. The index at which minimum distance found from left is calculated. If the index at which the least distance found is greater than image's length/3, the feature is set to true. More than 80% samples of digits five "o" and eight "^" satisfy the condition.

4.2.3.6 LeftDistIncrease

The upper half of the image is scanned horizontally at $THK/3$ for an increase in the distance from left boundary to the writing. If the distance from left increased by more than $2 * THK$, the feature is set to true. In Figure 27, two scan lines "a and b" are shown. The distances from left to the writing at each scan line are shown as vertical dashed lines at vertical indices Dist1 and Dist2. Note that the difference between Dist1 and Dist2 is greater than $2 * THK$ so the condition is satisfied. The feature is used mainly to detect digit six "6".

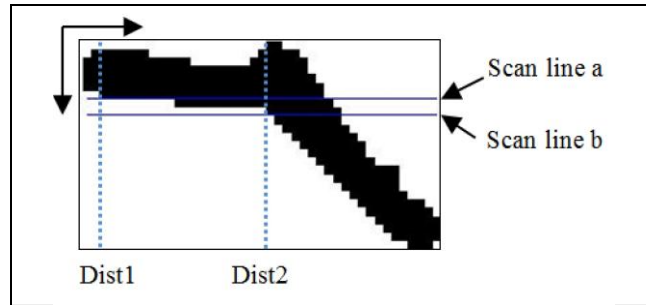


Figure 27 leftDistIncrease Feature Justification

4.2.3.7 MaxRightDist

The maximum distance from the right boundary to the nearest black pixel is calculated. The feature's condition is satisfied if the distance is greater than or equal to quarter of the image's width. Figure 28 shows the Arabic digit two "٢" that satisfies the condition.

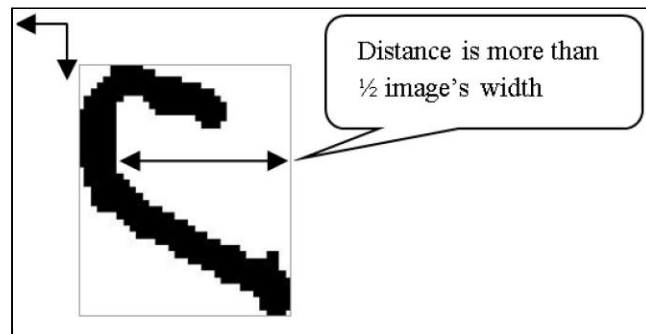


Figure 28 rightDist Feature Justification

4.2.3.8 LeftRightDist

The image is scanned horizontally and the number of times of getting decreased distance from right or left boundary is calculated. If both right and left distance decreasing counters are more than image's length normalized by THK, the feature's condition is satisfied. Normally, the Arabic digit eight "٨" satisfies the condition.

4.2.3.9 TopDist1

We are concerned with vertical lines between first and last quarter of image's width. Let v_1 and v_2 be the number of vertical lines with distance, from top, greater and less than quarter of image's length, respectively. If v_1 is less than v_2 then the feature is set to 1. The vertical lines in Figure 29 determine the boundaries for the region that we check for satisfying the condition. The horizontal one determines the index of quarter of image's length. In Figure 29 (a), it is clear that all vertical lines have distance, from top, less than quarter of image's length so the condition is satisfied. Figure 29 shows digit seven '7' that does not satisfy the condition.

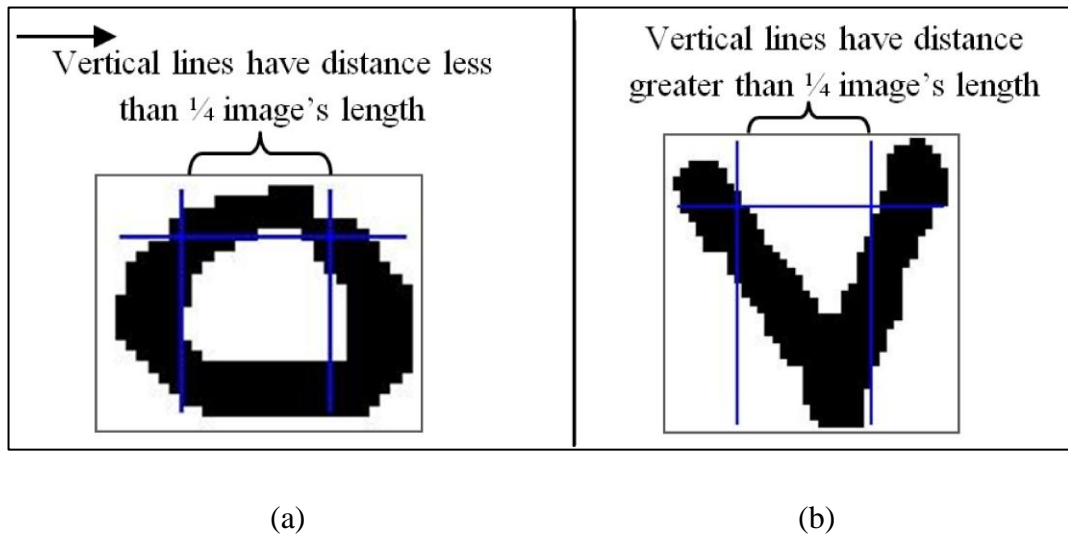


Figure 29 TopDist1 Feature Illustration: (a) Satisfied Condition, (b) not Satisfied Condition

4.2.3.10 TopDist2

Through all columns in the middle third of image's width, we find the horizontal index of the farthest black pixel from top boundary. If the index is greater than half image's length, the feature's condition is satisfied. The vertical lines in Figure 30 determine the

boundaries for the middle third of image's width whereas horizontal one determines the horizontal index of half images' length. Figure 30 illustrates this feature.

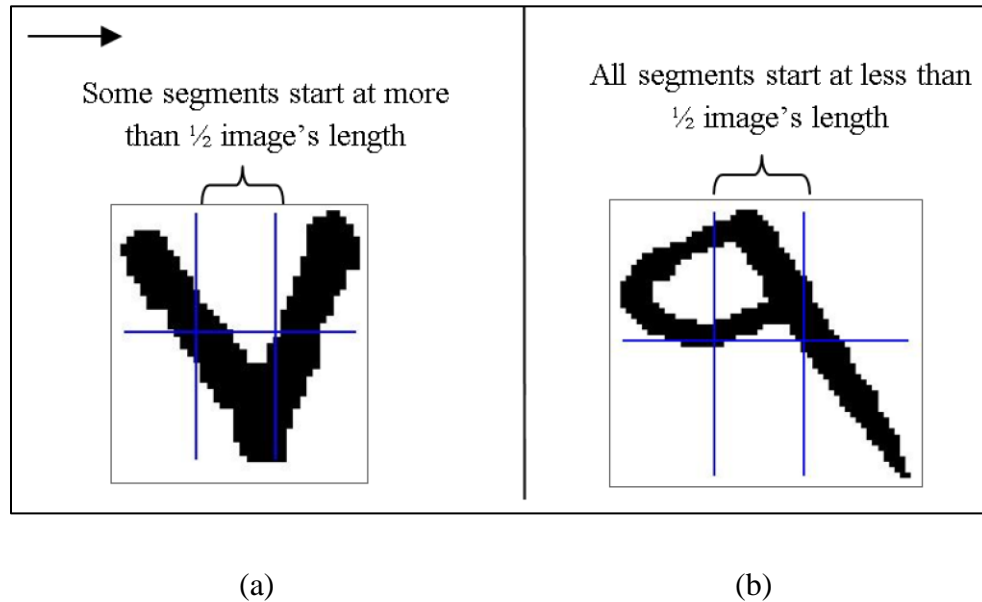


Figure 30 TopDist2 Feature Illustration: (a) Satisfied Condition, (b) not Satisfied Condition

4.2.3.11 TopDistCount

The number of vertical lines with distance, from top, greater than half of image's length is counted. The main condition that should be avoided is to have a number of lines greater than or equal to image's width normalized by $2 * THK$. The horizontal line in Figure 31 determines the index of half image's length. The vertical line determines the horizontal index where the rule is violated.

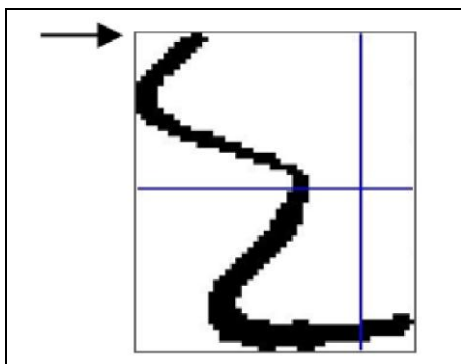
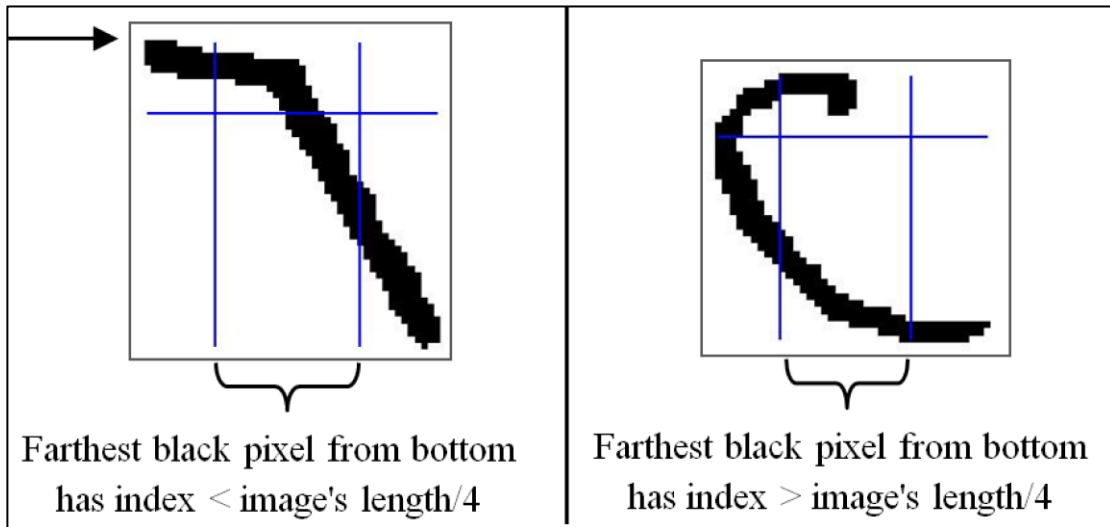


Figure 31 topDistCount Feature Illustration Example

4.2.3.12 BottomDist1

Through all columns between first and last quarter of image's width, we count the number of times of getting vertical segment with horizontal index greater than quarter of image's length $c1$ and with horizontal index less than quarter of image's length $c2$. If $c1$ is less than $c2$, the feature is set to 1. The vertical lines in Figure 32 determine the boundaries for the checked regions while the horizontal lines determine the index of quarter of image's length. It is clear that Figure 32 (a) satisfies the condition whereas Figure 32 (b) does not. This feature is used mainly to detect digit six "6".



(a)

(b)

Figure 32 bottomDist1 Feature Illustration: (a) Satisfied Condition, (b) not Satisfied Condition

4.2.3.13 BottomDist2

The second quarter of the image's width is scanned vertically. If N_{mvs} exists with distance, from bottom, greater than $(\text{image's length} - \text{THK})$, the feature is set to 1.

4.2.3.14 multiVSeg1

The feature is set to true if N_{mvs} is less than $\text{THK}/3$. It is used mainly to detect digits zero, one, six, seven, and eight.

4.2.3.15 multiVSeg2

This feature is the same as the last one but in this feature we calculate N_{mvs} regardless of the distance between the segments. If N_{mvs} is more than image's width normalized by THK , the feature is set to true. It is used mainly to detect digits two, three, four, five, and nine.

4.2.3.16 multiVSeg3

N_{mvs} between first and last quarter of image's width is calculated. If it is greater than $THK/3$, the feature's condition is satisfied. More than 80% samples of digits two, four, five, and nine are satisfy the condition.

4.2.3.17 multiVSeg4

This feature is the same as the last one but in this feature we calculate N_{mvs} regardless of the distance between the segments. If this number is more than THK , the feature is set to true.

4.2.3.18 NMVseg

In this rule we count N_{mvs} that have distance, from top, less than one third of image's length. The main condition gets satisfied when the number of those lines is greater than $2 * THK/3$.

4.2.3.19 HVSeg

We find first index at which N_{mhs} are found. Then we find a new image by eliminating the rows that have indices less than the calculated index. N_{mvs} for the new image is calculated. If N_{mvs} is more than or equal to $THK/2$; the feature's condition is satisfied. It is used mainly to detect digits five and nine.

4.2.3.20 SizeVSeg

There are two conditions to satisfy this rule. The image's length should be less than or equal to $4 * THK$ and the number of N_{mvs} in the image should be less than four. Almost all samples of digit zero satisfy this rule. Even though, digit five "o" in Figure 33 has

length less than four times of the writing thickness, there are more than four Nmvs. The feature is used mainly to differentiate between digits zero and five.

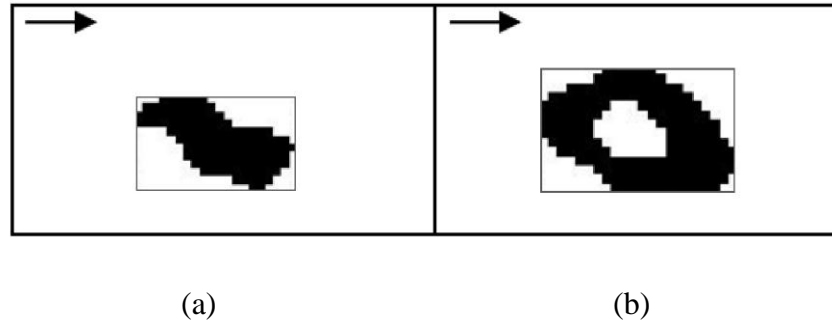


Figure 33 SizeVSeg Feature Justification: (a) Satisfied Condition, (b) not Satisfied Condition

4.2.4 Other Features

These are added features that are used to further help discriminating various classes. Total number of 43 features is used.

4.2.4.1 Size

Image's height and width are used as features. Both features are used with and without discretization. The height and width are discretized by the value of three and four writing thickness, respectively. In addition to these four features, another feature is used with condition of getting both image's length and width with value less than or equal to $4 * THK$. The last feature is used mainly to detect digit zero.

4.2.4.2 Length2WidthRatio

In this feature, two conditions should be satisfied. The image's length should be less than or equal to four times the writing thickness and the ratio of height to width should be less than 1. The feature is used mainly to detect digit zero.

4.2.4.3 LeftPeaks

We calculate horizontal and vertical indices for two peaks and a hole in the left using a change in the distance from the left side. We start by scanning the image horizontally and finding an increase in the distance from left boundary to the writing. The first left peak is found if there is an increase in the distance from left follows a decrease. In contrast, the left hole is found if there is an increase in the distance from left followed by a decrease as shown in Figure 34. After finding the first peak and a hole, we calculated the last peak by the same way. In addition to horizontal and vertical indices of first left peak and the hole, the vertical index of second left peak, and height and width of the peaks and the hole is calculated. The total number of extracted features is 15. The horizontal lines in Figure 34 determine, from top, the first left peak, the left hole, and the last peak. The arrows at the top left corner show the direction of processing the image. We scan the image from top and estimate the distance from left. The feature is mainly used to detect digit four "4".

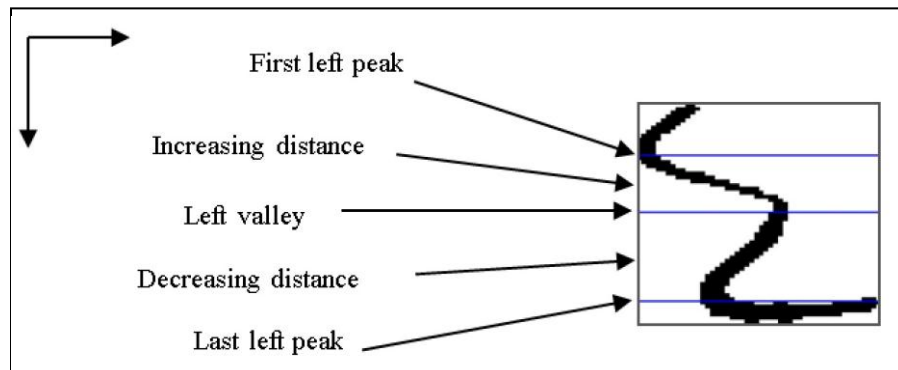


Figure 34 LeftPeaks Feature Illustration

4.2.4.4 Density

The average black pixel density is calculated. In addition to the value of density, other features related to it are calculated. The main condition that should be satisfied is to have

density greater than 0.5 and image's height less than or equal to $4 * THK$. Furthermore, each image is divided into a number of grids and the density for each segment is calculated. Experimentally, we have found that 3×4 divisions, as shown in Figure 35, give the best result. A total of 14 features are extracted from each image. The density features are already used in the literature.

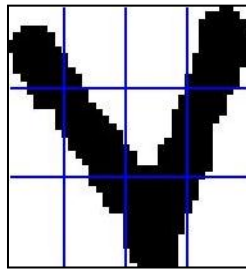


Figure 35 Digit '7' Divided into 3 x 4 Divisions

4.2.4.5 AverageDistance

The white space from left, right, top, and bottom boundary to the writing is calculated. The average distance from each direction is computed and used as feature.

4.2.4.6 Loop

Digit's image is checked whether it has a loop with area greater than $2 * THK$ or not. If a loop exists, the feature is set to 1. This feature is already used in the literature.

4.2.4.7 SlopeDifference

The slope at every point at the left between the upper leftmost and the lower rightmost points is calculated. The difference between maximum and mean slopes is used as a feature. The feature is mainly used to detect digit six "6".

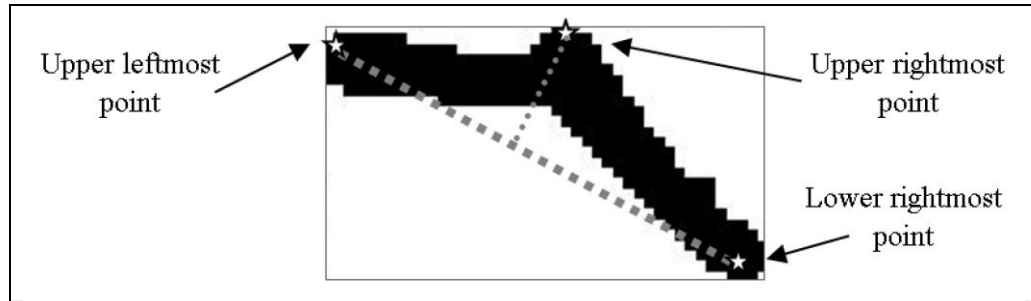


Figure 36 Illustrating Upper, Lower Rightmost and Upper Leftmost Points

4.2.4.8 maxDist

The distance from upper rightmost point to the line passing through upper and lower left most points is calculated as shown in Figure 36. The calculated distance is used as a feature. This feature help us discriminate between digits one "١" and six "٦".

4.2.4.9 hProjection

We find two maximum horizontal projections at the first and the last quarter. Both projections are tested whether they are less than $4 * THK$ or not. If both horizontal projections are less than $4 * THK$, the main condition is satisfied. More than 95% samples of digits zero, one, seven, and eight satisfy the condition.

4.3 Classification Methods

In this work, a rule-based classifier is implemented, in addition to SVM, LogitBoost, and RandomForest. The rule-based classifier is developed for Arabic digits based on the structural features. These are the steps that we followed to develop the rules for every digit:

1. Set the data as a two-class problem i.e. set the class for all samples as zero except for the current digit.

2. Find the most important features for the data generated in step #1.
3. Choose one of these features and find a threshold such that almost all samples of the digit have that threshold. The chosen feature is either the most important one or an important one that give the desired objective.
4. Study the samples that have the same threshold as the current digit and use other features to discriminate them.

As an example, Figure 37 shows a pseudo code for the rules used to recognize digit zero (0)

```

If a digit's length "DigL" is less than 4 writing thickness "THK" and (number of lines
with single horizontal segments>0.8* DigL or number of lines with single vertical
segments>0.72* digit's width "DigW") and (number of double and three vertical
segments<0.2* DigW or DigW <2.5*THK)
then the digit is zero
elseif DigL is less than 2*THK
then the digit is zero
end

```

Figure 37 Pseudo Code for the Rules of Digit Zero

The complete rules for all digits are defined in appendix I. It is worth mentioning that the rule-based classifier uses only a set of 40 feature values from the total of 160 values used by the other classifiers.

SVM was proposed by Vapnik, and is based on strong foundations from statistical learning theory [38]. A support vector machine constructs a hyperplane or set of hyperplanes in a high-dimensional space. In this work, the Sequential Minimal Optimization (SMO) algorithm [39] for training the SVM classifier is used. SMO works by breaking large quadratic programming (QP) problem into a series of smallest possible

QP sub-problems that may be solved analytically. A polynomial kernel of degree three is used as a kernel function.

LogitBoost is a boosting algorithm proposed by Friedman et. al. [40]. It uses the log-likelihood-loss and Newton optimization for fitting an additive symmetric logistic model. It is a "statistical" version of AdaBoost because it minimizes the negative binomial log-likelihood instead of the exponential loss [41].

Breiman developed Random forest (or random forests) [42] as an ensemble classifier that consists of collection of decision trees [43]. The method combines two machine learning techniques, viz. bagging and random feature selection. In bagging, a tree is independently constructed using a bootstrap sample of the training set [44]. Then, a tree is grown on a new training set using random feature selection.

4.4 Experimental Results

Many experiments are conducted using 10425 digit samples (7090 for training and 3035 for testing) of the CENPARMI Arabic checks database [2]. In addition to the rule-based classification, three statistical classifiers are used to predict the digit's class; viz. Support Vector Machine (SVM), LogitBoost, and RandomForest. Classifiers' fusion with majority voting method is used with and without rule-based classifier. Fusing statistical classifiers only achieves recognition rate of 98.95% while fusing statistical classifiers with rule-based Classifier achieves better performance of 99.08%. Both results are without rejection. In Table 4, the confusion matrix for classifiers' fusion with rule-based classifier is shown.

Table 4 The Confusion Matrix of the Tested Samples with Classifier's Fusion

Actual Category	Predicted Category										Recognition Rate %
	0	1	2	3	4	5	6	7	8	9	
0	1569	5	0	0	0	0	0	0	0	0	99.68
1	1	301	0	0	0	0	2	0	0	0	99.01
2	1	0	223	0	0	1	0	0	0	0	98.67
3	1	0	4	139	0	0	0	0	0	0	97.22
4	0	1	5	0	127	0	0	0	0	0	94.74
5	0	0	0	0	0	264	0	1	0	0	99.62
6	0	2	0	0	0	0	109	0	0	0	99.10
7	0	0	0	0	1	0	0	107	0	1	99.08
8	0	0	0	0	0	1	0	0	99	0	99.00
9	0	0	0	1	0	0	0	0	0	73	95.95
Average Recognition Rate											99.08

Table 5 shows the recognition rates of the digits with SVM, LogitBoost, RandomForest, and the rule-based classifiers. Furthermore, classifiers' fusion results are shown with and without rule-based classifier. Classifiers' fusion without rule-based classification is denoted as "Classifiers' Fusion 1" while the other type is denoted as "Classifiers' Fusion 2." LogitBoost achieves average recognition rate of 98.68% and hence it is superior to other classifiers. It is clear from the table that fusing the statistical classifiers with the rule-based one achieves best results for digits 1, 2, 4, 5, 8, and 9. In general, it achieves the best average accuracy result of 99.08%. Figure 38 illustrates the average recognition results with SVM, LogitBoost, RandomForest and the two types of classifiers' fusion.

Table 5 Recognition Rates of Svm, Logitboost, Randomforest, Rule-Based and the Two Types of Classifiers' Fusion

Digit	SVM	LogitBoost	RandomForest	Rule-Based	Classifiers' Fusion 1	Classifiers' Fusion 2
0	99.62	99.75	99.87	99.56	99.75	99.68
1	97.70	97.37	96.71	93.42	98.03	99.01
2	98.22	99.11	97.33	91.11	99.11	99.11
3	97.92	97.22	95.83	90.28	97.22	96.53
4	94.74	93.23	94.74	90.23	95.49	95.49
5	99.62	98.87	99.25	97.74	99.62	99.62
6	95.50	98.20	95.50	100.00	97.30	98.20
7	97.25	96.33	97.25	99.08	98.17	98.17
8	99.00	98.00	98.00	99.00	99.00	99.00
9	93.24	97.30	91.89	94.59	95.95	98.65
Average	98.62	98.68	98.39	97.17	98.95	99.08

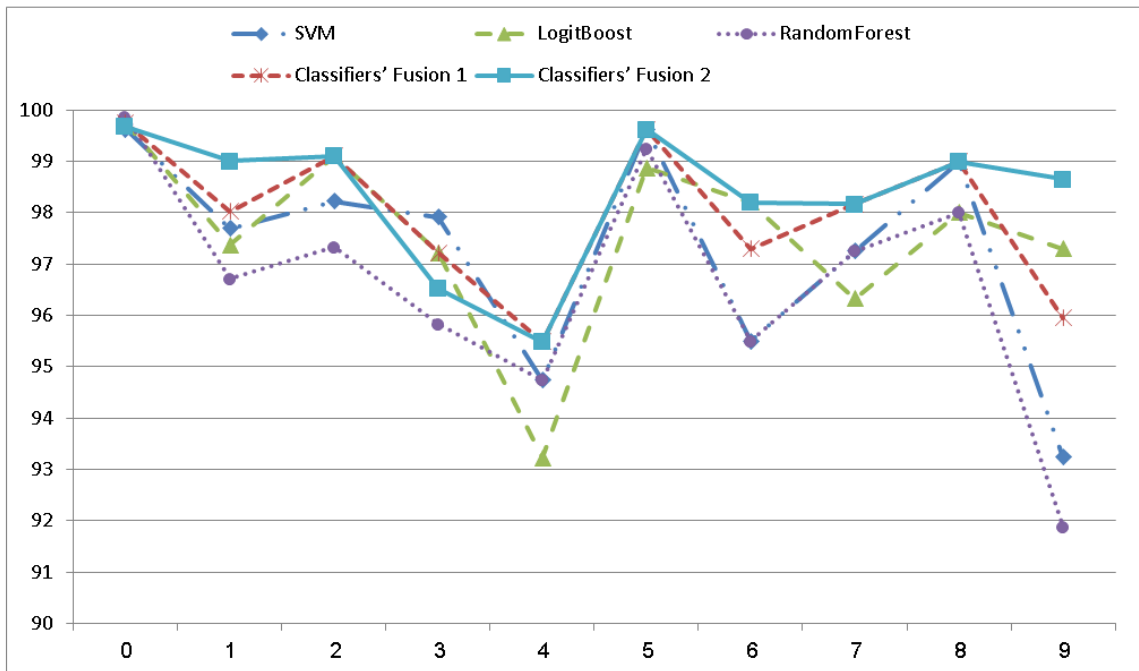


Figure 38 Comparing the Average Recognition Results with SVM, Logitboost, Randomforest and the Two Types of Classifiers' Fusion

Table 6 shows the misclassified samples with their actual and predicted classes. It is clear from the table that misclassification of some samples are due to bad data. Note samples number 6, 9, 11, 16, 19, 21, and 22. Samples 1 to 5 are for digit zero, however if these samples are presented to a human he will recognize them as 1's.

Table 6 Misclassified Samples



























No.	Image	Actual	Predicted	No.	Image	Actual	Predicted
1		0	1	15		3	2
2		0	1	16		4	2
3		0	1	17		4	2
4		0	1	18		4	1
5		0	1	19		4	2
6		1	0	20		4	2
7		1	6	21		4	2
8		1	6	22		5	7
9		2	0	23		6	1
10		2	5	24		6	1
11		3	2	25		7	9
12		3	2	26		7	4
13		3	0	27		8	5
14		3	2	28		9	3

Table 7 compares our results to the literature. It is clear from the table that our technique outperforms the published work.

Table 7 Comparing the Proposed Technique with the Literature

Ref.	Classifier	Language	Database	Recognition Rate
[26]	K-NN, NM, HMM, and SVM	Arabic	CENPARMI [2]	98.95%
[27]	classifier based on fuzzy logic and turning angle functions	Arabic	ADBase [28]	97.18%
[29]	cluster-based weighted SVM (CBWSVM)	Farsi/ Arabic	Farsi database [no reference]	96.5%
[31]	Nearest-neighbor	Farsi/ Arabic	Farsi database [no reference]	94.44%
[32]	Particle Swarm Optimization	Latin	MINST [34]	73.10%
proposed	Fusion of rule-based, SVM, LogitBoost, RandomForest	Arabic	CENPARMI [2]	99.08%

Feature extraction, rule-based classifier, and classifiers' fusion have been implemented using MATLAB 7.10. For statistical classification, we use SMO, LogitBoost, and RandomForest classifiers implemented in Weka package [45]. SVM and Decision Trees implemented in DTREG [46] are used to find the important features to build the rule-based classifier.

CHAPTER 5

COURTESY AMOUNT RECOGNITION

5.1 Introduction

This chapter describes the proposed technique to automatically recognize courtesy amounts extracted from the CENPARMI Arabic handwritten checks [2]. We tackle the problems of delimiters detection, noise removal, commas detection, and digits recognition. Figure 39 shows the followed phases in the recognition of courtesy amounts. Section 2 describes preprocessing steps. Courtesy amount delimiters detection is discussed in Section 3. Comma detection is addressed in Section 4. Courtesy amount digits recognition is presented in Section 5. Experimental results are discussed in Section 6.

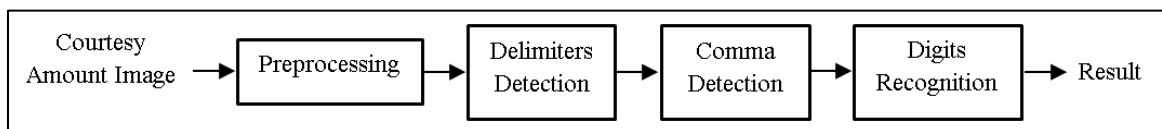


Figure 39 Courtesy Amount Recognition Phases

5.2 Preprocessing

In this phase, the courtesy amount's image is smoothed to reduce the effect of noisy pixels. A statistical based smoothing algorithm, described in [47], is used. An experimentally tested threshold of 5 is used. The algorithm enhances the courtesy amount image by eliminating very small components and filling small holes.

Furthermore, we eliminate small components on the top third part of the courtesy amount. The mean upper horizontal location "meanHUp" of all components is calculated. If there is a component located completely on top of meanHUp, it is detected as noise. The horizontal dashed line in the courtesy amount image in Figure 40 shows the mean upper horizontal location. Moreover, if a component's width is greater than the courtesy amount's width, as shown in Figure 41; it is considered as noise.

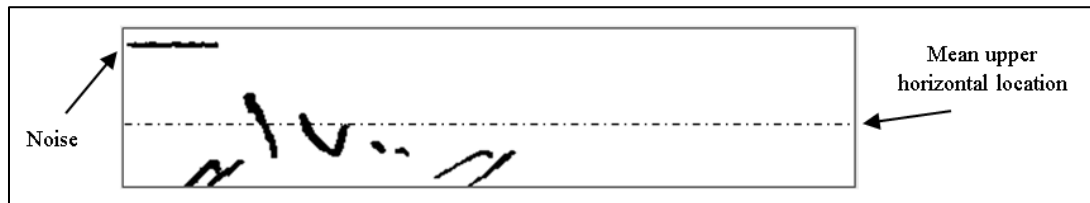


Figure 40 Mean Upper Horizontal Location

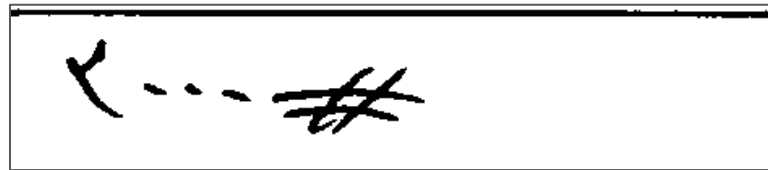


Figure 41 Courtesy Amount with Special Noisy Component

5.3 Courtesy Amount Delimiters Detection

Detecting delimiters in the courtesy amount images is an important step in the automatic recognition of courtesy amounts. The performance of the recognition of courtesy amount is highly affected by these symbols. What makes this problem so challenging is that there is no standard shape for the delimiters. In addition the delimiter could consist of one, two, or three strokes. Figure 42 shows courtesy amounts with different delimiters. A rule-based detection algorithm is developed. The used features and the rule-based classifier are presented.

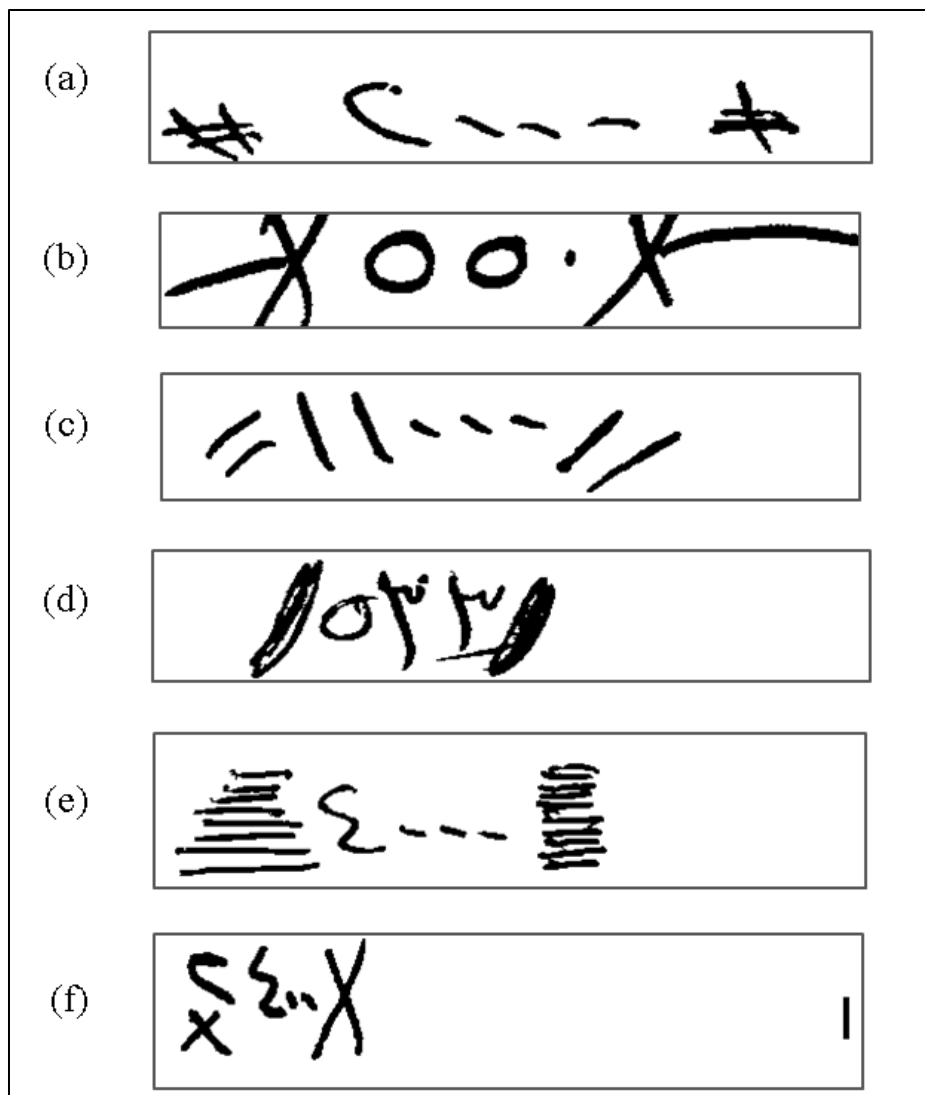


Figure 42 Courtesy Amounts with Different Delimiter Types

5.3.1 Features

The components width, length, density and segment-based features, described in section 4.2, are used to detect the delimiters. These are some additional features.

5.3.1.1 Overlap

Using components' bounding boxes, the overlap feature is set to true if the overlap between two successive components is greater than the one third of the minimum width

of the components. Figure 42 (c) shows a courtesy amount has delimiters with overlapped characters.

5.3.1.2 Inner-loop

There are some conditions that must be satisfied. The component should have a loop with area greater than two writing thickness as in Figure 43. In addition, the loop's horizontal boundaries are far away from image's boundary by one third of image's length, as shown in Figure 43. Similarly, the loop's vertical boundaries are far away from image's boundary by the one third of image's width. Furthermore, the distance from bottom image's boundary "d2" is not greater than the distance from image's top boundary "d1"

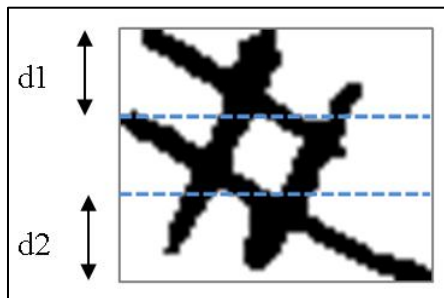


Figure 43 Inner-loop Feature Illustration

5.3.1.3 Not-one

It is set to true for samples similar to one but not one as in Figure 44.

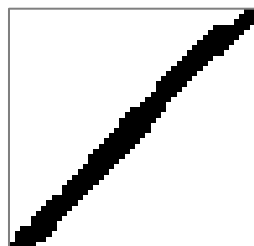


Figure 44 Not-one Feature Illustration

5.3.1.4 Terminal's Center of Gravity

The mean center of gravity for all components “mean1” except the first and last ones is calculated. The components with center of gravity less than mean1 are excluded and another mean center of gravity “mean2” is calculated. If the centers of gravity of the first and last components are less than mean2, they are determined as terminals. Figure 45 shows a sample with delimiters satisfy the given conditions.

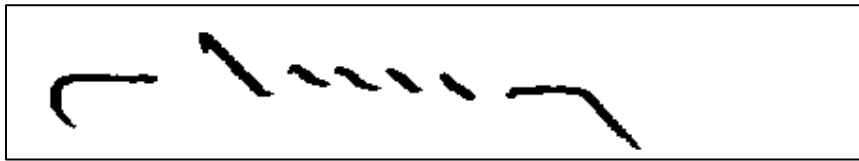


Figure 45 Terminals' Center of Gravity Illustration

5.3.2 Delimiters Detection Rule-Based Classifier

A rule-based classifier is developed for Arabic courtesy amount delimiters following the same steps described in section 4.3. The rules are used for at most three characters from left and right. Figure 46 shows a pseudo code for the delimiters' detection rules.

```

If there is an overlap, or
    a component's width "compW" is greater than five times component's length "compL", or
    compW normalized by compL is greater than 4 and compL is greater than 4 writing thickness "Th", or
    the index of last three horizontal segments is greater than  $0.8 * \text{compL}$ , or
    number of horizontal lines with four segment is greater than  $0.1 * \text{compL}$ , or
    (number of vertical lines with single vertical segment is less than  $0.8 * \text{compW}$ , and there is at least
    two vertical lines have four segments), or
    there is at least one vertical lines have five segments, or
    there is inner-loop, or
    (number of horizontal lines with single segment is greater than  $0.9 * \text{compL}$ , and number of vertical
    lines with single segment is greater than  $0.9 * \text{compW}$ , and not one, and compL is greater than  $4 * \text{Th}$ ),
or
    (compL is greater than  $4 * \text{Th}$  and there is no black pixels in the left top corner division and the one
    below it in  $3 * 4$  grid), or
    (non-zero index of first double, three or four vertical segments is less than  $0.05 * \text{compW}$  and number of
    vertical lines with one segment is less than  $0.3 * \text{compW}$ , and compL is greater than  $4 * \text{Th}$ ), or
    (number of horizontal lines with single segment is less than  $0.5 * \text{compL}$ , and non-zero index of first
    double horizontal segments is less than  $0.05 * \text{compL}$ , and the index of last double horizontal segments
    is greater than  $0.5 * \text{compL}$ , and there is black pixels in the left bottom corner division), or
    terminals' center of gravity is satisfied
then the input is terminal
end

```

Figure 46 Pseudo Code for the Rules of Delimiters

5.4 Comma Detection

There are two types of commas, a decimal comma or a separator. If the location of a comma is exactly before the last one or two digits on the right, the comma is considered as decimal comma. This problem is so challenging since some commas is written as dot which can be classified as zero. In contrast with mean upper horizontal location shown in Figure 40, we used mean lower horizontal location "meanLower" to detect commas as

shown in Figure 47. If the horizontal location of a character's center of gravity or upper horizontal location is less than the meanLower, it is considered as comma.

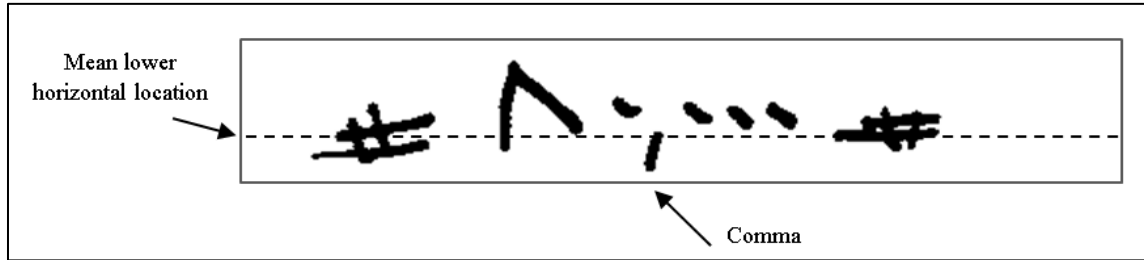


Figure 47 Mean Lower Horizontal Location

5.5 Courtesy Amount Digits Recognition

After eliminating noise, delimiters, and commas, we used the rule-based classifier described in section 4.3 to recognize the digits. If the rule-based classifier rejects a component, i.e. does not recognize it as digit, in the middle; it is considered a noise. Similarly, if the rejected component is at left or right-most locations, it is considered as delimiter because some delimiters are not detected in the delimiter detection phase.

5.6 Experimental Results

Many experiments are conducted using 2139 courtesy amount samples (1513 for training and 626 for testing) of the CENPARMI Arabic checks database [2]. After preprocessing steps, the rule-based classifier is used to detect courtesy amount's delimiters. Average accuracy of 89.77% is obtained for the tested samples. Table 8 shows the delimiters' confusion matrix of the tested samples.

Table 8 Delimiters Confusion Matrix of the Tested Samples

	Delimiter	Non delimiter	Accuracy
Delimiter	1044	393	72.65
Non delimiter	57	2906	98.08
Average			89.77

After eliminating the delimiters, the courtesy amount is checked whether it has comma or not. The location of the comma is noted so that we can decide whether it is a decimal comma or a separator. Average accuracy of 86.58% is obtained for the tested samples. In Table 9, the confusion matrix of the courtesy amount delimiters is shown.

Table 9 Comma or digit Confusion Matrix of the Tested Samples

	Comma	Digit	Accuracy
Comma	71	10	87.65
Digit	74	471	86.42
Average			86.58

The rule-based classifier described in section 4.3 is used to recognize the courtesy amount digits. An average accuracy of 91% is obtained for the tested samples. A courtesy amount is rejected if at least one digit is rejected or zero value is found. A rejection rate of 7.3% is found. The confusion matrix for the rule-based classifier for the extracted digits is shown in Table 10.

Table 10 Confusion Matrix of the Extracted Digits

Actual Category	Predicted Category										Recognition Rate %	
	0	1	2	3	4	5	6	7	8	9		Reject %
0	1275	4	0	1	0	1	0	0	0	0	6.36	93.20
1	3	231	0	0	1	0	4	0	0	0	9.81	87.17
2	1	1	168	1	2	2	0	0	0	0	9.33	87.05
3	0	1	4	111	0	0	0	1	1	0	7.81	86.72
4	0	1	4	0	85	0	0	0	0	0	22.41	73.28
5	4	1	0	0	0	218	0	1	0	0	3.45	93.97
6	0	1	0	0	0	0	94	0	0	0	3.06	95.92
7	1	0	0	0	0	0	0	92	0	0	5.10	93.88
8	0	1	0	0	0	1	0	0	81	0	3.49	94.19
9	1	0	0	0	0	0	1	0	0	51	11.67	85.00
Average Rate											7.3	91.0

The overall accuracy result for tested courtesy amount samples is 58.15% with 10.38% rejection rate. The misclassification is due to noisy samples, undetected delimiter or comma, or misclassified digits. All the experiments have been implemented using MATLAB 7.10.

CHAPTER 6

CONCLUSION AND FUTURE DIRECTIONS

6.1 Conclusion

In this thesis, we have conducted research on automatic processing of Arabic handwritten checks. Three main problems are addressed, viz. automatic extraction of courtesy and legal amounts and Arabic handwritten digits and courtesy amount recognition. Real check images, digits, and courtesy amounts from CENPARMI Arabic check database is used.

We have presented a technique for automatic extraction of courtesy and legal amount fields from check images. The process starts by image binarization and size normalization. Check layout information is utilized in extracting the regions. Image dilation and horizontal and vertical projections are used to extract courtesy amount region. To extract legal amount region, we utilized the information of the extracted courtesy amount region. Image registration and vertical projection are used to remove printed words from legal amount region. The obtained result of extracting courtesy amount field of 100% extraction rate shows the superiority of the used technique. Legal amount extraction rate of 91.71% is achieved.

The main contribution of this thesis is the proposed rule-based classifier and structural and statistical features. Structural features are extracted based on concavity shapes and

segments. Concavity features are extracted from two types of shapes; viz. convex and concave shapes from top and right directions. Total of four convex and four concave shapes are extracted from the two directions. For every shape, we extract the location of the peak or the hole, location of left and right ends, and height and width of the shape. The total number of concavity features is 67 features. The features under segments-based category are characterized into digit's horizontal segments, digit's vertical segments, and double segments distances. The total number of 27 segments' based features is extracted. A total of 23 features are extracted as statistics of structural features. Other features including size and density features, totaling 43 features, are used. The feature vector has total number of 160 features. SVM, LogitBoost, and RandomForest Classifiers are used. Classifiers' outputs are fused using majority voting. Fusing statistical classifiers with rule-based one achieves an average recognition rate of 99.08%. The achieved recognition rate outperforms the published work using the same database. The experimental results indicate the effectiveness of the extracted features and the rule-based classifier for recognizing Arabic (Indian) digits.

A technique for automatic recognition of courtesy amounts is presented. To reduce the effect of noisy pixels, courtesy amount's image is smoothed using a statistical based smoothing algorithm. One of the challenging tasks in courtesy amount recognition is the presence of delimiters that does not have standard shape and number of strokes. The components width, length, density and segment-based features are used to detect the delimiters. Additional features are extracted, viz. overlap, inner-loop, not-one, and terminal's center of gravity. A rule-based classifier is developed for Arabic courtesy amount delimiters. Existence of some commas as dots is another challenging task in

courtesy amount recognition because it can be classified as zero. Mean lower horizontal location for all courtesy amount components is used to detect commas. The rule-based classifier is used to recognize the digits after eliminating noise, delimiters, and commas. We obtained 89.77% and 86.58% for detecting delimiters and comma respectively. A recognition rate of 91.00% is obtained for digits recognition. The proposed technique has demonstrated promising performance. The proposed technique can be further extended and enhanced. To the best of our knowledge, our work is the first to tackle this problem using the given database.

6.2 Future Directions

These are some future research directions in Arabic handwritten check recognition:

1. Address the problem of courtesy amount recognition by improving the rule-based classifier for delimiters and comma detection and reducing the effect of noise.
2. Address the problem of legal amount recognition using the developed structural features and an adopted rule-based classifier.
3. Address the problem of date recognition.
4. Address the problem of signature verification.

With this discussion, we conclude this thesis. All praise is due to Allah by Whose aid are all good deeds completed. We ask Him to accept our good deeds and forgive our misdeeds.

References

- [1] R. Jayadevan, S. R. Kolhe, P. M. Patil, and U. Pal, "Automatic processing of handwritten bank cheque images: a survey," *International Journal on Document Analysis and Recognition (IJDAR)*, Jul. 2011.
- [2] Y. Al-Ohali, M. Cheriet, and C. Suen, "Databases for recognition of handwritten Arabic cheques," *Pattern Recognition*, vol. 36, no. 1, pp. 111–121, Jan. 2003.
- [3] D. Wang, "IFCRS: An Information Fusion Based Check Recognition System," in *2009 WRI Global Congress on Intelligent Systems*, 2009, pp. 243–247.
- [4] M. Shridhar, G. F. Houle, and F. Kimura, "Document Recognition Strategies for Bank Cheques," *IEEE International Conference on Electro/Information Technology*, pp. 170–173, 2009.
- [5] R. Palacios and A. Gupta, "A system for processing handwritten bank checks automatically," *Image and Vision Computing*, vol. 26, no. 10, pp. 1297–1313, Oct. 2008.
- [6] G. Kaufmann and H. Bunke, "Automated Reading of Cheque Amounts," *Pattern Analysis & Applications*, vol. 3, no. 2, pp. 132–141, Jun. 2000.
- [7] N. Gorski, V. Anisimov, E. Augustin, O. Baret, D. Price, and J. Simon, "A2iA Check Reader : A Family of Bank Check Recognition Systems," in *The 5th IEEE International Conference on Document Analysis and Recognition. ICDAR'99*, 1999, pp. 1–4.
- [8] L. Chandra, N. Software, T. Limited, S. V. Marg, and Q. I. Area, "Automatic Courtesy Amount Recognition for Indian Banks ' Checks," in *IEEE Region 10 Conference, 2008*, 2008, pp. 6–10.
- [9] M. T. El-melegy and A. A. Abdelbaset, "Global features for offline recognition of handwritten Arabic literal amounts," in *Information and Communications Technology*, 2007, pp. 125–129.
- [10] L. Souici-meslati and M. Sellami, "A hybrid approach for Arabic literal amounts recognition," *Arabian Journal for Science and Engineering*, vol. 29, no. 2, pp. 177–194, 2004.
- [11] M. Cheriet, N. E. Ayat, and C. Y. Suen, "Arabic cheque processing system-issues and future trends," *Chaudhuri, B.B. (ed.) Digital Document Processing*, pp. 213–234, 2007.

- [12] I. Ahmad and S. A. Mahmoud, "Arabic Bank Check Analysis and Zone Extraction," in *Image Analysis and Recognition*, 2012, pp. 141–148.
- [13] F. B. Samoud, S. S. Maddouri, H. E. L. Abed, N. Ellouze, and T. E. Fadoua, "Comparison of two handwritten Arabic zones extraction methods of complex documents," in *Int'l Arab Conf. on Information Technology*, 2008, pp. 1–7.
- [14] Y. Li, Y. Zheng, D. Doermann, and S. Jaeger, "Script-independent text line segmentation in freestyle handwritten documents.," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 8, pp. 1313–1329, Aug. 2008.
- [15] J.-M. Geusebroek, A. W. M. Smeulders, and J. van de Weijer, "Fast anisotropic Gauss filtering.," *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, vol. 12, no. 8, pp. 938–43, Jan. 2003.
- [16] S. Osher, R. Fedkiw, and K. Piechor, *Level Set Methods and Dynamic Implicit Surfaces*, vol. 153, Springer, 2004, p. B15.
- [17] R. Al-Hajj Mohamad, L. Likforman-Sulem, and C. Mokbel, "Combining slanted-frame classifiers for improved HMM-based Arabic handwriting recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 7, pp. 1165–77, Jul. 2009.
- [18] M. Pechwitz, S. S. Maddouri, and V. Märgner, "IFN/ENIT-Database of Handwritten Arabic Words," in *Colloque Int'l Francophone sur l'Ecrit et le Document*, 2002, pp. 1–8.
- [19] M. Pechwitz and V. Maergner, "HMM Based Approach for Handwritten Arabic Word Recognition Using the IFN / ENIT - Database," in *Int'l Conf. Document Analysis and Recognition (ICDAR)*, 2003, pp. 890–894.
- [20] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, vol. 7, no. 4. Wiley, 1973, p. 482.
- [21] A. Elbaati, H. Boubaker, M. Kherallah, A. Ennaji, H. El Abed, and A. M. Alimi, "Arabic Handwriting Recognition Using Restored Stroke Chronology," in *10th International Conference on Document Analysis and Recognition*, 2009, pp. 411–415.
- [22] A. Elbaat, M. Kherallah, A. Alimi, and A. Ennaji, "Restoration of the temporal order of the off-line handwriting using genetic algorithm," in *4th International Conference on Computer Science Practice in Arabic (CSPA)*, 2008.
- [23] S. Young, "The HTK hidden Markov model toolkit: Design and philosophy," *Department of Engineering, Cambridge University, Tech. Rep. CUED/FINFENG/TR152*, 1994.

- [24] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Trans On Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
- [25] N. Farah, L. Souici, and M. Sellami, "Classifiers combination and syntax analysis for Arabic literal amount recognition," *Engineering Applications of Artificial Intelligence*, vol. 19, no. 1, pp. 29–39, Feb. 2006.
- [26] S. A. Mahmoud and W. G. Al-Khatib, "Recognition of Arabic (Indian) bank check digits using log-gabor filters," *Applied Intelligence*, vol. 35, no. 3, pp. 445–456, May 2010.
- [27] M. T. Parvez and S. A. Mahmoud, "Arabic Handwritten Alphanumeric Character Recognition using Fuzzy Attributed Turning Functions," in *the 1st International Workshop on Frontiers in Arabic Handwriting Recognition*, 2010, pp. 9–14.
- [28] E. El-Sherif and S. Abdelazeem, "A two-stage system for Arabic handwritten digit recognition tested on a new large database," in *International Conference on Artificial Intelligence and Pattern Recognition (AIPR-07)*, 2007, pp. 237–242.
- [29] M. Salehpour and A. Behrad, "Cluster based weighted SVM for the recognition of Farsi handwritten digits," in *10th Symposium on Neural Network Applications in Electrical Engineering*, 2010, pp. 219–223.
- [30] M. Nahvi, M. Rafeei, R. Ebrahimpour, and E. Kabir, "Combination of two-class classifiers for the recognition of Farsi handwritten digits," *16th Iranian Conference on Electrical Engineering ICEE2008*, pp. 203–207, 2008.
- [31] S. Mozaffari, K. Faez, and M. Ziaratban, "Structural Decomposition and Statistical Description of Farsi / Arabic," in *8th International Conference on Document Analysis and Recognition (ICDAR'05)*, 2005, pp. 237–241.
- [32] N. O. S. Ba-Karait and S. M. Shamsuddin, "Handwritten Digits Recognition Using Particle Swarm Optimization," *2008 Second Asia International Conference on Modelling & Simulation (AMS)*, pp. 615–619, May 2008.
- [33] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.
- [34] Y. LeCun and C. Cortes, "The MNIST database." [Online]. Available: <http://yann.lecun.com/exdb/mnist/index.html>.
- [35] R. P. W. Duin, "The Combining Classifier : to Train or Not to Train ?," in *the 16th Int'l Conf on Pattern Recognition*, 2002, vol. 2, no. c, pp. 765–770.

- [36] S. Ruggieri, “Efficient C4.5,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp. 438–444, 1993.
- [37] S. Al-Ma’adeed, D. Elliman, and C. Higgins, “A Data Base for Arabic Handwritten Text Recognition Research,” *The International Arab Journal of Information Technology*, vol. 1, no. 1, pp. 117–121, 2004.
- [38] V. Vapnik, *Statistical Learning Theory. Adaptive and learning systems for signal processing, communications, and control*. Simon Haykin, 1998.
- [39] J. C. Platt, “Fast Training of Support Vector Machines using Sequential Minimal Optimization,” in *Advances in Kernel Methods—Support Vector Learning*, B. Schölkopf, J. Burges, and A. J. Smola, Eds. MIT Press, 1998, p. 185—208.
- [40] B. J. Friedman, T. Hastie, and R. Tibshirani, “Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors),” *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.
- [41] R. W. Lutz, “LogitBoost with Trees Applied to the WCCI 2006 Performance Prediction Challenge Datasets,” *International Joint Conference on Neural Networks*, pp. 1657–1660, 2006.
- [42] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [43] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*. Chapman & Hall/CRC, 1984.
- [44] L. E. O. Bbeiman, “Bagging Predictors,” *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [45] M. Hall, H. National, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The WEKA Data Mining Software: An Update,” *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10–18, 2009.
- [46] P. H. Sherrod, “DTREG predictive modeling software.” [Online]. Available: <http://www.dtreg.com>.
- [47] S. Mahmoud, “Arabic Character Recognition using Fourier Descriptors and Character Contour Encoding,” *Pattern Recognition*, vol. 27, no. 6, pp. 815–824, 1994.

Appendix 1: Rule-Based Classifier

Definitions:

- L: the digit's length
- W: the digit's width
- THK: the average writing thickness
- rightConcave, topConcave, rightConvex, and topConvex are used to refer to concave shapes from right and left and convex shapes from right and left respectively.
- slopeDifference: The slope at every point at left is calculated. The difference between maximum and mean slopes is used as a feature
- maxDist: The distance from upper rightmost point to the line passing through upper and lower left most points
- leftDistIncrease: If the distance from left increased by more than $2*Th$, the feature is set to true
- leftPeaks: convex shapes from left
- density: the average black pixel density
- loop: circle
- bottomDist2: the second quarter of the image's width is scanned vertically. If N_{mvs} exists with distance, from bottom, greater than $(\text{image's length} - THK)$,

Rules:

Here are set of rules for recognizing Arabic (Indian) digits.

Zero ('0'):

if $L < 4*THK$, **and**

(number of horizontal lines with single segment $> 0.8*L$ **or** number of vertical lines with single segment $> 0.72*W$), **and**

(number of vertical lines with double and three segments $< 0.2*W$ **or** $W < 2.5*THK$)

then the digit is zero

elseif $L < 2*THK$

then the digit is zero

end

One ('1'):

if $L > 4 * THK$, **and**

number of horizontal lines with single segment $> 0.8 * L$, **and**

no horizontal line has three segments, **and**

Not (density > 0.35 & $L/W < 2$), **and**

Not (slopeDifference > 1 & maxDist > 5), **and**

(number of vertical lines with double and three segments $< 0.2 * W$ or $W < 3 * THK$), **and**

topConcave exists with height $< 0.5 * THK$, **and**

No double rightConcave

then the digit is one

elseif $L > 4 * THK$, **and**

number of horizontal lines with single segment $= L$, **and**

If rightConcave exists its height $< THK$, **and**

Not (two rightConcave with height $> 0.5 * THK$ and width $> THK$), **and**

Not (slopeDifference > 1 & maxDist > 5)

then the digit is one

end

Two ('2'):

if $L > 2.5 * THK$, **and**

$W > 2.5 * THK$, **and**

Not (topConcave with horizontal index $< 0.7 * L$ & height $> 0.5 * THK$ & width $> THK$), **and**

Not (two rightConcave with height $> 0.5 * THK$ & width $> THK$), **and**

(rightConcave with horizontal index $< 0.7 * L$ & height $> THK$ & width $> 2 * THK$), **and**

no loop, **and**

Not (slopeDifference > 1 & maxDist > 5), **and**

Not (rightConvex with horizontal index $> 0.4 * L$ & width $> 2 * THK$), **and**

Not (number of horizontal lines with single segment $\geq 0.7 * L$ & number of vertical lines with single segment $\geq 0.8 * W$ & maxDist ≥ 8)

then the digit is two

end

Three ('r'):

if $L > 4 * THK$, **and**

number of horizontal lines with single segment $> 0.4 * L$, **and**

at least one horizontal line has more than one segment, **and**

the index of last double or three horizontal segments $> 0.2 * W$, **and**

no loop, **and**

Not (slopeDifference > 1 & maxDist > 5 & rightConcave with height $< 2 * THK$), **and**

at least one rightConcave, **and**

topConcave with horizontal index $< 0.5 * L$ & height $> 0.5 * THK$ & width $> THK$, **and**

If topConvex exists, its right end has horizontal index $< 0.7 * L$, **and**

Not (two rightConcave with height $> 0.5 * THK$ & width $> THK$)

then the digit is three

elseif topConcave with horizontal index $< 0.5 * L$ & height $> 0.5 * THK$ & width $> THK$, **and**

Not (two rightConcave with height $> 0.5 * THK$ & width $> THK$), **and**

rightConcave with height $> THK$, **and**

maxDist < 7 , **and**

no vertical line has four segments

then the digit is three

end

Four ('t'):

if $L > 4 * THK$, **and**

$W > 3 * THK$, **and**

number of horizontal lines with single segment $> 0.7 * L$, **and**

Not (topConcave with horizontal index $< 0.7 * L$ & height $> THK$ & width $> THK$), **and**

Not (number of vertical lines with single segment $\geq 0.9 * W$ & rightConcave with height $< THK$ & maxDist ≥ 8), **and**

(at least a vertical line has five segments **or** two rightConcave with height $> 0.5 * THK$ **or**

rightConcave with vertical index $> 0.7 * L$ & height $> 0.5 * THK$ **or** rightConcave with height $> 0.5 *$

THK & rightConvex with vertical index $>$ rightConcave's one)

then the digit is four

end

Five ('5'):

if $L > 2 * THK$, **and**

$W > 2 * THK$, **and**

number of horizontal lines with single segment $< 0.8 * L$, **and**

number of horizontal lines with double segments $> 0.2 * L$, **and**

number of vertical lines with single segments $< 0.7 * W$, **and**

(bottomDist2=1 **or** (loop=1 & number of horizontal lines with single segment $< 0.5 * L$ & horizontal index of last double horizontal segments $> 0.7 * L$)), **and**

Not (double rightConcave & two leftPeaks), **and**

Not (rightConcave with height $> 2 * THK$ and width $> 2 * THK$), **and**

Not (topConcave with horizontal index $< 0.5 * L$ & height $> THK$ & width $> THK$), **and**

Not (two rightConcave with height $> 0.5 * THK$)

then the digit is five

end

Six ('6'):

if $L > 4 * THK$, **and**

number of horizontal lines with single segment $> 0.8 * L$, **and**

number of vertical lines with double vertical segments $< 0.25 * W$, **and**

density < 0.42 , **and**

$W > 3 * THK$, **and**

No double rightConcave, **and**

(slopeDifference > 0.9 & maxDist > 5 pixels **or** leftDistIncrease > 0), **and**

Not (number of horizontal lines with single segment $< 0.9 * L$ & there is at least a vertical line with double vertical segments & the index of last double ones $< 0.5 * W$), **and**

Not (number of vertical lines with single segment $< 0.8 * W$ & the index of first double ones $< 0.4 * W$ & rightConcave exists)

then the digit is six

elseif number of horizontal lines with single segment $> 0.7 * L$, **and**

number of vertical lines with single segment $> 0.9 * W$, **and**

maxDist ≥ 8 pixels

then the digit is six

end

Seven ('7'):

if $L > 3 * THK$, **and**

number of vertical lines with double segments $\leq 0.35 * W$, **and**

no two leftPeaks, **and**

if rightConcave exists, its height $< 0.7 * THK$, **and**

number of horizontal lines with double and three segments $\geq 0.3 * L$, **and**

$(\frac{3}{4} * \text{distance between first double horizontal segments} > \text{distance between last double horizontal segments} \ \& \ \text{no loop} \ \& \ \text{Not}(\text{slopeDifference} > 1 \ \& \ \text{maxDist} > 5) \ \text{or} \ \text{topConcave with horizontal index} > 0.5 * L \ \& \ \text{height} > 2 * THK \ \& \ \text{width} > 3 * THK)$

then the digit is seven

end

Eight ('8'):

if $L > 3 * THK$, **and**

number of vertical lines with double and three segments $< 0.3 * W$, **and**

horizontal index of last double horizontal segments $> 0.62 * L$, **and**

$(\text{topConvex with height} > 3 * THK \ \& \ \text{width} > 3 * THK \ \text{or} \ \text{number of horizontal lines with double segments} > 0.5 * L \ \& \ \frac{3}{4} * \text{distance between last double horizontal segments} > \text{distance between first double horizontal segments})$

then the digit is eight

end

Nine ('9'):

if $L > 4 * THK$, **and**

bottomDist2 < 1 , **and**

$(\text{Loop} = 1 \ \text{or} \ \text{there is at least a horizontal line with double segments} \ \& \ \text{horizontal index of last double horizontal segments} < 0.7 * L \ \& \ \text{number of vertical lines with double and three segments} > 0.1 * W \ \& \ \text{Not}(\text{slopeDifference} < 0.9 \ \& \ \text{maxDist} < 4) \ \& \ \text{Not}(\text{rightConcave with height} > 3 * THK \ \& \ \text{width} > 3 * THK) \ \& \ \text{slopeDifference} \geq 0.5)$, **and**

$\text{Not}(\text{number of horizontal lines single segment} > 0.7 * L \ \& \ \text{number of vertical lines single segment} > 0.8 * W \ \& \ \text{maxDist} \geq 8)$, **and**

if topConcave exists, its height $< 0.5 * L / THK$

then the digit is nine

end

Vitae

Name : Sameh Abdullah Bellegdi |

Nationality : Yemeni |

Date of Birth : 3/25/1984 |

Email : s.bellegdy@gmail.com |

Address : KFUPM, Dhahran, Saudi Arabia |

Academic Background : Sameh Bellegdi completed his Bachelor of Science degree in Computer Science in June 2007 from Al-Ahgaff University, Mukalla, Yemen. He joined King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, Saudi Arabia in February 2009. He is currently pursuing his Master of Science degree in Computer Science at KFUPM. Mr. Bellegdi has awarded a scholarship to pursue MS study in Computer Science in KFUPM from Hadhramout Establishment for Human Development, Yemen. He has received a Merit and Superiority Certificate for academic superiority during undergraduate study. He works in Pattern Recognition, Machine Learning, Computational Biomedicine, and Arabic Computing. He has published a conference paper in Computational Biomedicine. |