



FAKULTÄT FÜR
ELEKTROTECHNIK UND
INFORMATIONSTECHNIK

UNCONSTRAINED RECOGNITION OF OFFLINE ARABIC HANDWRITING USING GENERATIVE AND DISCRIMINATIVE CLASSIFICATION MODELS

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur

(Dr.-Ing.)

von **M.Sc. Moftah M. Elzobi**

geb. am 30. Dezember 1970 in Kairo, Ägypten

genehmigt durch die Fakultät für Elektrotechnik und Informationstechnik

der Otto-von-Guericke-Universität Magdeburg

Gutachter:

Prof. Dr.-Ing. habil. Ayoub Al-Hamadi

Prof. Dr.-Ing. Abbas Omar

Prof. Dr. Aly Farag

Promotionskolloquium am: 28. März 2017

Acknowledgements

I would like to thank my supervisor Prof. Dr.-Ing. habil. Ayoub Al-Hamadi for his guidance and relentless support over the course of this PhD. His commitment of research inspires me over all these years and motivates me to do my best.

I am grateful to Prof. Dr. A. Omar and Prof. Dr. A. A. Farag for accepting to review my thesis. I would like to thank my colleagues in NIT research group for their countless stimulating conversation and invaluable technical advices. Special thanks to my family for their infinite support and understanding.

Abstract

Despite extensive research conducted over the past three decades, a fact-proof solution to the problem of offline Arabic handwriting is still elusive. Moreover, most of the current solutions that are usually specific to a particular personal handwriting or to a certain font, deliver recognition results that are fraught with problems and errors. As a consequence, the development of efficient OCR software that is capable of transcribing Arabic handwriting into a searchable text is still an active area of research. On the other hand, in the current digital age, it is evident that information resources that are not properly digitized will simply become inaccessible. The main objective of our research, in this thesis is to investigate and develop effective recognition approaches for offline Arabic handwriting that are applicable in unconstrained OCR's environments.

Furthermore, we contend that carefully designed and adequately annotated datasets are a vital prerequisite for any unconstrained OCR's solutions. Therefore, we have developed IESK-arDB, a new multi-purpose Arabic handwriting database. It is publicly available and contains more than 6000 word images each groundtruthed with segmentation information, and 285 pages of 14th century historical manuscripts that are transcribed into text files, and a page-by-page line-by-line alignment is enforced. A letter frequency has analysis showed that the database

exhibits letter frequencies very similar to that of large corpora of digital text, which proves the database usefulness. Additionally, and since manually creating handwriting databases is a cost and time prohibitive process, a handwriting synthesis approach is proposed. And about 28000 online handwritten letter samples are collected from several writers and used to build 100 Active Shape Models (ASM). ASMs are then used to generate unique letter representations in order to simulate the various handwriting styles. The developed system is used to create more than 12000 synthesized samples that have been added to the database.

It has been concluded that handwritten word segmentation is a fundamental step in building any general purposes OCR system. Hence, one of our contributions in this thesis is the proposal of a new topological segmentation methodology. It starts by performing a connected-component analysis in order to resolve sub-words overlapping. Then, topological feature based segmentation is carried out to split the word into a set of presumed letters. The proposed approach has been successfully tested on IESK-arDB and IFN-ENIT databases, achieving very promising results that indicate the efficiency of the suggested approach. Informative and non-redundant features, typically, facilitate the subsequent recognition process. In this thesis, we propose a robust yet simple approach for extracting two sets of shape descriptor features that have a number of desirable characteristics, e.g. less expensive in extraction and in processing, efficiently capture letter global shape characteristics, invariant to stroke width and less sensitive to handwriting distortions (e.g. skew and slant). It has even been argued that recognition is one of the most essential phases in any OCR system. Generally speaking, there is a wide spectrum of current solutions for this problem. Those solutions can be probabilistic, non-probabilistic, or may adopt generative or discriminative modeling approach. Unlike the mainstream approaches addressing the problem, in this dissertation, firstly, we propose a generative HMMs based approach that is built on top of an explicit segmentation module. Thanks to a threshold model that is constructed

by ergodically connecting all letters models, the suggested approach is capable to detect false segmentation and non-letter segments. The approach is validated on two different databases with satisfactory results. Furthermore, and because of the strong performance that the discriminative CRFs and its extension HCRFs recently showed in several pattern recognition fields. We introduced those two approaches to the problem of offline Arabic handwriting recognition. For training, testing, and performance comparison purposes of all proposed recognition approaches the IESK-arDB and IFN-ENIT databases are used. The achieved results indicate the superiority of discriminative approaches, where HCRFs achieved the best performance followed by CRFs.



Zusammenfassung

Trotz mehr als zwei Jahrzehnten intensiver Forschung im Gebiet der arabischen Offline-Handschrifterkennung, steht eine zufriedenstellende Lösung bisher noch immer aus. Sogar spezifische Ansätze für einzelne Schreiber oder bestimmte Schriftarten sind problematisch und fehlerbehaftet. Bis heute existiert daher keine OCR-Software, welche es ermöglichen würde, arabische Handschrift zuverlässig in digitalen Text zu überführen. Andererseits ist es im Kontext des digitalen Zeitalters ersichtlich, dass nicht angemessen digitalisierte Quellen zunehmend unzugänglich werden. Der wesentliche, in dieser Dissertation zusammengefasste Forschungsschwerpunkt besteht in der Erforschung und Entwicklung von effektiven Ansätzen zur Offline-Erkennung arabischer Handschrift für den Einsatz in allgemeinen OCR-Umgebungen. Sorgfältig konzeptionierte und adäquat annotierte Datensammlungen sind typischerweise Voraussetzung für effiziente, allgemeine OCR-Lösungen. Daher haben wir die IESK-arDB entwickelt, eine neue Vielzweck-Datenbank arabischer Handschrift. Die Datenbank ist frei verfügbar und umfasst über 6.000 Abbildungen von Wörtern sowie der zur Validierung von Vorverarbeitung, Segmentierung und Erkennung erforderlichen Grundwahrheiten. Weiterhin sind 285 Seiten historischer Dokumente aus dem 14. Jahrhundert beinhaltet, denen aus Unicode-Textseiten bestehende, zeilengetreue Grundwahrheiten beiliegen. Durch

Analyse der Buchstabenhäufigkeit wurde gezeigt, dass die Häufigkeiten der Buchstaben der IESK-arDB jener umfassender digitaler Textdatenbanken entspricht, und somit nützlich für allgemeine Anwendungszwecke ist. Aufgrund der mit manuell erstellten Datenbanken verbundenen Kosten, wird ein Ansatz zur Synthese von Handschrift vorgestellt. Um die 28.000 handschriftlichen Online-Buchstabenproben wurden von verschiedenen Schreibern gesammelt und verwendet, um über 100 Active-Shape-Models (ASMs) zu erstellen. Die ASMs werden benötigt, um unikale Buchstabenrepräsentationen zur Simulation verschiedener Schreibstile zu generieren. Das entwickelte System wurde eingesetzt, um über 12.000 synthetische Proben zu erstellen, welche zur Datenbank hinzugefügt wurden.

Segmentierung handschriftlicher Wörter ist ein fundamentaler Schritt beim Erstellen aller OCR Systeme, die für allgemeine Zwecke eingesetzt werden sollen. Daher stellen wir als wesentlichen Beitrag zur Forschung eine topologiebasierte Segmentierungsmethode vor. Zunächst wird durch eine Analyse zusammenhängender Komponenten das Problem überlappender Unterwörter gelöst. Anschließend wird, basierend auf topologischen Merkmalen, eine Segmentierung des Wortes in Buchstaben durchgeführt. Dieser Ansatz wurde mit vielversprechenden, auf seine Effizienz hinweisenden Ergebnissen an der IESK-arDB- sowie der IFN-ENIT-Datenbank getestet. Informative, nichtredundante Merkmale begünstigen typischerweise den anschließenden Erkennungsprozess. In dieser Dissertation wird ein robuster aber dennoch einfach gehaltener Ansatz zur Extraktion zweier Shape Deskriptoren vorgestellt. Diese weisen eine Reihe erwünschter Eigenschaften auf. So lassen sie sich z.B. mit geringem Aufwand extrahieren und verarbeiten und sind, da die globale Buchstabengestalt erfasst wird, invariant gegenüber der Strichdicke sowie weniger empfindlich gegenüber in Handschrift auftretenden Deformationen wie Schriftneigung oder -schräge.

Erkennung stellt offenkundig eine der wichtigsten Phasen jedes OCR Systems

dar. Allgemein steht ein weites Spektrum verschiedener probabilistischer und nicht-probabilistischer – auf generativen oder diskriminativen Modellen basierender – Lösungsansätze zur Verfügung. Im Gegensatz zu den gängigen Lösungsätzen wird in dieser Dissertation erstmal ein generativer Hidden-Markov-Model (HMM) basierter Ansatz vorgeschlagen, der an ein Modul expliziter Segmentierung anschließt. Ein Schwellwert-Modell, konstruiert durch ergodische Verknüpfung aller Buchstabenmodelle, ermöglicht es, Segmentierungsfehler und nicht Buchstaben Segmente zu detektieren. Der Ansatz wurde anhand zweier Datenbanken mit zufriedenstellenden Ergebnissen validiert. Ein weiterer Ansatz basiert auf diskriminativen Conditional-Random-Fields (CRFs), die neuerdings eine hohe Performanz in diversen Bereichen der Mustererkennung erzielen. Beide Ansätze werden hinsichtlich ihres Einsatzes für arabische Handschrifterkennung untersucht. Zum Trainieren, Testen und zum Vergleichen der Performanz von CRFs, HCRFs und HMMs wurden die IESK-arDB sowie die IFN-ENIT Datenbanken eingesetzt. Die Ergebnisse weisen auf die Überlegenheit der diskriminativen Ansätze hin, wobei HCRFs gefolgt von CRFs die beste Performanz aufweisen.



Dedication

*With love and gratitude, this work dedicated to my family who kept me going when I
wanted to give up halfway.*

Moftah

Declaration

I, hereby declare that the presented work is done without undue assistance from third parties and have made no use other than the indicated resources directly or indirectly. Some parts of the work presented in this thesis have been published in international conferences and journals (in publication list).

Magdeburg, 22. June 2016

Moftah Elzobi

Table of Contents

Acknowledgements	ii
Abstract	iii
Zusammenfassung	vii
Dedication	ix
Declaration	x
Table of Contents	xi
List of Figures	xvii
List of Tables	xxv
1 Introduction	1
1.1 Arabic Handwriting Characteristics	2
1.2 Motivations and Applications	4
1.3 Goals and Contributions	5
1.4 Previous Work	7
1.4.1 Databases for Arabic Handwriting	7
1.4.2 Handwriting Segmentation	10
1.4.3 Handwriting Recognition	12
1.5 Overview of the Manuscript	14

2	Fundamentals of Offline Handwriting Recognition	17
2.1	Methodologies of Handwriting Processing	18
2.1.1	Online and Offline Handwriting	18
2.1.2	Analytic vs. Holistic Recognition	19
2.2	Data Acquisition for Offline Handwriting Recognition	20
2.3	Document Image Pre-processing Techniques	22
2.3.1	Image Binarization	22
2.3.2	Thinning	25
2.3.2.1	Iterative Thinning Approaches	26
2.3.2.2	Non-iterative Thinning Approaches	26
2.3.3	Noise Suppression and Smoothing	27
2.3.4	Baseline Estimation and Skew Correction	29
2.3.4.1	Profile based Methods	30
2.3.4.2	Hough Transform based Methods	31
2.3.5	Slant Correction	32
2.3.6	Image Size Normalization for Handwriting	34
2.4	Segmentation	35
2.4.1	Explicit Segmentation	36
2.4.2	Implicit Segmentation	37
2.5	Generative and Discriminative Recognition	38
2.5.1	Hidden Markov Models (HMMs)	40
2.5.1.1	HMMs Basic Definitions	40
2.5.1.2	HMMs Modeling for Handwriting	42
2.5.1.3	HMMs Topologies	43
2.5.1.4	The Three Classic HMMs Problems	44
2.5.2	Conditional Random Fields CRFs	49
2.5.2.1	Learning Parameter for CRFs	51
2.5.2.2	Inference CRFs	52
2.6	Conclusion	53

3	A Benchmark Database for Offline Arabic Handwriting and Arabic Handwriting Synthesizing	55
3.1	Offline Handwriting Database	57
3.1.1	Data Acquisition	57
3.1.2	Database Annotation	59
3.1.3	Letters Frequency Analysis	62
3.2	Handwriting Synthesis	64
3.2.1	Online Data Acquisition	65
3.2.2	Modeling and Synthesizing of Letter Shapes	66
3.2.3	Synthesizing of Handwritten Words	67
3.3	Conclusion	68
4	Segmentation of Arabic Handwriting	71
4.1	Segmentation of Text Lines	73
4.1.1	Hough Transform based Approaches	73
4.1.2	Smearing based Approaches	74
4.1.3	The Proposed Approach	75
4.1.3.1	Page Skew Correction	76
4.1.3.2	Text Lines Segmentation	76
4.2	Segmentation of Arabic Handwritten Words	79
4.2.1	Handwriting Specific Pre-processing	79
4.2.1.1	Skew Correction Approach for Handwritten Arabic Words	80
4.2.2	Segmentation of Handwritten Words into Letters	82
4.2.2.1	Resolving Sub-words Overlapping	83
4.2.2.2	Segmentation of Letter Representative	85
4.3	Conclusion	88
5	GENERATIVE & DISCRIMINATIVE BASED APPROACHES FOR RECOGNITION OF HANDWRITTEN ARABIC WORDS	91
5.1	Sequential Shape Descriptions Features for Arabic Handwriting . . .	93
5.1.1	Extraction of Features Descriptors	94
5.1.2	Vector Quantization for Feature Sequences	97
5.2	HMMs based Recognition of Arabic Handwriting	98

5.2.1	Shape based Letters Taxonomization	98
5.2.2	The HMMs Topology and Hidden States Optimization	100
5.2.3	HMMs Parameters Initialization	101
5.2.4	HMMs Models Construction	103
5.2.5	Threshold Models Construction	105
5.2.6	HMMs based Recognition	107
5.3	Linear Chain CRFs based Recognition of Arabic Handwriting	109
5.3.1	CRFs Parameters Learning	112
5.3.2	Class Label Prediction	112
5.4	HCRFs for Arabic Handwriting Recognition	114
5.5	Conclusion	115
6	Experiment and Recognition Results	117
6.1	Evaluation of Text Lines Segmentation Method	118
6.2	Evaluation of Sub-words Overlapping Resolver	122
6.3	Evaluation of the Handwritten Word Segmentation Approach	123
6.4	Recognition Experiments	127
6.4.1	Evaluation of HMMs Recognition Performance	128
6.4.2	Evaluation of CRFs and HCRFs Recognition Performance	132
6.4.3	HMMs vs. CRFs vs. HCRFs	137
6.5	Conclusion	139
7	Conclusions & Future Perspectives	141
7.1	Summary and Key Contributions of the Thesis	141
7.2	Future Perspectives	145
	Bibliography	147
	Appendices:	163
	Appendix A IESK-arDB Database and Handwriting Synthesis	165
	Appendix B Segmentation Results	167
B.1	Text line segmentation	167
B.2	Resolving sub-words overlapping	169
B.3	Words Segmentation	170

CONTENTS

xv

Appendix C Recognition Results	173
Concise Curriculum Vitae	177
Related Publications	179

List of Figures

2.1	Online and offline handwriting: (a) trajectory of online Arabic handwritten word, where numbers indicate the sequence and red arrows show the direction of writing, (b) an image of offline handwritten Arabic word.	19
2.2	The typical structure of the holistic and analytic recognition of offline handwriting.	20
2.3	The most popular document scanning configurations: (a) Flatbed scanner, (b) overhead scanner [1].	22
2.4	Local vs. global binarization approaches: (a) A degraded manuscript image, (b) result of Otsu global based binarization approach, (c) result of Niblack local based binarization approach, (d) handwritten word image with a uniform background, (e) result of global approach, (f) result of local approach.	25
2.5	Thinning of word image: (a) The source binary image, (b) result of Stentiford algorithm and, (c) results of Zhang-Suen algorithm.	27
2.6	Word image preprocessing using closing and opening operations: (a) The noisy source image, (b) result of opening operation and, (c) result of closing operation.	29

2.7	Skew correction based on the image profile: (a) A skewed handwritten word and its corresponding projection profile, (b) The skew free version and its corresponding projection profile. The blue line is the handwriting base line and the red line is an imaginary line intersecting the peak of the profile.	30
2.8	Skew correction based on HT: (a) a skewed handwritten word, (b) the corresponding Hough space, (c) the estimated baseline.	31
2.9	Slant correction: (a) A slanted handwritten word and its profile histogram, (b) the slant-free version and the corresponding profile histogram.	32
2.10	Average slant angle estimation using chain code sequence [2].	34
2.11	Generative vs Discriminative models: (a) The basic probabilistic structure of generative models, (b) The basic probabilistic structure of discriminative models.	39
2.12	Variations in length and handwriting of two samples of the same words. Feature vectors are usually extracted from multiple vertical stripes along the word image.	42
2.13	A possible modeling of the handwritten word (two images of the same word) in FIG. 2.12 using HMMs.	42
2.14	Schematic representation of different HMMs topologies: (a) connection structure of an ergodic model, (b) Left-to-Right (Bakis) model, and (c) linear (banded) model.	43
2.15	A first order CRFs. Top are the states sequence and bottom are observation sequence. Dotted lines indicate connections between the observation and every other state [3].	50
3.1	A screen-shot from the IESK-arDB database website.	56
3.2	A Page from the form used to collect data.	58

3.3	Samples of handwritten words: (a) Gray scale images, (b) binary images, and (c) thinned images.	59
3.4	An XML ground truth file describing the word image shown in FIG. 3.5.	60
3.5	Example of the ground truth information given in FIG.3.4, plotted against its corresponding binary image.	61
3.6	Manuscript sample from the IESK-arDB database, and the equivalent machine-readable text with one-to-one line correspondences.	62
3.7	The frequency distribution of Arabic letter in IESKarDB, sorted according to the alphabet sequence.	63
3.8	The letters frequency in IESK-arDB compared to the letters frequency in the sources used in [4].	64
3.9	On-line data acquisition: (a) Acquisition system (the base unit and the digital pen), (b) screen-shot of system interface, and (c) an example illustrates the structure of the ground truth information file.	65
3.10	Samples used to build Active Shape Models (ASMs) for the letter Sin (س): (a) samples of training trajectories with 10 landmarks points, vs. , (b) with 25 landmarks points.	67
3.11	Paragraph synthesizing: (a) A digital paragraph taken from the Arabic site of the Deutsche Welle (http://www.dw.de/), (b) the synthesized handwriting corresponding to the text marked in yellow.	69
4.1	Issues that complicate text line segmentation: (a) Touching of components, from two consecutive lines, (b) vertical overlapping of components, (c) various skew degrees across the whole page and also along the same line.	72

4.2	Text lines segmentation: (a) Binary skew corrected image, (b) the sub-images, where each sum-image depicted along its horizontal projection profile, (c) the segmented text lines.	75
4.3	Performance on a handwritten signed and stamped document: (a) A binary image of document written in the begin of the 20th century, (b) the result of the proposed approach with each text lines is individually identified with different colors.	78
4.4	Performance on handwritten historical manuscripts: (a) A source image of a medieval manuscript, (b) the best result obtained at tolerance values $\eta = 1$	79
4.5	Skew correction and baseline estimation: (a) Original binary image and its corresponding horizontal projection profile, (b) corrected version using LMR only, (c) corrected version using the proposed combination of HT and LMR techniques.	81
4.6	Critical feature points CFPs: (a) Different types of CFPs, (b) a thinned handwriting image with all possible CFPs.	83
4.7	Sub-words Overlapping: (a) Overlapped sub-words within a word baseline, (b) the overlapping free version.	84
4.8	Handwriting segmentation: (a) All possible cut candidates, (b) candidates after excluding column that contains EP, LP, BP or DP, (c) candidates after excluding candidates with the direct left neighbor, (d) the result after applying the proposed heuristic rules.	86
4.9	Segmentation of Sub-words vertical overlapping.	88
4.10	Samples results of the proposed segmentation approach: (a) results of samples taken from the IESK-arDB database. (b) results of samples taken from IFN/ENIT database.	89

5.1	Extraction of features descriptors: (a) Extracted features computation, (b) segmented handwritten word, (c) clockwise features, and (d) anticlockwise features.	96
5.2	Letter shape descriptor profiles (for the letter in FIG. 5.1 (a)): (a) The profile of clockwise shape descriptor, (b) the profile of anticlockwise shape descriptor.	98
5.3	Structure of the proposed HMMs recognition system. The system contains 104 models for letters basic shapes and 18 models for letter shapes that may appear in more than one taxonomy.	99
5.4	Letters taxonomy, according to the number of segments and the existent of loop(s).	99
5.5	Comparative performance of HMMs based models using Left-to-Right (Bakis) topology vs. using Left-to-Right banded topology. . . .	101
5.6	Optimization of the HMMs model size, the best recognition rates for Tax. 1, 2, 3, and 4, are achieved with models of size 5, 8, 10, and 10, respectively.	102
5.7	A plain structure of the HMMs threshold model.	106
5.8	A simplified overview of the HMMs based recognition approach, where \oplus means the combination of the results of anticlockwise and clockwise classifiers for the letter "KAF" (ك). The rest of the letters are classified similarly.	110
5.9	A simplified overview of the proposed linear-chain CRFs based recognition approach.	111
5.10	A simplified overview of the proposed hidden-state CRFs (HCRFs.) based recognition approach.	115

-
- 6.1 Sample of the images used in experiments: (a) A page from the IESK-arDB manuscript collection, (b) a page of a modern handwriting from the IESK-arDB database, (c) a sample page from the AHDB database. (d), (e), and (f) are the corresponding results of our text line segmentation approach. 119
- 6.2 Optimization experiments for η and Υ parameters : (a) Performance rates on IESK-arDB modern handwriting collection, (b) performance rates on IESK-arDB manuscript collection, and (c) performance rates on AHDB handwritten page collection. 121
- 6.3 Types of observed errors: (a) Example of Type I error, (b) example of Type II error. The green arrow indicates the affected stroke. 122
- 6.4 Evaluation results of resolving of sub-word's overlapping: Experiments conducted on two different data sets from IESK-arDB and IFN-ENIT databases. 123
- 6.5 Examples for successfully resolved overlapping, where word images taken from IESK-arDB and IFN-ENIT databases, respectively. 124
- 6.6 Word based segmentation performance of our proposed approach on IESK-arDB and IFN-ENIT: Over Seg. and Under Seg. stand for the percentage (rounded) of words containing over segmentation and under segmentation errors, respectively. Correct Seg. is the percentage of correctly segmented words. 126
- 6.7 Over and Under segmentation errors: (a) Over segmentation problem where letter ش is erroneously segmented into two parts, (b) under segmentation problem with two letters ج and ك merged into one. 127
- 6.8 Examples for successful word segmentation. 128
- 6.9 Letter based recognition performance of HMMs: Recognition rates of letters samples from IESK-arDB and IFN-ENIT databases with the amounts of different classification of errors. 129

6.10	Word based recognition performance using HMMs: Percentage of recognized words with the respective recognition reliability values.	131
6.11	CRFs' and HCRFs' performance with different window sizes (ω): (a) Performance on Tax.1, (b) performance on Tax.2, (c) performance on Tax.3, and (d) performance on Tax.4	133
6.12	CRFs and HCRFs performance on letter shapes under taxonomy Tax.2: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database.	135
6.13	CRFs and HCRFs word based performance on evaluation sets of IESK-arDB and IFN-ENIT databases: (a) CRFs performance, (b) HCRFs performance.	136
6.14	CRFs and HCRFs training cost in terms of time: (a) Training cost of CRFs and HCRFs for Tax.1 and Tax.2, (b) training cost of CRFs and HCRFs for Tax.3 and Tax.4.	136
6.15	Overall recognition performance: The average performance of the three approaches for recognition of segmented letters, and for recognized words with confidence values greater than 0.5.	137
6.16	Letter forms based performance: Performance comparison of the three approaches according to the different letter written forms.	138
6.17	Letter forms based performance: Performance comparison of the three approaches according to the different letter written forms.	139
A.1	A manuscript page and its corresponding transcription.	165
A.2	Samples of single words synthesizing.	166
A.3	An example of a line-by-line synthesizing of Arabic handwriting from block of Unicode text.	166
B.1	Results of text line segmentation approach.	168
B.2	Correctly segmented words.	170

B.3 Samples results with under segmentation errors. Under segmentation errors happen when either a consecutive letter in a word is (fully) vertically overlapping the previous one, or when it extremely elongated to the right thereby vertically overlapping one or more previous letters. 171

B.4 Samples results with over segmentation errors. These types of errors are typical when CFPs occur inside the letter’s main body. 171

C.1 CRFs and HCRFs performance on taxonomy Tax.1: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database. 173

C.2 CRFs and HCRFs performance on taxonomy Tax.3: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database. 174

C.3 CRFs and HCRFs performance on taxonomy Tax.4: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database. 174

List of Tables

1.1	Standard Arabic Alphabet	3
1.2	Arabic and Farsi most commonly used databases.	9
3.1	Effects incorporated into the synthesized system.	68
6.1	The results achieved using IESK-arDB modern handwriting (HW.) test data, IESK-arDB manuscripts (Manu.) test data, and on AHDB test data, where <i>GT. lines</i> are the groundtruthed text line and <i>Det. lines</i> are the detected text lines.	121
6.2	Letter-based performance comparison of our proposed segmentation approach and the approach proposed in Xiu et al. [5].	125
6.3	Examples of the inputs and the results of the HMMs based recognition approach. Notice in the third row the third segment (from the right) is rejected hence it is replaced with #. Also the first segment (from the right) in the fourth row is recognized as "ﺝ" with 50% possibility of substitution error.	130
6.4	Letter based performance of CRFs and HCRs across IESK-arDB and IFN-ENIT database for the four letter taxonomies.	134

B.1	More results for the proposed sub-words overlaps resolving approach, samples are taken from IESK-arDB and IFN-ENIT databases.	169
C.1	Letter based recognition results on IESK-arDB database.	175
C.2	Letter based recognition results on IFN-ENIT database.	176

CHAPTER 1

Introduction

IN the current digital age, people expect that information resources (modern or historic) are digitally available and can be fast and easily accessed. Resources that are not properly converted into a machine-readable text (e.g., Unicode or ASCII formats) will soon become obsolete or even inaccessible. This would imply a significant loss of an important and huge amount of human cultural memory.

Despite the enormous advances in computing power, paper-and-pen handwriting has been (and still is) one of the main approaches for preserving and collecting information. In the 1970s, George Pake the founder and the executive head of Research and Development (R&D) at Xerox corporation (and many others) predicted that offices will be paperless by 1995. Obviously, the current 21st-century office life is contradicting such prediction. In a well-known book, published in 2003, Sellen et al. ¹, argue that paper-and-pen will continue to play an important role in offices, and they recommend related research efforts to be focused to solutions that make optimal use of written/printed paper and electronic document tools. In this context, the term Optical Character Recognition (OCR) is coined to refer to the process of converting images of handwritten, typewritten, or printed text

¹*The Myth of the Paperless Office* [6]

into a machine-readable text. For type-written and printed Latin based scripts, research reached a point where OCR software solutions are now everywhere, with accuracy rates exceeding 99%. Hence, it is considered as a solved problem by many researchers. However, Latin based cursive handwriting is still an open topic for research.

In comparison to Latin based scripts, Arabic alphabet based scripts which are the focus of this thesis, the OCRs research has started relatively recently and advances slowly, and hence it is still an open and challenging field of research for both printed and handwritten text. Considering the fact that solving the OCRs problem of printed Arabic text is, typically, a by-product of solving the more challenging problem of Arabic handwriting OCR. The main objective of this work will be to investigate and propose an OCR's solution for Offline Arabic handwriting.

1.1 Arabic Handwriting Characteristics

Arabic is a unicast alphabet that includes 30 letters, 28 basic and two additional contextual variants of two basic letters (TaMarbuta ة and AlifMaksura ى). As a result of adopting the alphabet to write other languages e.g., Persian, Urdu, Kurd, etc., the standard alphabet is modified by omitting and adding letters to represent vowels specific to each language. Many letters share the same main part (*RASM*) (e.g., ب , ت , ث), hence, dots are extensively used to distinguish one letter from the other, where the number and the position (i.e. above, under, or inside) of dots make the difference. In the standard alphabet, ten letters come with one dot, three with two dots, and two with three dots. Typically, Arabic letters are written right-to-left, ascending or descending from a clear baseline, and written cursively in both handwritten and in printed writing. Except six letters, Arabic letters appear in four different forms (Begin, Middle, End, and Isolated) according to their positions in a word. Furthermore, the same letter can have a completely different appearance

depending on its position (see the letter "ه" pronounced as "He" in Table 1.1). The other six letters can be connected to the preceding letter only, therefore they appear only in two different forms, namely, End and Isolated forms. As consequence, a word will be splitted into two or more parts (i.e. sub-word) whenever any of these six letters occurs within it. This phenomenon is specific to Arabic alphabet based scripts, and often causes word segmentation problems, such as sub-words overlapping and word - sub-word confusion².

Table 1.1. Standard Arabic Alphabet

	I	E	M	B		I	E	M	B
Alif	ا	ا			Taa	ط	ط	ط	ط
Ba	ب	ب	ب	ب	Dha	ظ	ظ	ظ	ظ
Ta	ت	ت	ت	ت	Ayn	ع	ع	ع	ع
Tha	ث	ث	ث	ث	Ghayn	غ	غ	غ	غ
Jim	ج	ج	ج	ج	Fa	ف	ف	ف	ف
Ha	ح	ح	ح	ح	Qaf	ق	ق	ق	ق
Kha	خ	خ	خ	خ	Kaf	ك	ك	ك	ك
Dal	د	د			Lam	ل	ل	ل	ل
Thal	ذ	ذ			Mim	م	م	م	م
Ra	ر	ر			Nun	ن	ن	ن	ن
Zai	ز	ز			He	ه	ه	ه	ه
Sin	س	س	س	س	Waw	و	و		
Shin	ش	ش	ش	ش	Ya	ي	ي	ي	ي
Sad	ص	ص	ص	ص	TaMarbuta	ة	ة		
Dhad	ض	ض	ض	ض	AlifMaksura	ى	ى		

Ligatures in Arabic based scripts are frequently used, where a ligature is formed when two or even three consecutive letters are vertically connected. Only one ligature is mandatory لا (Lam ل + ا Alif) and usually treated as an additional letter. Other ligatures, such as the combination of مح, نم, and لم are optional and occur only

²White-spaces can occur inside the same word, as well as, between different words.

as a result of the used font and/or the personal handwriting style. Table 1.1 lists letters of the standard Arabic alphabet along with their different forms.

1.2 Motivations and Applications

The major impediment in large-scale business application is how to digitize type-written, printed or handwritten data embedded in hard copy documents. The traditional way to overcome these obstacles is to manually transcript each document image, which seems to be expensive, repetitive, and time consuming. Alternatively, OCR is proved to be a practical and reasonable solution to generate digital redundancies of the hard documents.

As an example of the urgent need for Arabic alphabet based OCR is the case of the Ottoman Archives. These Archives are estimated to contain more than 150 million documents, which hold invaluable information about the history of current 39 independent states that emerged from the Ottoman Empire. This information is still fundamental for present issues like treaties, border disputes, inheritance, court documents, land deeds, applicable laws, historical demographics, etc. Until 1928, the Arabic alphabet based Ottoman Turkish was the official script of both the archives and the empire, then replaced with Latin-based Turkish alphabet. Up to today, people who can read Ottoman Turkish are limited to a very few numbers of historians and theologians [7]. In this particular case, an efficient OCR solution is a necessary, not only to preserve, easily access, and mine the information of such priceless historical resources, but also to transliterate the Ottoman Turkish resources into modern Turkish or any other language, which is a cost and time efficient alternative. In general and regardless of the underlying script, when developing an OCR system there are two possible approaches, namely, task-specific or general-purposes. Task-specific OCR systems find application in various domains such as, banking, postal, insurance, etc. On the other hand, general-purpose OCR systems

are typically needed when the task requires processing document images that contain unconstrained text, e.g., pages of newspapers, business letters, pages of handwriting, copies of historical manuscripts, etc. Lexicon-based task-specific OCR solutions have been successfully applied to a certain extent to Arabic based scripts. However, such solutions are inadequate or even unusable in large vocabulary or vocabulary-independent application environments. To develop a general propose (vocabulary-independent) OCR system, we claim that the segmentation step is of prime importance [8,9]. Such a fact has motivated the development of this work, in which segmentation based recognition approaches for offline Arabic handwriting are proposed.

1.3 Goals and Contributions

The main objective of this work is to investigate and develop efficient approaches for the recognition of offline Arabic handwriting. The proposed methods should be highly invariant to personal writing styles, the used font, and document type (modern or historical handwriting). Moreover, the suggested OCR solutions should be robust against geometric distortions (e.g., scaling, translation, etc.) and handwriting specifics (e.g. , skew, slant, etc.). The following are the main research contributions of this dissertation:

- Considering the vital importance of databases for developing and validating related algorithms, we create our own IESK-arDB database. To the best of our knowledge, we believe that the database is the first of its kind that comes equipped with segmentation information. It contains images of single handwritten words, pages of modern Arabic handwriting, and pages of handwritten historical manuscripts. All of images are adequately and properly ground-truthed. Furthermore, in order to avoid the expensive process of manually generating handwriting samples, we proposed a novel and effective

Active Shapes Model (ASM) based approach to synthesize Arabic handwriting. Using this approach, a large number of sample images of synthesized pages of text and synthesized single words are added to the IESK-arDB database.

- Correction of skew and slant of handwritten words typically enhances the performance of subsequent processes. We proposed a simple yet effective method that combines the advantages of two well-known techniques for the normalization of the skew and the slant. Moreover, sub-words overlapping is an Arabic script specific issue that has been addressed in our research and a proper solution is presented.
- Text lines segmentation is an inevitable process in any OCR system. Our work contributed to this research topic by proposing an efficient method that can be applied on modern as well as on historical handwriting documents. The method proved to be remarkably robust against several kinds of handwriting specific distortions, e.g., touching and overlapping of ascenders and descenders, global and local skewness of text lines, and the extensive use of dots and diacritics in case of Arabic historical manuscripts. Segmentation of handwritten words into letters is the main bottleneck that hinders an unconstrained OCR solution. Therefore, having a solution for this problem would be a great step forward in the field. In our work, we developed a topological features based approach for segmentation of handwritten Arabic words. The proposed approach has been successfully tested on two different databases and results were very promising, indicating the efficiency of the suggested approach.
- A new set of features that are less expensive to extract, invariant to stroke width, less sensitive to different distortions, capable of capturing distinctive shape characteristics, and easy to convert to a sequence, are proposed and used for recognition.

- For the recognition task, we firstly proposed generative Hidden Markov Models (HMMs) equipped with an adaptive threshold model designed to cope with any possible meaningless shapes probably occurring as a result of segmentation errors. Secondly, to investigate the performance of the discriminative Conditional Random fields (CRFs) and the Hidden Conditional Random fields (HCRFs), to our knowledge, we are the first to introduce CRFs and HCRFs classifiers for the recognition of Arabic handwriting. Finally, the performances of HMMs, CRFs, and HCRFs are fully (separately and combined) evaluated; the strengths and weakness of each approach are discussed, and recommendations for possible future application of each are also given.

1.4 Previous Work

In this section, we briefly survey the most related research works addressing the main relevant sub-problems, namely, the demand for carefully designed and well articulated datasets, the handwritten words segmentation, and finally the offline handwriting recognition issue.

1.4.1 Databases for Arabic Handwriting

In the recent literature, there are multiple research works which make use of several off-line Arabic text databases. Table 1.2, summarizes most of the published databases that support Arabic OCR research. The most common database in the field of off-line Arabic handwriting recognition is the IFN/ENIT database, which contains exclusively tunisian town/village names. The database was created by the Institute of Communication Technology (IFN) at Technical University Braunschweig in Germany and the Ecole Nationale d'Ingénieurs de Tunis (ENIT) in Tunisia [10]. It has been reported that 411 writers have participated in generation of 26459 handwritten tunisian town/village names. The database is freely available at

www.ifnenit.com. The word images come with automatically generated ground-truth information, such as Baseline coordinates, number of sub-words, number of characters, etc. It contains no information about character borders or sub-word borders that can be helpful in facilitating the training and validation processes of any segmentation-based recognition approach. In the Center of Excellence for Document Analysis and Recognition (CEDAR) 10 persons participated in creation of an Arabic database by writing 10 different page of text each. Every page comprises approximately 150-200 word. In total, the database contains 20,000 word images [11]. Currently, this database is not available online.

In [12], a database for off-line Arabic handwriting (AHDB) is presented. Samples are collected from 100 writers. The database contains words used in writing bank legal amounts, the most popular arabic words, and freely available handwritten pages of text. Even though it has been reported that the database is freely available it can not be found on the Internet. Another database of handwritten checks (CENPARMI) is introduced in [13]. It consists basically of 3000 check images. From which 29,498 sub-word images, 15,175 digit images, and 2,499 legal amount images are extracted and labeled. The database is designed to facilitate automatic check reading research for the banking and finance applications. The database is freely available. The Applied Media Analysis (AMA) developed an Arabic handwritten full page dataset. It contains a set of 5000 pages of handwritten text, transcribed by 49 different writers from six different countries. The collection contains various document types including, forms, memos, poems, diagrams, and number lists in both Arabic and Indic digits. The dataset is available and can be downloaded at charge [14].

Compared to offline handwriting, databases for type-written and printed text are larger in size and more comprehensive, since the process of collecting or producing such text can be easily automated. A large-scale database called APTI, is synthetically generated for Arabic Printed words. The database is extracted from

Table 1.2. Arabic and Farsi most commonly used databases.

Database	Availability	Size	Purpose
IFN/ENIT	Freely available	26,459 tunisian city names	Off-line hand-writing.
CEDAR Arabic dataset	Not available	100 pages of text, each comprises 150-200 word	Off-line hand-writing.
AHDB	Not available	Not reported	Off-line hand-writing
CENPARMI	Available	29,498 subwords, 15,175 digits, and 2,499 digits	Off-line hand-written legal mounts and bank checks.
Arabic-Handwritten 1.0	Partially available	5000 handwritten pages	Off-line hand-writing
APTI	Available	45,313,600 word images	Arabic printed words.
ERIM	Not available	750 pages of text	Typewritten and printed text.
IBN SINA	Available	more than 1000 Sub-words	Arabic Manuscript.
IfN/Farsi	Available	7271 word images	Farsi Handwriting.
FHT	Available	1000 form images	Farsi Handwriting.

lixicons and contains 45,313,600 single word. It consists of more than 250 million characters, in 10 different fonts, 10 font sizes and 4 font styles [15]. The database is available through the web site (<https://diuf.unifr.ch/diva/APTI>) of DIVA group at the University of Fribourg, Switzerland. A database for machine printed Arabic text consists of 750 pages is created by the Environmental Research Institute of Michigan (ERIM). Images with different quality degrees are extracted from books and magazines typed in different font types and sizes [16]. The database is currently inaccessible.

Farrahi and Chariet in [17], illustrate the process of creating IBN SINA database. The database contains 1000 Arabic sub-words extracted from 50 folios of a historical Arabic handwritten manuscript. It is freely available for download, with sub-word images and their corresponding ground-truth information stored as ".MAT" files.

The alphabet and the writing styles of both Arabic and Farsi scripts are almost the same, hence OCR techniques or databases developed for one script can be used for the other. A database for Farsi handwriting called IfN/Farsi is presented in [18]. The database is developed in the same way as of the above-mentioned IFN/ENIT database. It contains 7.271 binary images of handwritten samples for Iranian province/city names, where data are collected from 600 writers³.

Another database for Farsi handwritten text is illustrated in [19]. The database called FHT, it contains 1000 forms filled out with passages of handwritten text. About 250 writers from different ages and education levels are participated in creating the database. The ground truth is a digital text correspond to the handwritten samples. To obtain this database, authors should be contacted.

1.4.2 Handwriting Segmentation

While the bulk of related literature did not report the results of segmentation separated from the recognition results, there are few numbers of research works dedicated only to the segmentation of Arabic handwriting [5,9,20]. In the following, we will briefly review the most important related literature, starting by works, in which, the segmentation is strongly linked to recognition. The last two works are samples of methods that focus mainly on the problem of Arabic handwriting segmentation.

One of the earliest segmentation-based approaches as suggested for the recognition of Arabic handwritten text, is the one proposed by Almulim and Yamaguchi [21]. In this approach, words are over-segmented into their basic strokes,

³The IfN/Farsi database can be easily obtained at no charge by contacting the authors.

where a stroke is the curve between any two structure points (end points or branch points). Each stroke is then classified to one of five groups according to its shape. Two of these groups contain what is called secondary strokes, and the other three groups contain primary strokes. Furthermore, a set of heuristics is proposed to assign secondaries to their primary strokes, to construct the corresponding character. No segmentation results are reported, while the reported recognition rate is 81.25%.

In [22], Bushofa and Spann propose a segmentation-based recognition methodology for off-line printed Arabic text. The segmentation algorithm starts by locating the text baseline. Having the baseline discovered, a fixed size window is used along the baseline to search for specific types of angles that are expected to be formed when letters joined together. Multiple heuristic rules are used to confirm the segmentation results. Finally, features are extracted and a decision tree based classifier is used for recognition. A recognition rate of 94.17% is reported.

In an attempt to avoid over-segmentation, Atici and Yarman-Vural [23] proposed an analytical segmentation approach for type-written Arabic text, which attempts to extract the whole stroke that represents the character by means of the so-called Character Key Feature Set (i.e., End-Points, Branch-Points, Loop-Points and Dot-Points) from the word thinned image. First, the minima and maxima of the thinned image are calculated, then the so called Key Features Segments (KFSg) are determined. Secondly, A set of heuristic rules that employ KF set, are applied on the set of the minima in order to elect cut candidates among them. A set of chain code features are extracted and a HMMs based classifier is used for recognition. Authors reported a recognition rate of 96%.

Abuhaiba et al. [24] presented a recognition system for off-line Arabic handwritten text. Their system is a segmentation-based approach, in which thinned and smoothed images of the strokes (sub-word) are processed and converted into 1D representations called direct straight-line approximation. The representation is processed further to produce a loop-less graph called the reduced graph, where

loops replaced by vertices. Ultimately, the reduced graph representation is segmented into tokens that are fed to a fuzzy sequential machine for recognition. As a sub-word (without segmentation) recognizer, the proposed system achieved 55.4% recognition rate. When segmentation is involved, the recognition rate degrades to 51.1%.

Lorigo and Govindaraju [9] propose an algorithm for segmentation of handwritten Arabic text. The algorithm integrated a gradient based, and down-up based techniques to detect all possible break-points. Then by exploiting a prior knowledge of letter shapes, all inaccurate candidates are filtered out. When testing the algorithm on a set of 200 images, a successful segmentation rate of 92.3% is reported.

In [5], Xiu et al. proposed probabilistic segmentation model, in which a tentative, contour-based over-segmentation is first performed on the text image. As a result, a set of what so-called graphemes is produced. The approach differentiates among three types of graphemes. The confidence of each character is calculated according to the probabilistic model, respecting other factors e.g., recognition output, geometric confidence and logical constraint. The authors tested their proposed approach on five different test sets, achieving 59.2% success rate.

1.4.3 Handwriting Recognition

This subsection will review the literature of recognition approaches that are comparable to the three different approaches proposed in this thesis, namely, the HMMs, CRFs, and HCRFs. Approaches that use HMMs to handle the considered problem has appeared at the beginning of the last decade. By integrating a right-to-left discrete HMM and Kohonen self-organizing map for features quantization, Dehghan et al. [25] were the pioneers in demonstrating the feasibility of applying HMMs for a holistic recognition of off-line Farsi and Arabic handwriting. Experiments were conducted on a database of Iranian cities names, and a sliding-window based

feature vectors are constructed from histograms of chain-code of contour directions. The reported accuracy rate of the approach is 65%. Pechwitz and Maergner [26] used 1-D semi-continues-HMMs based approach for the recognition of handwritten words, where a sliding window based feature vectors are extracted from normalized gray word images. Before passing features to HMMs, a Loeve-Karhunen transform is used to reduce dimensionality. The IFN/ENIT database is used for training and testing, and a maximum recognition rate of 89% is reported. For the recognition of off-line Arabic handwritten words, Al-Hajj et al. [27] proposed a combination of three right-to-left HMMs classifiers. Each classifier was constructed upon specific sliding window orientation to overcome the major substantial problems of offline handwriting such as, inclination, overlap, and shifted position of diacritics. Training and testing are conducted on the IFN/ENIT database, and the overall recognition decision is made by experimenting different combination techniques (i.e., sum rule, majority vote rule, and neural networks-based combination) among the results of three classifiers. The best recognition rate achieved is 90%, when a combination of neural networks is chosen. Dreuw et al. [28] proposed discriminatively trained HMMs for recognition of Arabic and Latin offline handwritten words. Instead of the popular expectation maximization (EM) training technique, authors suggested maximum mutual information (MMI) and minimum phone error (MPE) to train holistic models. The Arabic IFN/ENIT and the English IAM databases are used to evaluate the approach, where error-rates are reduced by 33% and 25% respectively, compared to the EM.

Motivated by the successful application of CRFs and HCRFs in a number of fields such as, natural language processing, bioinformatics, and etc., several works, recently, suggest the use of CRFs for the recognition of offline Chinese and Latin handwritings. To the best of our knowledge, we are the first [29] to introduce CRFs and HCRFs to the recognition of Arabic handwriting. Zhou et al. [30] propose a method for the recognition of Chinese/Japanese text based on semi-Markov

CRFs. They start by defining a semi-CRFs on a lattice of all possible segmentation-recognition hypotheses of a string to directly estimate their posteriori probabilities. CRFs feature functions are defined on top of geometric and linguistic information of character recognition, and negative log-likelihood is used for optimizing the model parameters. At the level of characters, recognition rates of 95.20% and 95.44% are achieved for Chinese and Japanese respectively. In their early related work, Feng et al. [31], explore and compare the performance of CRFs and HMMs for the task of word recognition in historical handwritten documents. A set of discrete features is extracted from 20 pages of George Washingtons manuscripts and used to train and evaluate the CRFs- as well as HMMs based classifiers. They conducted several experiments using different beam search methods in order to speed up the training process of CRFs classifier, and they have proved that CRFs is superior to HMMs. However, to boost performance, they found out it is necessary to reduce the state space by applying CRFs at the characters' level, which we adopted in our proposed approaches.

1.5 Overview of the Manuscript

This section provides an overview of the remainder of this thesis that is organized around five major chapters excluding this introductory chapter and the conclusion. In addition, there is an appendix accompanying the five chapters that contains tables, charts and graphics. The following chapter, introduces the fundamentals and theory that underpin OCR in general. Then, the main contributions of the thesis are described and discussed in detail within the scope of Chapters Three, Four, Five, and Six. In the seventh and last chapter, several conclusions are drawn and future perspectives are envisaged.

- **Chapter 2** describes the necessary fundamentals of OCR for handwriting, and illustrates the theoretical basics of methodologies that have been used

throughout this work. It begins by highlighting the differences between offline and online handwriting and explains the holistic and analytic processing approaches. Besides the pre-processing steps necessary for handwriting processing, the theory of the generative HMMs and the discriminative CRFs and HCRFs are also discussed.

- **Chapter 3** starts by giving an overview of the IESK-arDB database, where the details of data acquisition, samples annotation, and frequency analysis processes are respectively given. The second part of the chapter is dedicated to describe the ASM based handwriting synthesis method, where the details of the data collection process, the used equipments, and the letters as well as words synthesizing processes are illustrated.
- **Chapter 4** focuses mainly on the handwriting segmentation problem. Firstly, the proposed solution for the problem of text line segmentation is fully discussed, and the claimed effectiveness compared to other approaches is justified. Secondly, the suggested dual-phase segmentation method that begins with handwriting specific pre-processing such skew correction and sub-words overlapping is presented. Then, the process of generating candidates for segmentation of handwritten word and the used heuristic rules for segmentation are described in detail.
- **Chapter 5** is dedicated to introduce probabilistic based classification approaches suitable for labeling sequences of features. Two sets of sequential features describing the shape of the segmented unit along two different directions, namely, clockwise and counter clockwise, are extracted and used to train the proposed classifiers. The first adopted approach is a generative HMMs, that is built directly on top of an explicit segmentation module. The second and the third recognition approaches are the discriminative CRFs and its extension HCRFs. Moreover, the chapter demonstrates how confidence values

are computed and attached to the recognition results of the three classifiers to explain how reliable the obtained results are.

- **Chapter 6** summarizes and discusses the experimental results obtained from the various solutions suggested in this dissertation. Modern and historical samples drawn from IESK-arDB and AHDB databases are used to tune and evaluate the text line segmentation approach. The IFN-ENIT and IESK-arDB databases are used to assess the performance of sub-words overlapping and the word segmentation approaches. A pixel-matching based metric is adopted in the assessment, and different kinds of achieved rates and occurred errors are documented. As for recognition, all models are built using samples from IESK-arDB. In order to test the reliability of the adopted classifiers, additional experiments are conducted using unseen samples from the IFN-ENIT database. Finally, the chapter outlines several recommendations for future applications of the three classification methods (i.e., HMMs, CRFs, and HCRFs).

Fundamentals of Offline Handwriting Recognition

THE objectives of this chapter are, firstly, to describe the necessary fundamentals of offline optical handwriting recognition, and secondly to illustrate the theoretical basics of methodologies that have been used throughout this work. We believe that this chapter is essential for readers in order to evaluate the task, clarify terminologies, comprehend methodologies, and eventually appreciate results. To avoid any confusion, the chapter starts by explaining the difference between the offline and the online handwriting signals. Since offline handwriting recognition is the main theme of this thesis, the differences between the two main paradigms (i.e. analytic and holistic) of processing the offline handwriting signals are explicitly stated. Moreover, the signal acquisition systems and their various setups as well as the signal interpretations are discussed.

Before any recognition process can take place, it should be preceded by proper document image enhancement and preprocessing procedures, e.g., image thresholding, stroke thinning, outcropping pixels removal, etc. Section 2.3 presents the theoretical basics of the used techniques. Specific approaches have been used to handle handwriting artifacts such as skew and slant. These approaches are illustrated and their pros and cons are explained. Segmentation is an inevitable process in any unconstrained handwriting recognition system, hence Section 2.4 provides an overview of the segmentation of offline Arabic handwriting and emphasizes its importance within the context of our thesis.

Obviously, classification is one of the most essential and important phases in any recognition system. In general, there is a wide spectrum of solutions for the problem of classification. These solutions can be probabilistic, non-probabilistic, or may adopt generative or discriminative modeling approach. By analyzing the performance of three popular candidates, our work tries to highlight their strengths and drawbacks in terms of training costs and recognition results. Hence, in Section 2.5, the theoretical foundations of the generative Hidden Markov Models (HMMs) and the discriminative Conditional Random Fields (CRFs) are fully presented, and their application for handwriting recognition is briefly pointed out.

In the rest of this thesis, terms pair such as ("letter" and "character"), ("label" and "class), and ("sub-word" and "part of word (PAW)") are used interchangeably.

2.1 Methodologies of Handwriting Processing

Handwriting signal can be captured by scanning the image of the writing, or it can be extracted as the text is written using a special pen. These two different methods of signal acquisition generate two different representations of the handwriting (i.e., offline and online, respectively). Moreover, a handwritten word is typically processed as a whole (holistic), or as fragments (analytic) after a segmentation process. This section begins by explain the differences between online and offline handwriting, and then compares the analytic and the holistic recognition approaches.

2.1.1 Online and Offline Handwriting

According to the signal extraction process, there are two types of handwriting recognition approaches, namely online recognition and offline recognition. The signal of online handwriting is usually acquired during the writing with a special electronic styli on PDAs, tablet PCs, smart phones, infrared sensed white-boards, etc. The movement of the styli generates temporal sequences of coordinates that represents handwritten strokes (a stroke is the trajectory from a pen-down to a pen-up).

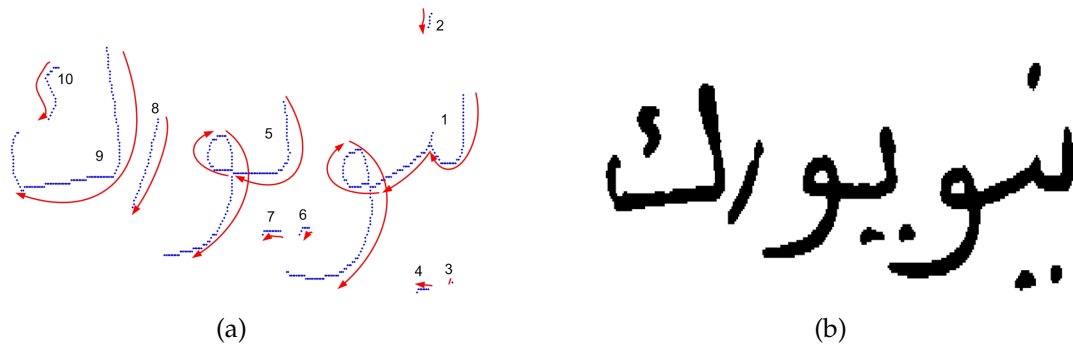


FIG. 2.1. Online and offline handwriting: (a) trajectory of online Arabic handwritten word, where numbers indicate the sequence and red arrows show the direction of writing, (b) an image of offline handwritten Arabic word.

This temporal information along the strokes trajectories help reduce the handwriting signal into a 1-D ordered vector of (x, y) coordinates, which in turn significantly eases the problem of handwritten word's segmentation into their constituent letters. In case of offline handwriting recognition, the signal representing the handwritten text is captured by scanning a previously handwritten or printed text, thus lacking any additional temporal or dynamic information. Moreover, the offline handwriting signal is only represented through a 2-D array of pixel values.

Generally speaking, the online handwriting recognition problem is proved to be less complex to be solved than that of the offline, due to the attached temporal information [32]. FIG. 2.1 (a) and (b), show two samples for online and offline handwritten word respectively.

2.1.2 Analytic vs. Holistic Recognition

Analytic recognition includes all approaches that perform explicit¹ or implicit² segmentation of handwritten words (into letters or primitives) prior to recognition [1]. The main advantage of such paradigms is their capability to cope with the high variability nature of the problem. The disadvantage, however, is the complexity and the error-prone characteristics of the segmentation process, which are attributed to the unconstrained nature of handwritten text. Difficulties such as diversity of character patterns, ambiguity and illegibility of characters, and overlapping of

¹Explicit segmentation approach performs an exhaustive search for potential segmentation points at character boundaries.

²An implicit approach segments (indiscriminately) a handwritten word into a sequence of equal-width frames.

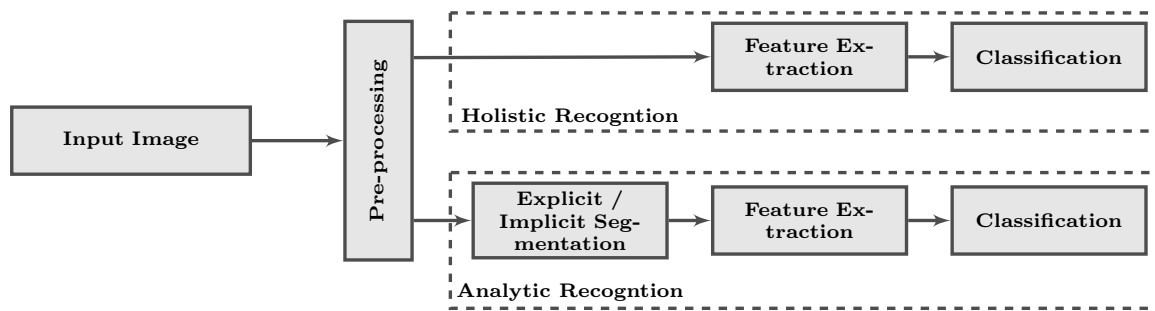


FIG. 2.2. The typical structure of the holistic and analytic recognition of offline handwriting.

characters within a word complicated the task of developing unconstrained and efficient segmentation algorithms. Until the date of writing this thesis, a robust segmentation based handwriting recognition system is still an elusive research challenge. Holistic recognition as its name implies, covers all approaches that are treating each word image as single entity from which features are extracted. Even though such approaches prove to be successful in some application areas, they are completely lexicon dependent and incapable to serve in an unconstrained environment. Nowadays, optical character recognition systems built upon holistic approaches are put successfully into service in a number of application areas, such as, automatic reading of postal addresses, bank checks processing, forms processing, etc. [33,34]. The reason why such systems cannot efficiently cover the wide spectrum of application areas, is due to the fact that such systems are completely avoiding character segmentation, which is an intuitively prerequisite operation to reduce the infinite domain of possible words into a limited number of classes (graphemes or characters) that can be accommodated and processed further [9, 35]. FIG. 2.2, illustrates the adapted recognition methodologies in both cases.

2.2 Data Acquisition for Offline Handwriting Recognition

The very first step in handwriting recognition is to convert the handwritten text into a digital form. There are various types of digitizing devices, where the choice of the appropriate one is usually subject to the nature of the handwriting (online or offline). For online handwriting, signal is often obtained during the writing process, either through the detection of the styli movement on touch sensitive surfaces, processing

of a light beam transmitted by an infrared pen, or by extracting the trajectory of a hand/finger that is writing in a video sequence [36].

In case of offline handwriting recognition, which is the main theme of this thesis, the signal is captured by scanning written, typewritten, or printed text that has been previously written on traditional writing mediums such as paper, canvas, leather, etc. Offline text recognition has a wide scope of application areas; starting from tabular form and bank cheque processing, postal code reading, to the recognition of historical manuscripts. Scanning process starts usually by steadily moving a source of light across the document, meanwhile a system of mirrors and lens is used to direct the reflected light on an array of tiny light-sensitive diodes called CCD (Charge-Coupled Devices). Diodes convert photons (light) into electrons (electrical charge), where the amount of the electrical charge is proportional to the light brightness that hit the diode.

Among others factors, optical resolution and color depth are the most important and accordingly the required scanners vary widely. Roughly speaking, resolution is the quantity of details an image has, which is often measured in dpi (dot per inch) or ppi (pixel per inch). The horizontal resolution (x-axis) are determined by the number of diodes along a single row of CCD, whilst the vertical resolution (y-axis) is determined by the step size of the scanner stepper motor. The color-depth or bit- depth term refers to the number of different colors a scanner may support, which resulted from the color quantization process. As an example, an 8-bit depth scanner will support up to 256 color levels, while 24-bit depth scanner support nearly 17 million different colors. Modern scanners usually enable a very high optical resolution (not interpolated) such as 9600×9600 dpi, and supporting up to 48 color-depth. A good-quality handwriting scanning with a reasonable file size is usually achieved with a resolution between 300 dpi to 600 dpi and a color-depth of 8 bits. For the purpose of document scanning, there are mainly two different types of scanners, namely flatbed scanner (see FIG. 2.3 (a)) and overhead or top-view scanner (see FIG. 2.3 (b)).

The former is the most widespread since they are versatile, handy and relatively cost-effective; nevertheless, they are inappropriate for fragile documents and book scanning, due to the fact that objects have to be pressed and detached (in case of a bound document), which may lead to disfiguration of the originals. To prevent



FIG. 2.3. The most popular document scanning configurations: (a) Flatbed scanner, (b) overhead scanner [1].

damaging valuable documents and enable effective book scanning, the latter type of scanners has been developed through the last decade. In the overhead scanners, the source of light and the CCDs (or a camera-system) are attached to an overhead arm by which a copy of the document can be captured from a distance.

2.3 Document Image Pre-processing Techniques

This section will describe some image pre-processing techniques that are considered as prerequisites for any further document image manipulation. Most often documents scanned in grayscale; hence the first step is to separate foreground pixels from the background, which is called binarization. Furthermore, to minimize the number of pixels to the minimum necessary for processing, a so-called thinning or skeletonization process is needed. Ultimately, the binarization and thinning processes may result in noisy and outcropping pixels, respectively, that need to be dealt with.

2.3.1 Image Binarization

Thresholding of grayscale images (binarization), i .e. the separation of foreground pixels (the writing) from the background, is a necessary first step before further

analysis. A popular and straightforward approach for image thresholding is to use an intensity value in pixel classification, assuming sufficient difference in pixel intensities. According to this approach, a pixel is classified as foreground pixel if its intensity value is smaller than the threshold value, otherwise it will be considered as background pixel [37].

Threshold based approaches can be categorized into two main categories; namely global threshold based approaches and local threshold based approaches. In the global threshold methods, a unique (global) threshold value is calculated from the entire image intensity values, whereas in the local based methods, a threshold is estimated for each pixel in the image according to local information extracted from the pixel neighborhood. Typically, a histogram³ constructed from the pixel gray values is used to estimate the global threshold value in the former category. Intuitively, such a histogram describes the gray values distribution over the entire image. For each gray level i , P_i denotes the probability density function of i level, where $\sum_{i=0}^n P_i = 1$ and $n + 1$ is the number of the gray levels.

In a gray level document image, the pixel intensities are most often cluster around two well-separated values resulting in the so-called bi-modal histogram. In order to find the threshold value that optimally separates the two modes, there are several proposed approaches [38]. The most common of them is the popular global Otsu approach [39]. In this approach, the optimal threshold value that minimizes the sum of the weighted variances (within-class variance) of both modes is calculated. Firstly, an initial guess t for the threshold value is assumed, whereby $i \leq t$ are pixels of one class and $i > t$ are the pixels of the other (foreground and background pixels). The corresponding variance weights q are then estimated using the sum of the relative probability P_i as follows.

$$q_1(t) = \sum_{i=1}^t P_i, \quad q_2(t) = \sum_{i=t+1}^N P_i. \quad (2.1)$$

Using Eq.(2.1), the means of both classes can be calculated from Eq.(2.2)

$$\mu_1(t) = \sum_{i=1}^t \frac{iP_i}{q_1(t)}, \quad \mu_2(t) = \sum_{i=t+1}^N \frac{iP_i}{q_2(t)}, \quad (2.2)$$

³A graph showing the number of pixels in an image at each different intensity value found in that image.

furthermore, the correspondence variances are also calculated using Eq.(2.3)

$$\sigma_1^2(t) = \sum_{i=1}^t [i - \mu_1(t)]^2 \frac{P_i}{q_1(t)}, \quad \sigma_2^2(t) = \sum_{i=t+1}^N [i - \mu_2(t)]^2 \frac{P_i}{q_2(t)}. \quad (2.3)$$

Otsu's approach suggests that, an optimal threshold \hat{t} that separates the two classes from each other can be approximated by minimizing the sum of the weighted classes variances $\sigma_w^2(t)$, and Eq.(2.4) stated the proposed formula.

$$\hat{t} = \underset{t}{\operatorname{argmin}} \{ \sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t) \}. \quad (2.4)$$

One of the several problems that may impair performance in global approaches (more often in document scanning), is the variation in contrast and/or presence of illumination across the image. In such cases, binarization using a local threshold is more efficient [40]. That is, instead of a unique global threshold a dynamic or adaptive threshold is calculated for each pixel from its neighborhood. A popular adaptive binarization method has been proposed by Niblack [41]. In this approach, the mean μ and the standard deviation σ of intensities of a local squared window of neighboring pixels centered at the considered pixel are used to estimate a local threshold $t(x, y)$ at (x, y) as in Eq.(2.5)

$$t(x, y) = \mu(x, y) + k\sigma(x, y). \quad (2.5)$$

Where k is an adaptation parameter and its values are determined by the characteristics of the underling image. In case of document images with black text and relatively small number of foreground pixels compared to background pixels, small negative values are chosen, so that the threshold value bias is in favor of small intensity values.

FIG.2.4, shows examples of the results of the global based Otsu approach as well as the results of the local based approach of Niblack. As seen in this figure, local based approaches demonstrate a better performance (FIG.2.4 (c)) as compared to global approaches (FIG.2.4 (b)), when the document images (FIG.2.4 (a)) are degraded or contains considerable noisy pixels⁴. When, however, the contrast distribution in the input images is uniform (FIG.2.4 (d)), global based methods (FIG.2.4 (e)) outperform local based one (FIG.2.4 (f)).

⁴ There are many pixels that cannot be easily classified as foreground or background

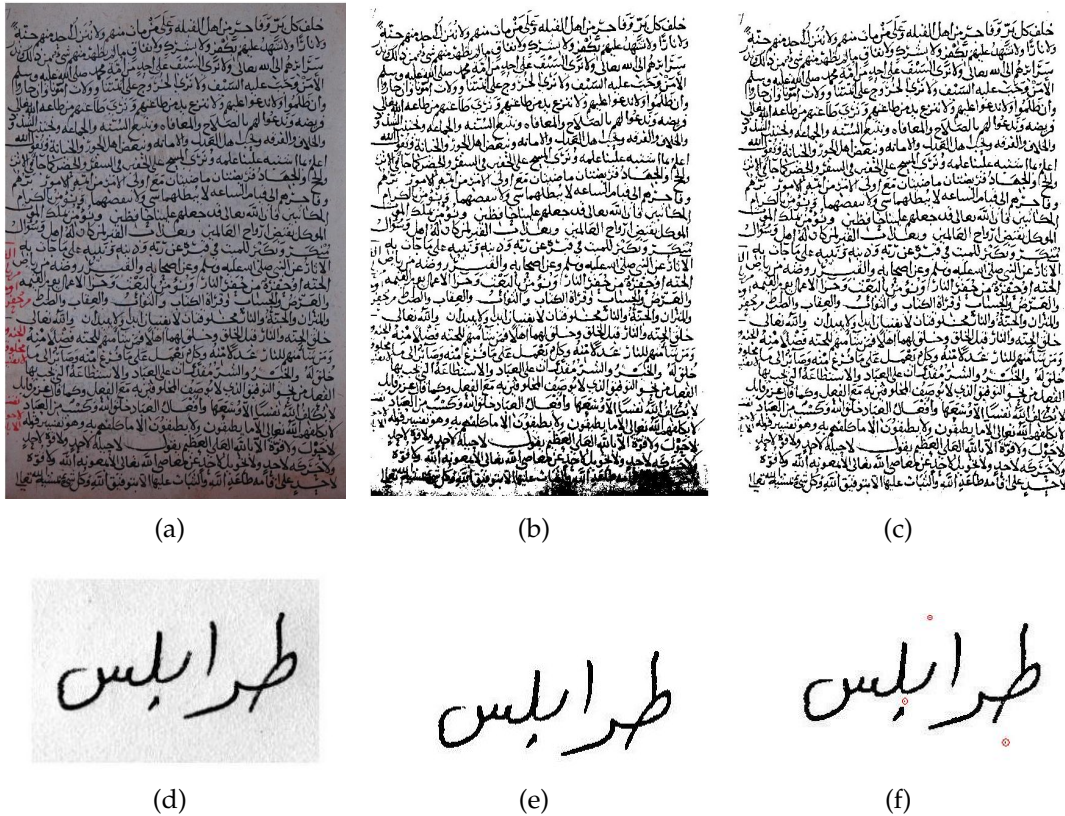


FIG. 2.4. Local vs. global binarization approaches: (a) A degraded manuscript image, (b) result of Otsu global based binarization approach, (c) result of Niblack local based binarization approach, (d) handwritten word image with a uniform background, (e) result of global approach, (f) result of local approach.

2.3.2 Thinning

Essentially, thinning is a pre-processing morphologic operation that is performed to delete a set of selected foreground pixels in binarized images [42]. Hence, to reduce the amount of information to be processed to the minimum necessary for processing. Furthermore, to ease extraction of structural features (e.g. cross points, end points, loops, etc.), thinning operation is applied on the document images. In general, all thinning algorithms usually fall under one of two main categories namely iterative and non-iterative. Intuitively, iterative algorithms are more expensive in terms of computation costs; however, they achieve far better results compared to non-iterative [43,44]. Regardless of the followed approach, any handwriting thinning algorithm should meet the following requirements:

- (i) All strokes in the output images should be of one-pixel width.

- (ii) The topological structure and connectivity of the original image should be preserved in the thinned version.
- (iii) The result should be stable, so upon repeating the algorithm on the same input image, the result is expected not to be changed.
- (iv) Thinned strokes should be an approximation (at least) of the medial-axis of original strokes.

2.3.2.1 Iterative Thinning Approaches

Iterative thinning can be performed using morphological operators like Erosion and Dilation, where the ultimate goal is to detect specific forms inside the binary image using what so-called Hit-Miss-Masks. According to the searched form, a Hit-Miss-Mask is defined and applied on every pixel in the binary image. A pixel is inserted in the result image only when all values of the mask and the original image are matching. The two most famous examples of these approaches are the popular Zhang-Suen algorithm [45] and the Stentiford algorithm [46]. In the former algorithm, image pixels subject to deletion are only tested against its eight neighbor pixels and against the result of the previous iteration. Such a trait allows to implement the algorithm in parallel across multiple processors, which in turn proved to be more efficient compared to sequential implementation. Stentiford thinning algorithm is another parallel template based algorithm, that starts by a preprocessing stage to minimize artifacts, removing isolated spots, and emphasizing acute angles. Then, a set of four 3×3 Hit-Miss-Masks are designed and used to scan the image from left to right and from top to bottom. A heuristic based procedure is then used to determine pixel candidates for deletion. FIG.2.5 shows the thinning results of both algorithms on a sample binary word image.

2.3.2.2 Non-iterative Thinning Approaches

In order to reduce the computational costs associated with the widespread iterative based algorithms, non-iterative and pixel independent thinning algorithms are designed and implemented [42]. The main idea behind most non-iterative algorithms is to accurately generate a median line of a stroke in only one pass rather than iterative raster scan of the entire image. Several approaches using techniques such

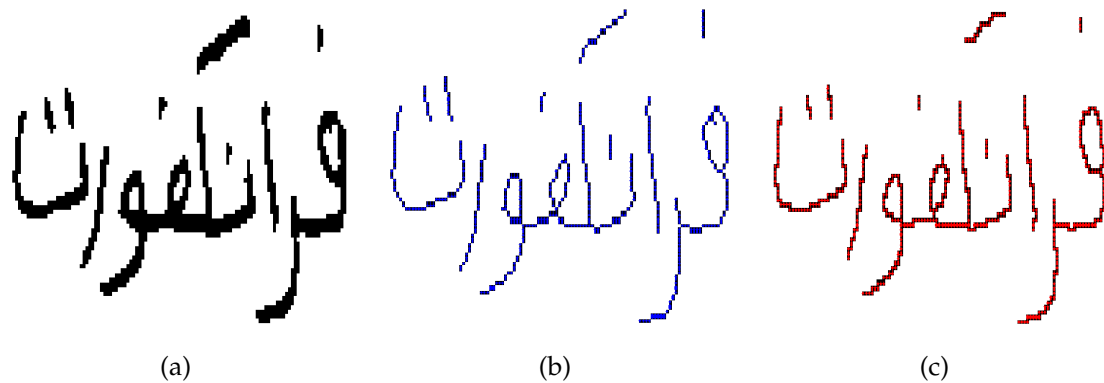


FIG. 2.5. Thinning of word image: (a) The source binary image, (b) result of Stentiford algorithm and, (c) results of Zhang-Suen algorithm.

as the medial axis transform, distance transform, or contour following are proposed to perform the task [44,47]. The non-iterative approaches are much faster than their iterative counterparts, but their result is less accurate.

2.3.3 Noise Suppression and Smoothing

Noise is usually the result of errors that may occur during the extraction process. Several approaches have been proposed to enhance the resolution and eliminate noise in gray images. Those approaches can be basically classified into linear and non-linear approaches [48,49]. A popular linear approach is the Gussian filter, which uses the 2D Gaussian kernel defined in Eq.(2.6).

$$g(x, y) = \frac{1}{2\pi\sigma^2} \cdot e^{-(x^2+y^2)/2\sigma^2}. \quad (2.6)$$

A 2D Gussian filtering operation can be performed using a two 1D Gussians, by first applying a 1D Gussian, then applying the same Gussian on the transposed result and finally transpose the last result. This property significantly speeds up performance and saves computational resources. A disadvantage of linear filtering, in general, is the blurring of edges. Non-linear approaches are designed to remove noise while preserving edges. As an example of such approaches is the median filter that works by running a window of neighbors through the image, entry by entry, replacing the central values by the neighbors median values [50,51]. In binary images, and as a result of binarization and normalization processes, noise may take the form of randomly distributed pixels being set to black or white, out-cropping

pixels that may protrude from the main stroke, small holes inside strokes, and/or small isolated pixel areas.

To eliminate such artifacts, a two fundamental template based morphological operations, namely erosion and dilation (or a combination of them) are usually used [52]. Given a binary image I that is to be eroded, an erosion operator also needs as input a matrix M of zeros and ones called the structuring element. M can be of any size smaller than the size of I and the arrangement of zeros and ones in M specify its shape and the needed effect. Furthermore, for each M , one of its pixels is considered as its center. To erode I , the structuring element M is centered upon each foreground pixel in the image, if any pixel in M has a background match in I , the pixel in I corresponding to the M centroid pixel is deleted. Thus, erosion shrinks the area of foreground components by removing the foreground boundary pixels and increasing the size of background holes within foreground area. On the other hand, since dilation has exactly the reverse effect of erosion, it is straightforwardly defined as an erosion operation of background rather than of foreground pixels. Dilation works by adding pixels into outer as well as inner boundaries resulting in smaller in-between gaps and filling up in-between holes.

In the literature symbols \ominus and \oplus are used to denote erosion and dilation operations respectively. Accordingly, the expression $\hat{I} = I \ominus M$, formulate the erosion of a binary image I using the structuring element M and \hat{I} is the returned eroded binary image. Likewise, \oplus is used to designate the dilation operator. Furthermore, erosion and dilation can be combined together to form more complex morphological operations. The so-called opening and closing are two of the most common used morphological operations which consist of a sequence of erosion and dilation [53,54]. Opening is the dilation of an eroded binary image I using the same structuring element M . It eliminates small foreground areas that are smaller in size than the structuring element while preserving the stroke width.

FIG. 2.6(b) shows that the open operation completely removes noisy pixels (being shown circled in yellow on the source image). Closing, on the other hand, is constructed by applying first dilation on the image I followed by an erosion. FIG. 2.6(c) demonstrates the capability of the closing operation in bridging gaps, solidify boundaries, and filling small holes (being shown circled in red on the source image). The mathematical notations of opening and closing are shown in Eq.(2.7)

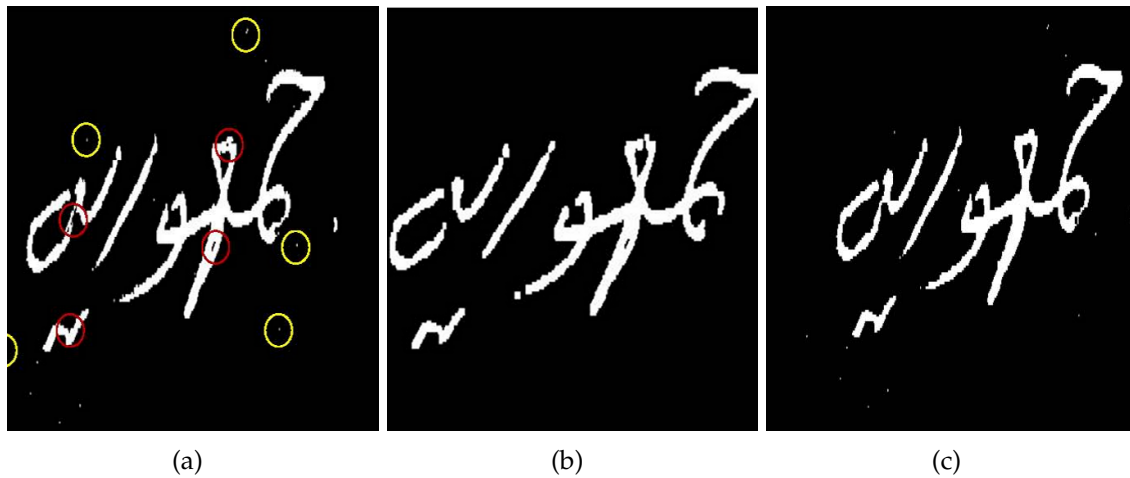


FIG. 2.6. Word image preprocessing using closing and opening operations: (a) The noisy source image, (b) result of opening operation and, (c) result of closing operation.

and Eq.(2.8), respectively, as follows.

$$I \circ M = (I \ominus M) \oplus M \quad \text{Opening,} \quad (2.7)$$

$$I \bullet M = (I \oplus M) \ominus M \quad \text{Closing.} \quad (2.8)$$

2.3.4 Baseline Estimation and Skew Correction

In horizontally written scripts (e.g. Latin, Arabic, etc.), words usually written around a horizontal imaginary line called baseline (see the blue line in FIG. 2.7). As a result of the scanning process or the personal writing style, the image of the handwritten word or the entire page of handwriting may appear rotated with respect to the horizontal (e.g., the baseline is diverged from the x-axis), this artifact is often referred to as the text/word skew. Accurately detecting the baseline and subsequently correcting the page or the word skew are proven of vital importance for the performance of the following phases of character recognition process [55].

In case of Arabic alphabet based scripts, characters are usually rising or descending from a base stroke parallel to the horizontal, such a fact affects positively the detection of baseline, in a way, that even simple baseline detection methods may yield quit satisfactory results comparing to apply the same methods on other scripts [20]. As an example, FIG.2.7 (a) illustrates a skewed handwritten Arabic

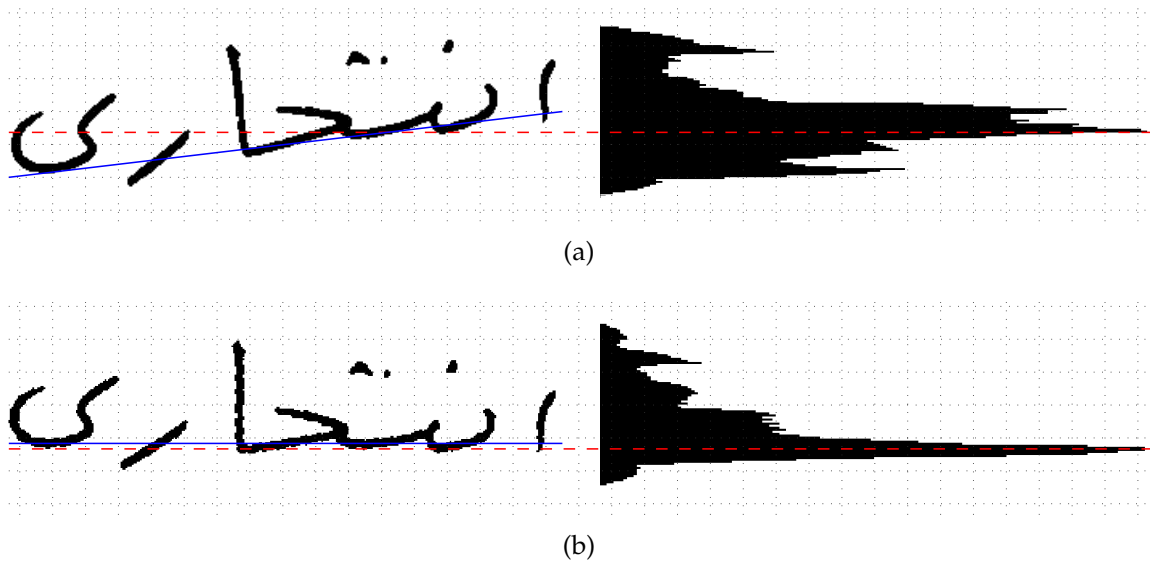


FIG. 2.7. Skew correction based on the image profile: (a) A skewed handwritten word and its corresponding projection profile, (b) The skew free version and its corresponding projection profile. The blue line is the handwriting base line and the red line is an imaginary line intersecting the peak of the profile.

word and its corresponding horizontal profile histogram, where the horizontal line that is passing through the histogram peak (the red dashed line in FIG.2.7 (a)) is not in parallel with its downward sloping baseline. Broadly speaking, and regardless of the used writing script, various baseline detection techniques have been reported in the literature each has its pros and cons. In the following, we briefly describe the basic of some of the most commonly used techniques.

2.3.4.1 Profile based Methods

Projection profile based is the simplest principle for baseline estimation and skew correction [56]. The technique, typically, uses a projection function that projects the word or the line of text pixels along parallel lines into an accumulator array. The accumulator array is partitioned into fixed height bins. The angle of projection is, usually, varied within a limited angular search interval (e.g., ± 45), and a projection is done for each angle. This process creates a sequence of accumulator arrays corresponding to the search angles. Then an optimization function is used to calculate the alignment premium for each accumulator array. The angle corresponding to the accumulator array that maximizes the alignment premium is then given as the

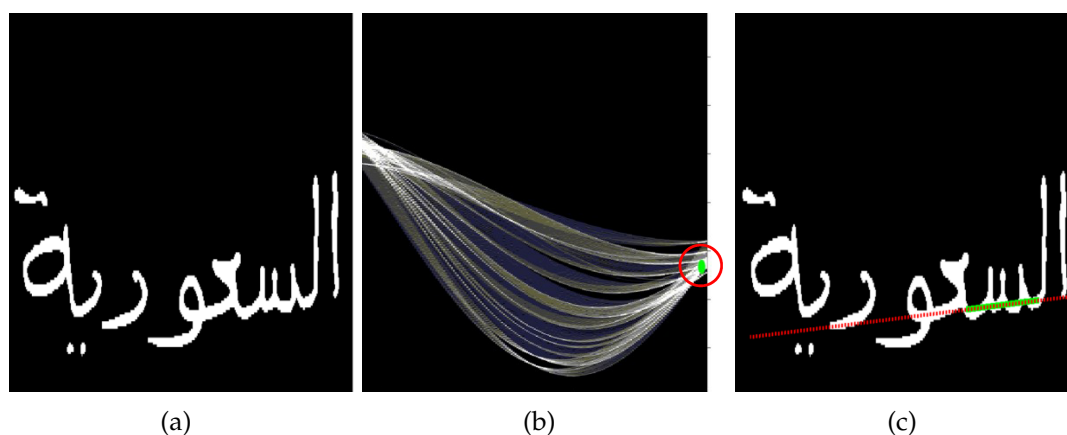


FIG. 2.8. Skew correction based on HT: (a) a skewed handwritten word, (b) the corresponding Hough space, (c) the estimated baseline.

skew estimation [1,57]. After applying a profile based correction on the handwritten word in FIG. 2.7 (a), FIG. 2.7 (b) shows the skew free handwritten word and the equivalent profile histogram. Note that the horizontal line across the profile histogram peak (i.e., the red dashed line) is parallel to the word baseline. Even though, profile histogram based methods perform well better on lines of text and on elongated words, however, they are inappropriate in case of short and isolated words where other methods showed better performance s [58,59].

2.3.4.2 Hough Transform based Methods

Hough transform (HT) based methods are another popular alternative to detect the word's baseline and correct its skewness. Basically, this technique is employed to detect simple shapes like a line, circle or ellipse. The idea is to parameterize the candidate shape first, then perform a voting procedure among image data points over the candidate shape [60]. The main advantages of HT are, firstly, it tolerates gaps that may occur inside a word (in case of Arabic script) or along a line of text among words; secondly, it is robust to noise and partial occlusion. According to the above-mentioned definition of the baseline and by using HT, the baseline estimation process can be seen as equivalent to the process of finding the line that intersects the maximum numbers of pixels in the image. Instead of looking for data points that constitute a line in the image space, HT uses the straight line parameters, i.e., the slope m and the intercept b to discover candidate lines. Usually, a straight line is annotated as $y = bx + m$, which is characterized as (m, b) point in Cartesian space,

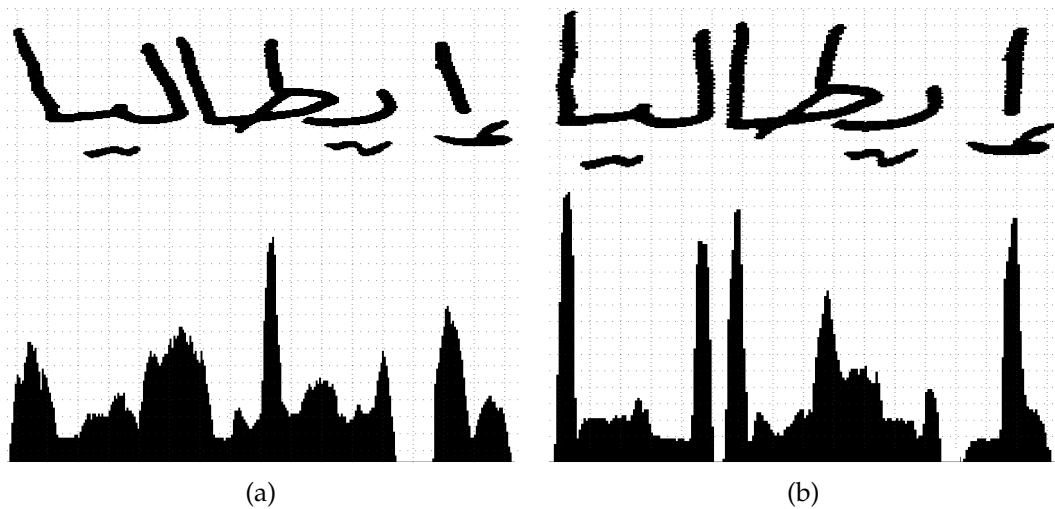


FIG. 2.9. Slant correction: (a) A slanted handwritten word and its profile histogram, (b) the slant-free version and the corresponding profile histogram.

or preferably as (ρ, θ) point in the polar parameters space. Hence, every point in the image space is corresponding to a line in the parameters space and vice versa. The point $(\hat{\rho}, \hat{\theta})$ where the maximum number of lines intersect is ultimately considered as the baseline. Drawbacks associated with HT are mainly the high computational cost of the voting scheme, and the relatively high-sensitivity to outlier pixels [61,62]. The baseline of the word in FIG. 2.8 (a) is identified first as a point (circled in red) in the Hough space in FIG.2.8 (b), which is corresponding to the longest line segment in the image highlighted in green in FIG. 2.8 (c).

2.3.5 Slant Correction

Slant angle is the angle made by the vertical strokes with the absolute vertical direction. In order to reduce variability within handwritten character classes, it is necessary to normalize slant variations [34]. Intuitively, slant correction improves accuracy in case of explicit segmentation based recognition approaches, by increasing spaces between vertical strokes, which in turn significantly eases the challenging segmentation process. As for implicit segmentation based approaches, where features are extracted from a vertical projection histogram, slant correction is found to be a necessary procedure to avoid the overlap of characters with their direct neighbors [63]. FIG.2.9 (a) and (b) show the slanted and the slant-free version of a handwritten word, FIG.2.9 (c) and (d) depict their vertical profiles, respectively.

To correct the slant, shear transform is usually used, which is defined in terms of an estimated slant angle α and the (x, y) pixels coordinates. Each pixel in the image space is horizontally mapped to a new coordinates (\acute{x}, \acute{y}) in the slant-free image space according to Eq.(2.9)

$$\acute{x} = x - y \cdot \tan(\alpha), \quad \acute{y} = y \quad (2.9)$$

Often slant angle estimation is performed in two steps. Firstly, and due to the fact that slant is defined as the angle between the non-horizontal strokes and the vertical, the non-horizontal strokes are extracted. Secondly, the angle between each stroke and the vertical axis is calculated [43]. A popular approach for non-horizontal strokes extraction is detailed in [8]. It starts by calculating the horizontal gradient image $\acute{I} = \frac{\partial I}{\partial x}$, then proceeds with binarizing and morphologically processing the result \acute{I} to get rid of small pixel areas and outliers. Estimation of slant angle is then performed using the output image of the non-horizontal strokes only. The reasons behind choosing the horizontal gradient image for calculation are; firstly, vertical strokes will be emphasized at the expense of horizontal ones. Secondly, computational cost will be reduced, since relatively fewer pixels need to be processed.

For the estimation of the slant angle, profile based methods are most widely adopted due to their simplicity [59]. The basic idea of these methods is to shear the image at a discrete number of angles, usually in the range $[\pm 45]$ around the vertical direction. Then, at each shearing angle, the vertical projection histogram H is calculated, as follows,

$$H(\acute{x}_l; \alpha) = \sum_{k=1}^M \acute{i}(\acute{x}_l, \acute{y}_k) \quad (2.10)$$

where M is the number of rows in the images.

Within each histogram, a variation analysis between consecutive bins is performed according to the following equation:

$$A(\alpha) = \sum_{l=0}^{N-1} [H(\acute{x}_l; \alpha) - H(\acute{x}_{l+1}; \alpha)]^2 \quad (2.11)$$

where N is the number of histogram bins.

Finally, the angle corresponding to the histogram with maximum variation is considered the estimated slant angle:

$$\acute{\alpha} = \arg \max_{\alpha} V(\alpha). \quad (2.12)$$

Alternatively, chain code can be used to estimate the average slant angle, which tends to be less expensive compared to profile based approaches [64]. A straightforward method proposed in [2], in which a 3-bins histograms of chain code is calculated. Each bin corresponding to one of three different angles, namely 45° , 90° , and 135° . Additionally, let n_1, n_2 , and n_3 be the number of chain code elements at the different angles, as illustrated in FIG. 2.10. The slant angle is then calculated as follows,

$$\alpha = \tan^{-1}\left(\frac{n_1 - n_3}{n_1 + n_2 + n_3}\right) \quad (2.13)$$

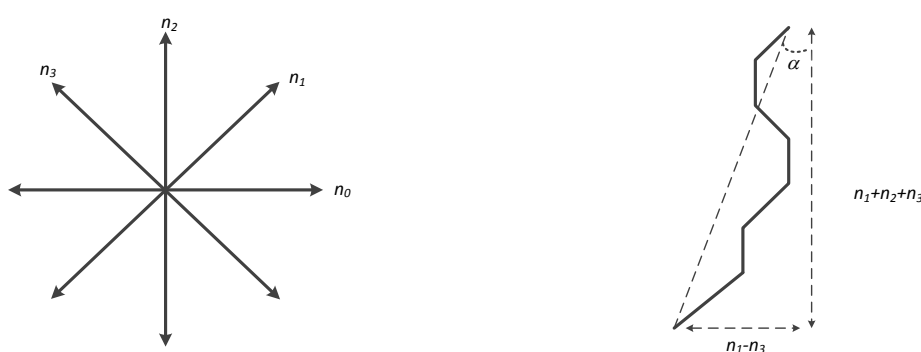


FIG. 2.10. Average slant angle estimation using chain code sequence [2].

2.3.6 Image Size Normalization for Handwriting

Size normalization of the handwriting images is considered to be of crucial importance for the subsequent feature extraction process. It is used to reduce the within class variation and hence enhances the recognition results. Its basic idea is to map the handwriting images (of various size) into a squared standard plane of size $N \times N$, where $N = 32$ or $N = 64$ are the most common used values. Size normalization algorithms should preserve the aspect ratio of the input image in order to alleviate distortion in the output image. According to [1, 65], the input image aspect ratio R_1 and the normalized image aspect ratio R_2 are defined according to Eq.(2.14).

$$R_1 = \frac{\min(W_1, H_1)}{\max(W_1, H_1)}, \quad R_2 = \frac{\min(W_2, H_2)}{\max(W_2, H_2)} \quad (2.14)$$

where the pairs (W_1, H_1) and (W_2, H_2) are the width and height of the input image and the normalized image, respectively. In general, to copy a pixel value from the

input plane to the output plane, the corresponding coordinates in the output plane must be first calculated. Thus, forward mapping or backward mapping are usually employed to achieve this task. For each pixel in the input plane, Forward mapping is used to compute a new coordinates in the output plane and copies the pixel value to it. Backward mapping, on the other hand, iterates over each pixel in the output plane and calculates the position in the input plane from which the new pixel value must be drawn.

A common size normalization technique is the aspect ratio adaptive normalization (ARAN) [66], in which the aspect ratio R_2 of the output image is adaptively computed using the aspect ratio of the input image R_1 . For the mapping of R_1 to R_2 , multiple linear and non-linear mapping functions (e.g., $R_2 = R_1$, $R_2 = \sqrt{R_1}$, or $R_2 = \sqrt{\sin(\frac{\pi}{2}R_1)}$) are proposed. In ARAN, the normalized image is first copied into a tentative plane of size $W_2 \times H_2$, then the tentative plane is shifted to overlap the standard plane by aligning boundaries or centroid.

Alternatively, image moment based approach is used for the image size normalization task. It starts by calculating the centroid (x_c, y_c) of the original image $I(x, y)$ using to Eq. (2.15)

$$x_c = m_{10}/m_{00}, \quad y_c = m_{01}/m_{00}, \quad (2.15)$$

where $m_{pq} = \sum_x \sum_y x^p y^q I(x, y)$ is denoting any geometric moments. Further, the pair (W_1, H_1) is given as follows

$$W_1 = a\sqrt{\mu_{20}/m_{00}}, \quad H_1 = a\sqrt{\mu_{02}/m_{00}}, \quad (2.16)$$

where $\mu_{pq} = \sum_x \sum_y (x - x_c)^p (y - y_c)^q I(x, y)$ denotes the central moments and a denotes a coefficient that experimentally set to $a = 4$. Given W_1 and H_1 , R_1 is calculated according to Eq.(2.14), and R_2 is computed in terms of R_1 by using one of the aforementioned mapping functions. Finally, forward or backward mapping is applied on pixels coordinates and the centroid of the normalized image is aligned to the center of the standard plane.

2.4 Segmentation

Segmentation is defined as the process of identifying the borders of basic elements in the signal [32]. In document analysis, segmentation is aimed at extracting lines

of text from a page image, obtaining words from a line of text, spotting characters' borders in a word, or even identify a stroke in a character. It is considered as a prerequisite process for subsequent features extraction and classification processes, by many automatic handwriting recognition systems. Typically, systems are categorized based on the type of the adopted segmentation methodology. There are two different segmentation methodologies, i.e., explicit and implicit segmentation. Before feature extraction, explicit segmentation breaks the handwritten words or strings into a series of segments representing their constituent characters. Then, features are extracted from each segment, and eventually recognized. Instead of trying to explicitly segment each string into a number of segments, assuming that each segment can be later related to a character class, in the implicit based approach, segmentation and recognition are both integrated in one process. Moreover, each pixel column is considered as a potential candidate cut. Thus, by performing an over-segmentation, we end up with several segmentation hypotheses, the best of them is chosen based on the recognition results [20,67]. In the following sections, and instead of addressing the broad handwriting segmentation problems (e.g., pages into line of text, and line of text into words), we will focus on the problem of handwritten words segmentation.

2.4.1 Explicit Segmentation

As previously stated, the ultimate objective of explicit segmentation is to split the underlying image into segments that can be assigned in one-to-one classification to pre-built models for letters. In addition to the difficulty of the explicit segmentation task, the main drawback of recognition using explicit segmentation is the inability of employing the contextual information (e.g., related lexicons, the language model) to improve the final recognition results. Methods addressing this problem can be conceptually divided into two distinct categories: vertical projection analysis based methods, and contour analysis based methods [68]. Under the former category, falls all approaches that perform analysis of the vertical projection of the handwritten word images. These approaches assume that valleys in the projection are most likely corresponding to border areas between letters. Heuristic rules are usually used to elect the optimal threshold point in the valley that is most likely separate two letters.

In general, such approaches require very well written samples to provide satisfactory results. In case of Arabic alphabet based scripts, these approaches show weak performances [8], since valleys appear inside the main stroke of many letters resulting of splitting the letter into two or more segments. Approaches fall into the second category are based on the analysis of the handwritten word's contour [69].

By tracing the contour, all points laying between subsequent (spatial) maxima and minima (or vice versa) are marked as potential segmentation points. A set of heuristics formulated over topological features are then used to choose the most probable cutting points. Prior estimation of the numbers of letters in the word, skew correction, as well as normalization of the slant, are all considered as main factors that affect the performance. For the sake of reliable recognition results, explicit segmentation algorithms still need more efforts to overcome several challenging issues. A less demanding segmentation strategy to follow would be the implicit segmentation which will be detailed in the following section.

2.4.2 Implicit Segmentation

Implicit segmentation is usually followed to avoid two types of obstacles. Firstly, the complexity and the error-prone nature of the explicit segmentation, and secondly, the dependency on application areas specific lexicons, which ultimately limit the flexibility of the recognition systems. To evade the prior and definitive segmentation decisions, and in meantime retain the flexibility of explicit segmentation based approaches, segmentation and recognition processes should be fully integrated [70, 71]. Segmentation methods adopting such a paradigm are called implicit segmentation based recognition or recognition-based segmentation. A common and simple approach for implicit segmentation is the sliding-window based technique.

The basic idea here is to slide a window of a given width along the text line from left to right or vice versa, based on the written script. By varying the window width and the sliding step size, multiple sequences of temporal segmentation are produced. Then, sequences are evaluated based on the recognition results. Where recognition is typically carried out using the well known hidden Markov models, due to their efficiency in modeling of sequences.

2.5 Generative and Discriminative Recognition

Typically, the objective of the recognition process is to predict the most probable value of a vector of labels \mathbf{y} for which the value of a vector \mathbf{x} of input features should be assigned. Basically, offline handwriting recognition problem is a classification problem and there are many different modeling approaches for the solution. These approaches can be categorized into two main categories, namely, generative and discriminative [72]. FIG.2.11, illustrates the fundamental computation characteristics of both categories.

According to the first category, we start first by calculating the joint distribution $p(\mathbf{x}, \mathbf{y})$ and then a new value of the observed data \mathbf{x} is assigned to the most likely label \mathbf{y} by estimating the conditional probability $p(\mathbf{y}|\mathbf{x})$ using the popular Bayes rule. Recognition approaches that fall under this category are called generative approaches, as modeling the joint probability allows the generation of synthetic instance of vector \mathbf{x} . Generative recognition models [73,74], usually exhibit several advantages such as:

- (i) Their capability of handling missing data, since they attempt to model the distribution of the observed data.
- (ii) Instead of adhering to a complete supervised learning approach, generative based models can switch to a semi-supervised learning strategy such that labeled training data and unlabeled training data can be exploited in the training process.
- (iii) Classes are modeled individually, that facilitates the process of updating the model in case of detecting any new changes in the underlying distribution.
- (iv) Given that they are built by modeling the distribution of the data, generative models tend to be less sensitive to outliers and in the same time relatively robust against the over-fitting problem⁵.

⁵Over-fitting occurs when the model describes perfectly the training data (including random error or noise) and fails to generalize. Models with too many parameters relative to the number of observations, are more prone to such a problem.

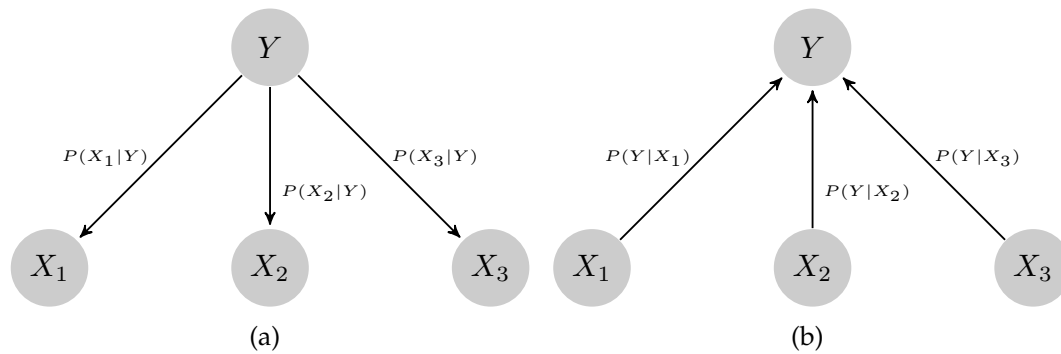


FIG. 2.11. Generative vs Discriminative models: (a) The basic probabilistic structure of generative models, (b) The basic probabilistic structure of discriminative models.

The major disadvantage of generative models, however, is their tendency to focus on modeling the data distributions rather than directly discovering the boundaries that separate classes within the data, thus further computation costs are necessary. Moreover, the recognition performance of generative models is tightly related to the data actual distribution that is proved to be difficult to be accurately captured in any model. Finally, and as common characteristic among generative models, it is noticed that the misclassification rate is relatively proportional to the growth in data size. Some common examples of the most popular generative based models are Naive Bayes, Gaussians Mixtures, and HMMs [75]. Approaches falling under the discriminative category can be further categorized into two categories such as, probabilistic based and non-probabilistic based. According to the probabilistic based approaches (e.g., Logistic regression, and CRFs, and instead of attempting to model the entire data (like in generative models), the focus here is to directly model the posterior probability $p(y|x)$ using a parametric model, where the parameter values are learned from the training data. Non-probabilistic discriminative approaches (e.g., Support vector machines (SVM), and Traditional neural networks (ANN)), on the other hand, directly learn the mappings from the training data points to the class labels.

In general, the strength of the discriminative based approaches stems from the fact that such models are solely focusing on the task of modeling the boundaries between data points, which boost the performance of discriminative based classifiers compared to their generative counterparts. Furthermore, it is stated in [76] that discriminative classifiers are outperform generative ones on large training sets. Roughly speaking, even though generative and discriminative approaches

have different characteristics, however, it is reported in several published works (e.g., logistic regression and Naive Bayes) that they have complementary advantages and disadvantages in performance. One of the main contribution in this thesis is to investigate the two alternative learning methodologies on the offline Arabic handwriting problem, by comparing the performance of three different candidates namely HMMs for generative model, CRFs for fully-observed probabilistic discriminative model, and Hidden-state Conditional Random Fields (HCRFs) for probabilistic discriminative based model with hidden states. In the following sections the theoretical bases of the generative HMMs as well as the discriminative CRFs will be discussed.

2.5.1 Hidden Markov Models (HMMs)

HMMs is the most widely used statistical based classifier for observation sequences of variable length. Inspired by the successful application of HMMs in the field of automatic speech recognition, nowadays, many approaches propose HMMs for the problem of offline handwriting recognition. In coming subsections, we will detail the most important HMMs modeling aspects as well as the fundamental application concepts. Furthermore, the three model's basic tasks, namely evaluation, decoding, and training will be explained. Furthermore, it is worth pointing out that our notations are mainly inspired by the works presented in references [43,77,78].

2.5.1.1 HMMs Basic Definitions

A discrete statistical process \mathbf{q} is a sequence of random variables, where each may have any value from an infinite set of states $S = \{s_1, s_2, \dots, s_N\}$:

$$\mathbf{q} = q_1, q_2, \dots, q_t, \quad q_t \in S. \quad (2.17)$$

Where the parameter t of the statistical process may represent a time point. If the process \mathbf{q} satisfies the Markov property, which states that the value of the current state $q_t = s_i$ is depending on the value of the predecessor state $q_{t-1} = s_j$ only,

$$P(q_t = s_j | q_{t-1} = s_i, \dots, q_0 = s_k) = P(q_t = s_j | q_{t-1} = s_i), \quad (2.18)$$

and if \mathbf{q} is a stationary process which implies the process statistical properties do not vary with time, then we call \mathbf{q} a homogeneous Markov chain.

Additionally, the conditional probabilities $P(q_t = s_j | q_{t-1} = s_i)$ are termed transitional probabilities since they represent the probability that a random variable q in time t takes the value s_j , where in a previous time $t - 1$ its value was s_i .

For each homogeneous Markov chain, a quadratic $N \times N$ matrix of transition probabilities is constructed with the following equation:

$$\mathbf{A} = [a_{ij}], \text{ where } a_{ij} = P(q_t = s_j | q_{t-1} = s_i), a_{ij} \geq 0, \text{ and } \sum_{j=1}^N a_{ij} = 1. \quad (2.19)$$

To initialize the Markov chain, a vector $\boldsymbol{\pi}$ containing the initial probabilities is introduced:

$$\boldsymbol{\pi} = (\pi_1, \dots, \pi_N), \quad \pi_i = P(q_1 = s_i), \text{ where } \sum_{i=1}^N \pi_i = 1. \quad (2.20)$$

Parameters $\boldsymbol{\pi}$ and \mathbf{A} fully capture the Markov chain characteristics which is regarded as the first phase of a two-phase hidden Markov models classifier. The second phase of an HMMs, is a process of generating emission probabilities in accordance to the current state. In an HMMs only the sequence of emissions probabilities is observable, while the sequence of states the model goes through to generate the emissions is hidden. Note that the term "hidden" refers to this peculiarity. Usually an emission sequence represents either symbols of a finite set, of vectors of D -dimensional vector space. Usually an emission sequence $\mathbf{O} = o_1, \dots, o_T$ represents either symbols of a finite categorical set $\nu_k \in \{\nu_1, \dots, \nu_D\}$, or vectors $\mathbf{x} \in \mathbb{R}^D$ of D -dimensional vector space. In the former case HMMs called discrete HMMs, and the corresponding emission probabilities are arranged in a $N \times D$ matrix:

$$\mathbf{B} = [b_{jk}], \text{ where } b_{jk} = P(o_t = \nu_k | q_t = s_j), b_{jk} \geq 0, \text{ and } \sum_{k=1}^D b_{jk} = 1. \quad (2.21)$$

For the second case, typical, emission values are generated from a continuous density distribution and the corresponding emission probabilities are estimated, as follows,

$$\mathbf{B} = [b_j], \text{ where } b_j(\mathbf{x}) = P(o_t = \mathbf{x} | q_t = s_j), b_j \geq 0, \text{ and } \int_{\mathbb{R}^D} b_j(\mathbf{x}) d\mathbf{x} = 1. \quad (2.22)$$

From the above discussion and for more convenience, HMMs is typically defined compactly in term of its basic probability measures, i.e. \mathbf{A} , \mathbf{B} , and $\boldsymbol{\pi}$ as :

$$\boldsymbol{\lambda} = (\boldsymbol{\pi}, \mathbf{A}, \mathbf{B}) \quad (2.23)$$

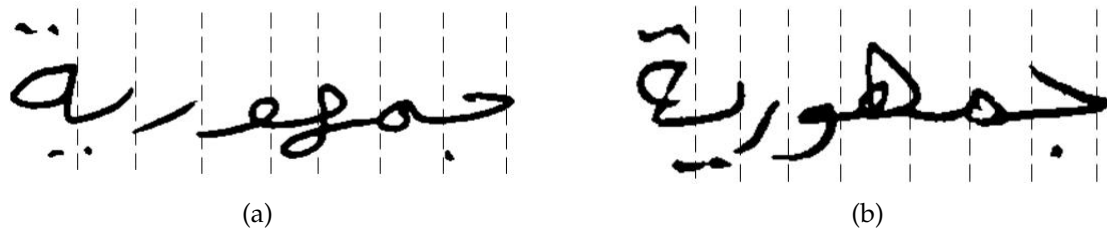


FIG. 2.12. Variations in length and handwriting of two samples of the same words. Feature vectors are usually extracted from multiple vertical stripes along the word image.

2.5.1.2 HMMs Modeling for Handwriting

For handwriting recognition, FIG.2.12 illustrates the variation of two handwritten samples of the same word, where the font as well as the distances between letters are different. Consequently, even though both samples are of the same word, the corresponding feature vectors extracted will be of different lengths. To accommodate and model such variability, FIG. 2.13 (as an example) suggests an HMM model that can sufficiently model the word samples. The depicted HMM model composed of seven states equivalent to the number of letters in the "to be modeled" word. According to this model, each state has a two different transition probabilities one to represent a self-transition (return to the current state), and the other representing transition to the next state. Accordingly, the variability in length of handwritten words (as a result of variations in lengths of spaces and KASHIDA between letters) can be modeled through state self-transitions, and different letters that constitute a word will be modeled using the transitions to the next state.

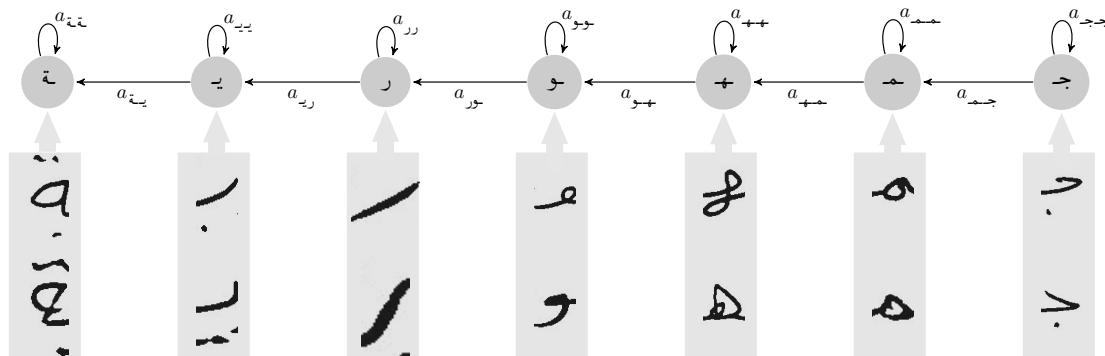


FIG. 2.13. A possible modeling of the handwritten word (two images of the same word) in FIG. 2.12 using HMMs.

2.5.1.3 HMMs Topologies

A very important issue in building an HMM based model, is the choice of the model topology, i.e., the way in which model states are connected to each other. As a consequence to the choice of the topology, the shape of the transition matrix \mathbf{A} will be formed accordingly. For HMMs there are three popular topologies, in the first topology, the model states are fully connected. Where every state of the model can be reached from every other state. Such topology is called ergodic, and offers the highest degree of model flexibility, where each element a_{ij} in the corresponding transition matrix \mathbf{A} is nonzero value.

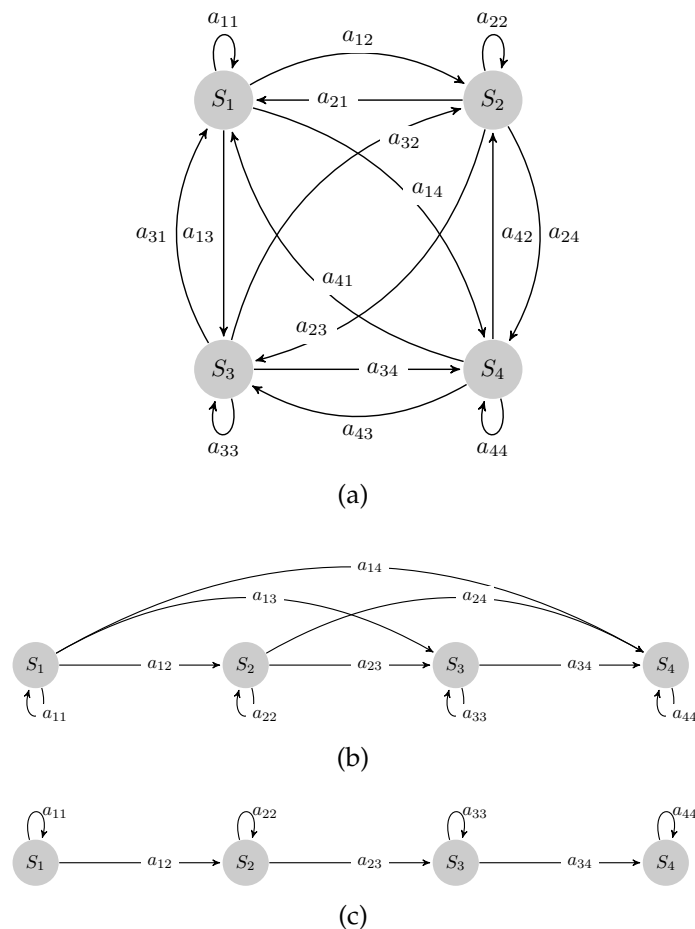


FIG. 2.14. Schematic representation of different HMMs topologies: (a) connection structure of an ergodic model, (b) Left-to-Right (Bakis) model, and (c) linear (banded) model.

An example for a fully-connected topology (ergodic), an HMMs model of $N=4$ states is illustrated in FIG. 2.14 (a), and the equivalent states transition matrix is

given by equation Eq. (2.24).

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \quad (2.24)$$

A less flexible modeling topology is the so called Left-Right or Bakis model. According to this topology, models can only have transitions to the following states as well as to the state itself. The topology is shown in FIG. 2.14 (b) and the equivalent transition matrix is given in Eq.(2.25)

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{pmatrix} \quad (2.25)$$

The last and the simplest model topology is the so called linear or left-to-right banded model; this topology stated that models have only transitions to the respective next state and to the current state. FIG. 2.14 (c) shows the linear model and Eq. (2.26) gives the topology corresponding transition matrix.

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & 0 & 0 \\ 0 & a_{22} & a_{23} & 0 \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{pmatrix} \quad (2.26)$$

To sum up, it is worth mentioning that the choice of the model topology is a trade-off issue between the model flexibility and the model calculation costs. Since the degree of the model connectivity is proportional to the number of state transitions, implying expensive training and decoding computation cost.

2.5.1.4 The Three Classic HMMs Problems

When applied to real-world classification problem tasks, there are three different kinds of problems that may encounter an HMMs based model and hence should be dealt with. Those problems are :

- (i) **Evaluation problem:** Given the observation sequence \mathcal{O} and the model parameter λ , how probable that \mathcal{O} is produced by λ or $P(\mathcal{O}|\lambda)$?

Solving this problem makes it possible to pick the model that best match \mathcal{O} , or even better, scoring multiple models according to their degrees of matching to \mathcal{O} .

- (ii) **Decoding problem:** Given the observation sequence \mathcal{O} and the model parameter λ , how to determine the best path through λ that generates $\mathcal{O} = \{o_1, o_2, \dots, o_T\}$ with maximum likelihood?

Possible application could be to learn the model structure, in order to estimate the optimal state sequences for handwritten word in a handwriting recognition system.

- (iii) **Training problem:** Given the observation sequence \mathcal{O} , how to adjust and re-estimate the model parameters $(\pi, \mathbf{A}, \mathbf{B})$ to maximize $P(\mathcal{O}|\lambda)$?

In this case, \mathcal{O} is typically called training sequence where it is used to optimize the model parameters to best describe the observations generation process, i.e., modeling the real-world classification problems.

In the following, we will discuss the proposed solutions of the three problems.

Evaluation Problem

As explained above, the classification decision is usually a function of the conditional probability $P(\mathcal{O}|\lambda)$. Using dynamic programming, the calculation of this probability can be performed efficiently, i.e., costs is kept linear in the length of the observation sequence. For the calculation, we either use the so called forward probabilities.

$$\alpha_t(j) = P(o_1, o_2, \dots, o_t, q_t = s_j | \lambda) \quad (2.27)$$

or alternatively, the so called backward probabilities

$$\beta_t(j) = P(o_{t+1}, o_2, \dots, o_T | q_t = s_j, \lambda) \quad (2.28)$$

The forward probability $\alpha_t(j)$ define the probability of observing the partial sequence o_1, o_2, \dots, o_t at time t in state s_j . Analogous to the forward probability, but just in the other direction, backward probability $\beta_t(j)$ denotes the probability of

observing the partial sequence o_{t+1}, o_2, \dots, o_T at time $t + 1$, where the current time and state are t and s_j respectively. As a result of the aforementioned definitions, $P(\mathbf{O}|\boldsymbol{\lambda})$ can be estimated as :

$$P(\mathbf{O}|\boldsymbol{\lambda}) = \sum_{i=1}^N \alpha_t(i) \beta_t(i). \quad (2.29)$$

Based on the Markov property of a stochastic process (mentioned in Section 2.5.1.1), both $\alpha_t(j)$ and $\beta_t(j)$ can be recursively estimated using their corresponding previous values. To start the calculation process the forward variable α is initialized as follows:

$$\alpha_1(i) = \pi_i b_j(o_1), \quad (2.30)$$

while backward variable β is initialized as:

$$\beta_T(j) = 1 \quad (2.31)$$

After initialization, a recursive calculation process is then performed to estimate the forward probabilities for the subsequent time points according to:

$$\alpha_{t+1}(j) = \sum_{i=1}^N [\alpha_t(i) \cdot a_{ij}] \cdot b_j(o_{t+1}), \quad 1 \leq t \leq T - 1. \quad (2.32)$$

Similarly, the calculations of the backward probabilities are recursively performed according to the following equation:

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad T - 1 \geq t \geq 1. \quad (2.33)$$

Eventually, the required conditional probability can be estimated either by using the forward probabilities,

$$P(\mathbf{O}|\boldsymbol{\lambda}) = \sum_{i=1}^N \alpha_T(i) \quad (2.34)$$

or alternatively, by using the backward probabilities,

$$P(\mathbf{O}|\boldsymbol{\lambda}) = \sum_{j=1}^N \pi_j b_j(o_1) \beta_1(j). \quad (2.35)$$

Both calculations are equivalent; the only difference is in the direction of the recursion process.

Decoding Problem

The objective of the decoding task is to identify the HMMs internal path with the heights probability, i.e., estimating the state sequence that maximizes the probability,

$$P(q|\mathbf{O}, \lambda) = \frac{P(\mathbf{O}, q|\lambda)}{P(\mathbf{O}|\lambda)} \quad (2.36)$$

Given the model λ and an observation sequence \mathbf{O} , and since the denominator in the above formula is independent of the searched optimal state sequence q^* , therefore q^* can be formulated as following:

$$q^* = \arg \max_q P(\mathbf{O}, q|\lambda) \quad (2.37)$$

Similar to the estimation of the forward probabilities, dynamic programming can also be employed here for the estimation of the optimal state sequence, but instead of summing over transitions of states, we compute the maximum. To denote the maximum probability of generating the observation sequence o_1, o_2, \dots, o_t along the optimal path at time t and state s_j , the quantity $\vartheta_t(i)$ is defined as:

$$\vartheta_t(i) = \max_{q_1, \dots, q_{t-1}} P(o_1, \dots, o_t, q_1, \dots, q_{t-1}, q_t = s_i | \lambda) \quad (2.38)$$

Additionally, to save all arguments that maximized Eq.(2.38) for each t and i , a backtracking matrix $\psi(j)$ is constructed, from which the optimal state sequence will be retrieved. Typically, the complete solution for this problem is a dynamic programming and known as Viterbi algorithm. The main steps of the Viterbi algorithm can be illustrated as follows,

- (i) **Initialization:** At the beginning of the algorithm, i.e. $t=1$, $\vartheta_t(j)$ and $\psi(j)$ are assigned initial values as follows,

$$\vartheta_1(j) = \pi_j b_j(o_1) \quad \text{and} \quad \psi_1(j) = 0, \quad \forall j : j = 1, \dots, N$$

- (ii) **Recursion:** For $\forall t : t = 1, \dots, T - 1$ and $\forall j : j = 1, \dots, N$, the probability for the next time point is recursively calculated by the following formula,

$$\vartheta_{t+1}(j) = \max_i \{ \vartheta_t(i) a_{ij} \} b_j(o_{t+1}) \quad \text{and} \quad \psi_{t+1}(j) = \arg \max_i \vartheta_t(i) a_{ij}$$

- (iii) **Termination:** All computation are achieved upon the estimation of

$$P(\mathbf{O}, q^* | \lambda) = \max_i \vartheta_T(i) \quad \text{and} \quad q_T^* = \arg \max_i \vartheta_T(i)$$

- (iv) **Path backtracking:** Using the backtracking matrix $\psi_t(j)$ and beginning from the optimal path of the end state q_T^* , each relative optimal path can be estimated, as follows,

$$q_t^* = \psi_{t+1}(q_{t+1}^*).$$

In addition to the above basic steps, and in order to reduce the algorithm quadratic complexity⁶, search space reduction techniques are usually adopted to speed up the algorithm performance.

Training Problem

The problem that should be solved here is given a set of training sequences, how to automatically adjust HMMs parameters $\lambda = (\pi, \mathbf{A}, \mathbf{B})$ to maximize the probability of generating those sequences? Since there is no analytical solution for this problem, an iterative approach is adopted. Using the training set, the method starts with an initial model λ^0 and then iteratively computes a set of improved models $\lambda^1, \lambda^2, \dots$. The mostly popular technique used is the so-called Baum-Welch algorithm (BW), which is basically a variation of the generalized expectation maximization algorithm. As an optimization criteria for the HMMs parameters, BW uses the conditional probability $P(\mathbf{O}|\lambda)$, where any improved model $\hat{\lambda} = (\hat{\pi}, \hat{\mathbf{A}}, \hat{\mathbf{B}})$ should satisfy $P(\mathbf{O}|\hat{\lambda}) \geq P(\mathbf{O}|\lambda)$. To calculate the optimal path, BW basically uses the forward $\alpha_t(j)$ and backward $\beta_t(j)$ probabilities from the evaluation phase. These two variables are, in turn, used to define two additional variables that help mathematically explain the algorithm. Where $\gamma_t(i, j)$ represents the transition posterior probability of $s_i \rightarrow s_j$, for a given observation sequence \mathbf{O} , a model λ , and at time t ,

$$\begin{aligned} \gamma_t(i, j) &= P(q_t = s_i, q_{t+1} = s_j | \mathbf{O}, \lambda) = \frac{P(q_t = s_i, q_{t+1} = s_j, \mathbf{O} | \lambda)}{P(\mathbf{O} | \lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \alpha_t(i) \beta_t(i)}. \end{aligned} \quad (2.39)$$

Furthermore the variable $\gamma_t(i)$ is also introduced to represent the probability of being in the state s_i at time t ,

$$\gamma_t(i) = P(q_t = s_i | \mathbf{O}, \lambda) = \sum_{j=1}^N \gamma_t(i, j). \quad (2.40)$$

⁶In the recursion phase, each step needs N^2 maximizations to evaluate the local optimal path.

Given the definitions in Eq.(2.39) and Eq. (2.40), it is possible now to estimate the initial probabilities $\hat{\pi}$, the transition probabilities \hat{A} , and also the emission probabilities (in case of discrete HMMs) \hat{B} for the improved model $\hat{\lambda}$, as follows

$$\hat{\pi}_i = P(q_1 = s_i | \mathbf{O}, \lambda) = \gamma_1(i). \quad (2.41)$$

The expected state transition probabilities are computed by summing over each state transition probability, then normalizing the output by the total number of outgoing transition from the state,

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} P(q_t = s_i, q_{t+1} = s_j | \mathbf{O}, \lambda)}{\sum_{t=1}^{T-1} P(q_t = s_i | \mathbf{O}, \lambda)} = \frac{\sum_{t=1}^{T-1} \gamma_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad (2.42)$$

Estimations of the discrete observation probabilities are similarly performed, by summing the number of observing a symbol for a given state and normalizing the result by the total number of emission generated in the same state.

$$\hat{b}_{jk} = \frac{\sum_{t=1}^T P(q_t = s_j, o_t = v_k | \mathbf{O}, \lambda)}{\sum_{t=1}^T P(q_t = s_j | \mathbf{O}, \lambda)} = \frac{\sum_{t:o_t=v_k} \gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)} \quad (2.43)$$

Finally, it is useful to mention that we limited the above discussion to the standard or discrete HMMs (since it is related to this work), continuous HMMs or continuous HMMs are not considered; we refer the interested reader to [43, 79] for details.

2.5.2 Conditional Random Fields CRFs

In the previous subsection, we have presented the basics of HMMs as a candidate for generative based classification approaches. This subsection will briefly discuss the fundamentals of a discriminative probabilistic classifier, namely, the Conditional Random Fields (CRFs) [80, 81]. CRFs are undirected graphical models which are particularly well suited for labeling sequential data, they are discriminative models since they are directly modeling the conditional distribution $P(\mathcal{S} | \mathbf{O})$ of a set of labels (states) \mathcal{S} given a sequence of observation \mathbf{O} [82]. For the Arabic handwriting classification task, we assigned each letter basic shape to a single label, e.i, a letter BAA (ب), for example, has four different labels corresponding to its different basic shapes (ب , بـ , بـ , بـ). And observations are sequences of shape descriptive features that are extracted from the segmented letter which will be detailed in Chapter 5. The CRFs model can be thought of as an undirected graph that composed

of two parallel chains of random variables, the first is depicting the state sequence \mathcal{S} , while the other represents the observed (output) sequence \mathcal{O} . FIG. 2.15, shows an example of a first order CRFs where label sequence forms a chain.

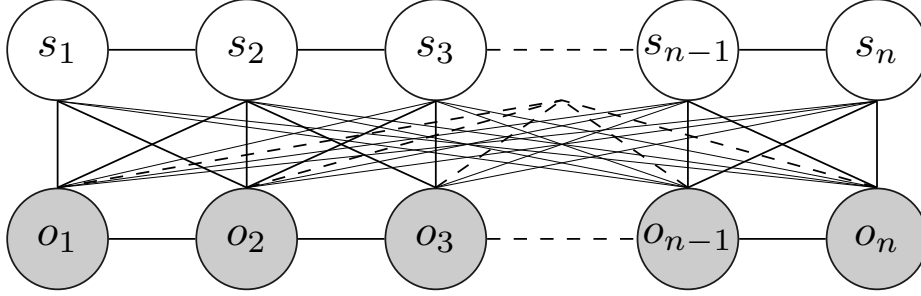


FIG. 2.15. A first order CRFs. Top are the states sequence and bottom are observation sequence. Dotted lines indicate connections between the observation and every other state [3].

As previously mentioned that CRFs are basically undirected graphical model, hence their probability distribution is calculated as a globally normalized product of feature transition functions over the labels and the observation sequence⁷. The CRFs that we adopted is involving transition feature functions that operates on adjacent label s_{i-1} and s_i . Accordingly, the probability of label sequence \mathcal{S} for a given observation sequence \mathcal{O} , is estimated as:

$$p_{\theta}(\mathcal{S}|\mathcal{O}) = \frac{1}{Z_{\theta}(\mathcal{O})} \cdot \exp \left(\sum_{i=1}^n F_{\theta}(s_{i-1}, s_i, \mathcal{O}, i) \right) \quad (2.44)$$

where $Z_{\theta}(\mathcal{O})$ is a global normalized factor given by:

$$Z_{\theta}(\mathcal{O}) = \sum_{\mathcal{S}} \exp \left(\sum_{i=1}^n F_{\theta}(s_{i-1}, s_i, \mathcal{O}, i) \right) \quad (2.45)$$

and parameter $\theta = (w_1, w_2, \dots, w_{N_f}; \mu_1, \mu_2, \dots, \mu_{N_g})$ represents weight values learned from the training data, N_f refers to the number of transition feature functions, N_g is the number of state feature functions and n is the length of the observation sequence \mathcal{O} . Furthermore, the feature function F_{θ} is defined over a pair of states (s_i, s_{i-1}) and an observation sequence \mathcal{O} as follows:

$$F_{\theta}(s_{i-1}, s_i, \mathcal{O}, i) = \sum_f w_f t_f(s_{i-1}, s_i, \mathcal{O}, i) + \sum_g \mu_g r_g(s_i, \mathcal{O}, i) \quad (2.46)$$

⁷Feature function is represented in an exponential form to ensure positivity.

where $t_f(s_{i-1}, s_i, \mathbf{O}, i)$ is a transition feature function over the entire observation sequence and at state positions i and $i - 1$ in the label sequence. Transition feature functions are usually used to indicate whether a feature value is observed between two states or not, and defined by

$$t_f(s_{i-1}, s_i, \mathbf{O}, i) = \begin{cases} 1 & \text{if } s_{i-1} = L_a \text{ and } s_i = L_b \\ 0 & \text{otherwise,} \end{cases} \quad (2.47)$$

where L_a and L_b are two labels of the model.

The state feature function $r_g(s_i, \mathbf{O}, i)$, on the other hand, indicates whether a feature value is observed at a particular label or not, and defined as follows

$$r_g(s_i, \mathbf{O}, i) = \begin{cases} 1 & \text{if } s_i = L_a \\ 0 & \text{otherwise,} \end{cases} \quad (2.48)$$

Eventually, w_f and μ_g represent the weights of the transition feature function and the state feature functions respectively, and they are typically estimated from training data. Given equations Eq. (2.44) and Eq. (2.46), the posterior CRFs probability in Eq. (2.44) of a sequence of labels \mathbf{S} and a sequence of observation \mathbf{O} , can be expanded and rewritten as follows,

$$p_\theta(\mathbf{S}|\mathbf{O}) = \frac{1}{Z_\theta(\mathbf{O})} \cdot \exp \left(\sum_{i=1}^n \sum_f w_f t_f(s_{i-1}, s_i, \mathbf{O}, i) + \sum_{i=1}^n \sum_g \mu_g r_g(s_i, \mathbf{O}, i) \right) \quad (2.49)$$

2.5.2.1 Learning Parameter for CRFs

CRFs training, is a maximum entropy based process of estimating the parameter $\theta = (w_1, w_2, \dots, w_{N_f}; \mu_1, \mu_2, \dots, \mu_{N_g})$ that maximizes the logarithm of the likelihood ℓ (known as log-likelihood) of labeled sequences in some training data set $D = \{(o^{(j)}, s^{(j)})\}_{j=1}^{T_d}$, where, $o^{(j)}$ is an observation sequence, $s^{(j)}$ represents the corresponding label sequence, and T_d refers to the number of training sequences. For a probability distribution, the highest degree of uniformity is typically achieved when the distribution entropy has the maximum possible value, thus the maximum entropy principle is employed to justify the selection of the (most representative)

probability distribution selection. For CRFs, the log-likelihood objective function that should be optimized is given by:

$$\ell(\theta) = \sum_{j=1}^{T_d} \log p(s^{(j)}|o^{(j)}, \theta) = \sum_{j=1}^{T_d} \left(\sum_{i=1}^n F_{\theta}(s_{i-1}^{(j)}, s_i^{(j)}, o^{(j)}, i) - \log Z(o^{(j)}, \theta) \right) \quad (2.50)$$

Since there is no closed-form solution for Eq.(2.50) , several iterative approach is proposed to search for the best possible optimal solution. Generally, it is found that exponential based model such as CRFs can be trained relatively faster using quasi-Newton methods, e.g. Broyden-Fletcher-Goldfarb-Shanno (BFGS) , Limited-memory BFGS (L-BFGS), DavidonFletcherPowell formula (DFP), etc [83]. These methods approximate the derivatives of the log-likelihood as follows,

$$\frac{\partial \ell(\theta)}{\partial \theta} = \sum_{j=1}^{T_d} \left(\sum_{i=1}^n \frac{\partial F_{\theta}(s_{i-1}^{(j)}, s_i^{(j)}, o^{(j)}, i)}{\partial \theta} - \sum_o p(s|o^{(j)}) \sum_{i=1}^n \frac{\partial F_{\theta}(s_{i-1}, s_i, o^{(j)}, i)}{\partial \theta} \right) \quad (2.51)$$

In our work, the likelihood maximization is performed using a gradient descent method and adopting the BFGS technique for optimization.

2.5.2.2 Inference CRFs

The calculation of the probability $p_{\theta}(S|\mathcal{O})$ of a label sequence S , given a sequence of observation \mathcal{O} and set of learned weight values θ , can be efficiently implemented by defining a set of arrays M [81]. Assuming S is the set of the entire label space of the training data, and s_i, s_{i-1} are two labels drawn from S , firstly, the set M of $n + 1$ squared arrays are estimated, where each element M_i is defined by,

$$M_i(s_i, s_{i-1}|o) = \exp (F_{\theta}(s_i, s_{i-1}, \mathcal{O})), \quad s_i, s_{i-1} \in S \quad (2.52)$$

Secondly the normalization factor $Z(\mathcal{O})$ over the observation sequence \mathcal{O} is computed as the multiplication of observation sequence with all $n + 1$ arrays,

$$Z_{\theta}(\mathcal{O}) = \left(\prod_{i=1}^{n+1} M_i(\mathcal{O}) \right) \quad (2.53)$$

Eventually, the CRFs conditional probability is given as the product of the elements of the $n + 1$ arrays as follows,

$$p_{\theta}(\mathcal{S}|\mathcal{O}) = \frac{1}{Z_{\theta}(\mathcal{O})} \cdot \prod_i^{n+1} M_i(s_{i-1}, s_i|\mathcal{O}) \quad (2.54)$$

2.6 Conclusion

The value of this chapter in the thesis is to attempt to further broaden and deepen the understanding of the basics of the problem of automatic handwriting recognition in general. In the context of this thesis, the chapter has discussed most of the sub problems, presented related literature solutions, and also highlighted our respective contributions. Most of the common employed processing steps from the signal acquisition until the classification are adequately reviewed focusing primarily on techniques, algorithms, and methods that we used, combined, or improved throughout our work. The chapter initialized by explaining the differences between online and offline handwriting signals in terms of their formats and their resources. Then, document related image enhancement techniques such as, binarization, smoothing, thinning, etc., are considered. Furthermore, special attention has been paid to handwriting specific distortions like the handwriting skew and slant. In this regard, we have presented the common HT based technique and the contour local minima regression technique. These techniques are combined in Chapter 4 to propose an improved solution. As being a prerequisite for any unconstrained recognition system, the bottleneck issue of handwriting segmentation has been also addressed, and the two different paradigms (explicit and implicit) of approaching the problem are explained. The chapter concluded the discussion by shedding light on the theoretical foundations of the generative HMMs and the discriminative CRFs.

A Benchmark Database for Offline Arabic Handwriting and Arabic Handwriting Synthesizing

IN this chapter, we address the problem of performance evaluation for developing Arabic handwriting recognition approaches; in particular, how to construct databases of handwriting images with verified ground truth. Typically, databases are of crucial importance for developing better algorithms and also for profound comparisons of the proposed methods [8]. This chapter is generally divided into two major sections. In the first part, we will present the database that is constructed directly from real handwriting such as handwritten words, handwritten letters, and handwritten historical manuscripts.

In the second part, we explain our proposed approach for generation of synthesized handwriting. Our offline handwritten Arabic database has been developed in the Neuro-Information Technology department, Institute for Electronics, Signal Processing and Communication (IESK) at Otto-von-Guericke University Magdeburg, Germany¹. It is called IESK-arDB, where "IESK" is the German abbreviation of the Institute name and "arDB" stands for "Arabic Database" [8]. The main motive behind developing IESK-arDB database, is the demand for a database that is carefully designed, well articulated, covering different parts of speech, and most importantly useable for segmentation-based as well as segmentation-free recognition approaches. Unlike other databases in the field, IESK-arDB is the first database

¹The institute new name has become, Institut für Informations und Kommunikationstechnik (IIKT).

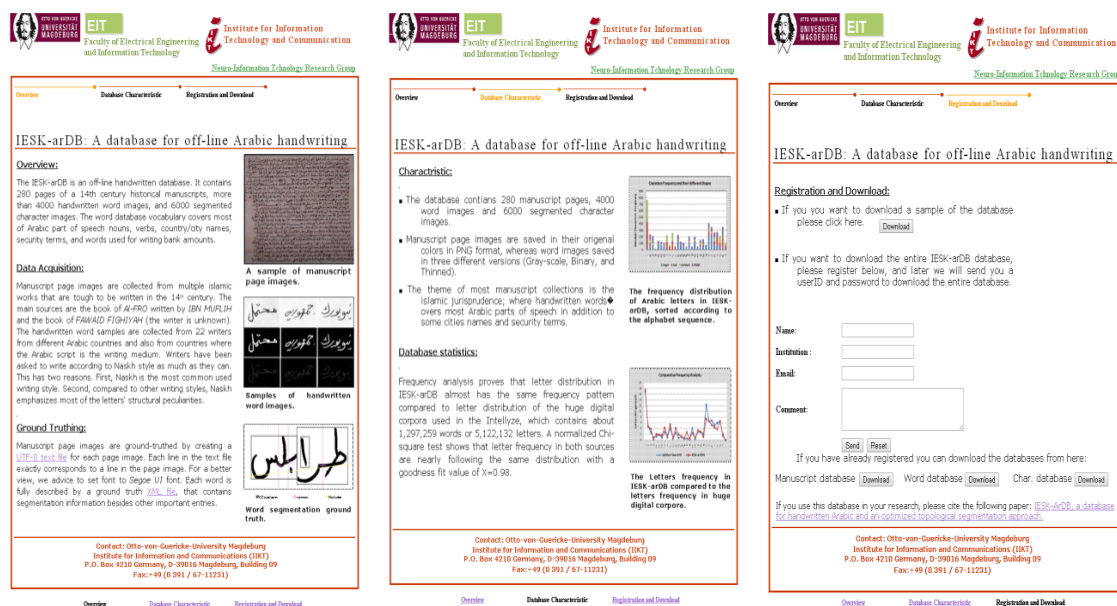


FIG. 3.1. A screen-shot from the IESK-arDB database website.

that is equipped with manually annotated information regarding letters boundaries in the handwritten word images, in addition to all other necessary ground truth information. We believe that segmentation ground truth is of crucial importance for the training and validation purposes of any explicit segmentation-based recognition approach. The database contains 6000 handwritten word images gathered from 22 people, 8000 letter images representing all letters of the Arabic alphabet in all different forms, and 285 manuscript page images collected from multiple Islamic documents thought to be written in the 14th century.

Due to the high cost associated with the process of creating a comprehensive database, in the second part, we introduce our approach to automatically generate synthesized words and/or lines of handwritten text from Unicode text [84]. A total of 28000 online handwritten letter samples are collected from several writers, and used to build 104 Active Shape Models (ASM) (a model for each letter in each form). ASMs are then used to generate unique letter representations in order to simulate the different user's writing style. Furthermore, to produce smooth text, counter is interpolated using B-Spline and affine transforms are employed to imitate slant and skew distortions. The developed system is used to create more than 12000 synthesized samples that have been added to the database. The database is free of charge and available for download at (www.iesk-ardb.ovgu.de).

3.1 Offline Handwriting Database

This section details the process of developing the offline handwritten contents of the database, which consists of two different kinds of handwritten material. The first is composed of handwritten words that are mainly derived from the language basic part of speech (e.g., noun, verb, etc.), countries and cities names from around the globe, glossary of security terms, and words used for writing amounts on checks. The main purpose of this data is to train and test the segmentation as well as holistic based approaches for the recognition of handwritten Arabic words. The second part makes available hundreds of pages of medieval Arabic manuscripts along with their respective ground truth files. Historical manuscripts that are carefully transcribed are a necessary resource for developing solutions for related problems such as manuscript image enhancement, molding of manuscripts handwriting, and most importantly the recognition of the manuscript content. To the best of our knowledge, this is the first database that freely offers such a large number of transcribed historical Arabic documents.

3.1.1 Data Acquisition

The database vocabulary covers most of Arabic part of speech (noun, verb, etc.), country/city names, security terms, and words used for writing bank amounts. For the purpose of collecting data, we used an entry form consisting of eight pages, each page is filled with eight handwritten words presented to the writers. Samples are collected from 22 writers from different Arabic countries and from countries (non-Arabic speaking) where the Arabic alphabet is used in writing. FIG. 3.2 shows a page from the form used for collecting handwritten samples. Writers have been asked to write according to Naskh style as much as they can. This was done for two reasons. First, Naskh is the most common used writing style. Second, compared to other writing styles, Naskh emphasizes most of letters structural peculiarities [85].

The extraction process started by scanning the filled forms with a flatbed Epson Perfection V300 Photo scanner. Usually, scanning with 300 dpi is enough for subsequent OCR processing steps. In Arabic, besides the alphabet there are what so called diacritic symbols, which are used to indicate vowels, and written in a very small size (compared to letters size) above or below a letter (e.g., أصدقاؤه, الرحمن). To

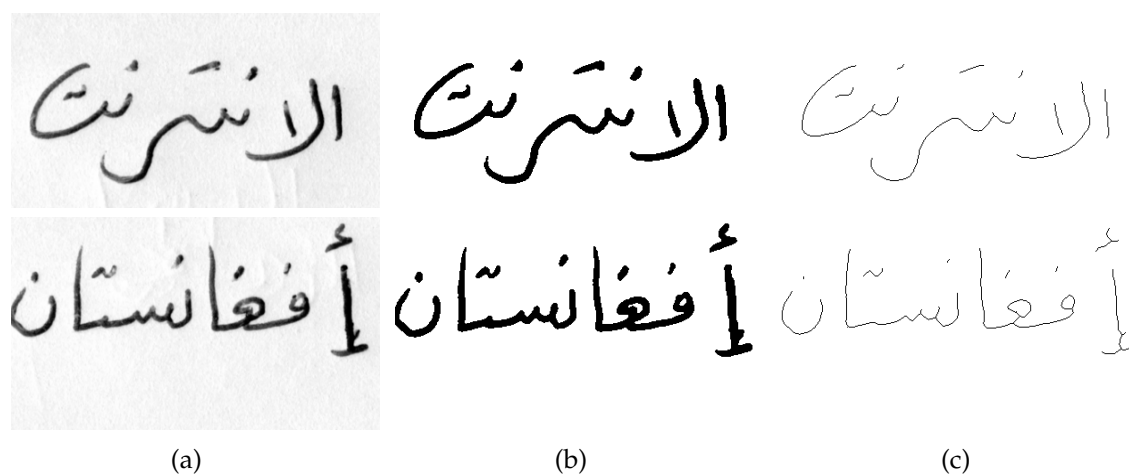


FIG. 3.3. Samples of handwritten words: (a) Gray scale images, (b) binary images, and (c) thinned images.

types. The first format is an enhanced gray scale version; the second is a globally binarized white on black version; and the third is a thinned version, where the thinning approach adopted is based on the Zhang-Suen's thinning algorithm [42]. FIG.3.3 shows gray samples from the database with their binary and thinned versions.

The manuscripts part of the database consists of 285 page images extracted mainly from two Islamic theological works thought to be written in the 14th century². The first is the book of AL-FRO, which is written by IBN-MUFLIH, a 14th century prominent scholar of religious law of Sunni Islam. The second is the book of FAWAID FIQHIYA, written also in the same period by an unknown author. The ink used in those manuscripts is the iron gall ink, with brown and black are the most dominant colors. Manuscripts are open-face scanned in 96×96 PPI with 24 bit depth and stored in PNG format. As result of open-face scanning, each image file contains two pages, files are then cropped into two separate 1-page image files, which are named after the original 2-pages image file, adding ".1" or ".2" suffixes to differentiate them.

3.1.2 Database Annotation

Ground truth information is always an essential part of every OCR database, since it is a prerequisite for recognition experiments [10]. In our database, each word is

²The manuscripts are donated by Dr. Mostafa Ahmad.

fully described by a ground truth Extensible Markup Language (XML) file, that contains several important entries. FIG.3.4 shows an example of one of such files. All ground truth files are named after their corresponding gray word image file name. As FIG. 3.4 depicting each XML ground truth file starts with a main tag

```

<?xml version="1.0" encoding="UTF-8" ?>
- <Imagefile>
  <Id Id="101_003" word="طرابلس" Meaning="Tripoli" />
  - <LetterLabel>
    <Letter5 name="Taa" shape="ط" unicode="1591" />
    <Letter4 name="Ra" shape="ر" unicode="1585" />
    <Letter3 name="Alif" shape="ا" unicode="1575" />
    <Letter2 name="Ba" shape="ب" unicode="1576" />
    <Letter1 name="Lam" shape="ل" unicode="1604" />
    <Letter0 name="Sin" shape="س" unicode="1587" />
  </LetterLabel>
  <Baseline ax="349" ay="128" bx="22" by="132" />
  - <Subwords>
  - <Subword0>
    <Bound ax="183" ay="39" bx="342" by="171" />
    - <Letter>
      <Dividing_point0 x="271" y="131" />
    </Letter>
  </Subword0>
  - <Subword1>
    <Bound ax="185" ay="58" bx="218" by="135" />
    <Letter />
  </Subword1>
  - <Subword2>
    <Bound ax="36" ay="55" bx="180" by="171" />
    - <Letter>
      <Dividing_point0 x="150" y="119" />
      <Dividing_point1 x="128" y="126" />
    </Letter>
  </Subword2>
  </Subwords>
  <Additional_Information Age="1" Gender="m" Education="1" Region="0" Quality="2" />
</Imagefile>

```

FIG. 3.4. An XML ground truth file describing the word image shown in FIG. 3.5.

Imagefile, which contains the following XML elements, sub-tags, and attributes:

1. "Id" tag, contains three attributes "Id", "Word", and "Meaning", that hold the name of the gray image file, the transcript of the word in Arabic, and the meaning in English, respectively.
2. "LetterLabel" tag has usually a number of elements equivalent to the number of letters in the word. Each element with three attributes "Name", "Shape", and "Unicode", show the pronunciation of the letter in English, the letter basic shape, and the corresponding Unicode, respectively.

3. The "**BaseLine**" element with four attributes, representing the coordinates of two points identifying the word base-line.
4. The "**Subwords**" tag contain one or several sub-tags "**Subword#**", each has the coordinates of the upper-left and the lower-right corners of the bounding box enclosing a sub-word. In case of a sub-word that is made up of multiple letters, and according to the number of letters in the sub-word, tag\s "**Letter**" and "**Dividing_point#**" elements with two attributes will be added to indicate the coordinates of the border points between letters in the sub-word.
5. The last element "**Additional Information**", gives personal information about the writer, e.g., education level, the region where the writer learned to write, and etc., as well as human assessment of the clarity of the handwriting.

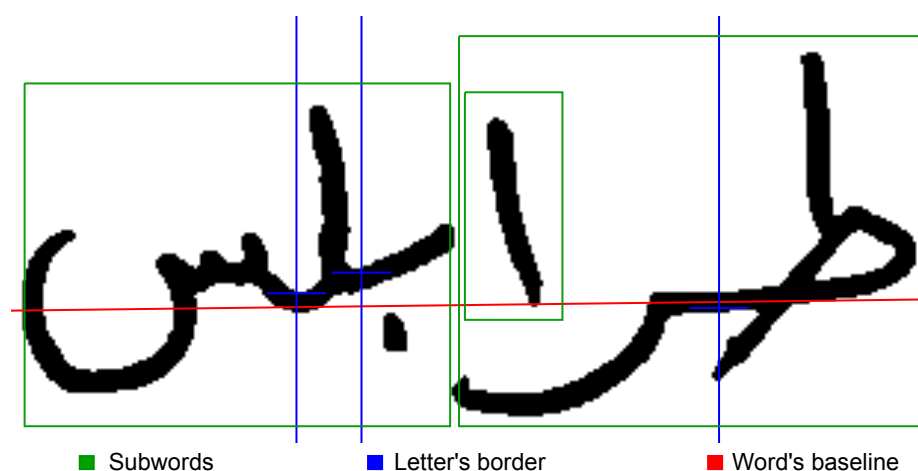


FIG. 3.5. Example of the ground truth information given in FIG.3.4, plotted against its corresponding binary image.

FIG.3.5 shows the binary image corresponding to the ground truth XML file shown in FIG.3.4. The word consists of three sub-words, which are indicated with the three sub-word tags bounded with green boxes in FIG.3.5. The first sub-word from the right consist of two letters with their border point indicated by the blue crosshair. The second sub-word contains only one letter; hence it is a sub-word as well as a letter at the same time. Similarly, the third sub-word contains three letters and two dividing points; and the word baseline is plotted in red.

As for manuscripts part annotation, it is being observed that several prominent libraries and archives (e.g., The British Library, Princeton digital library of Islamic

manuscripts, Bibliotheca Alexandrina, Ottoman archives, etc.) are digitizing their collection of Arabic, Persian, and Ottoman manuscripts and make them available through the Internet. However, libraries offer only scanned images without any machine-readable text redundancies that might be used as ground truth to benchmarking OCR methods for manuscripts. In IESK-arDB database, we manually created a digital Arabic transcription for each manuscript page, and we enforce a line-by-line alignment between text lines in the transcript and the page image. Transcription files are then saved in plain text where each letter in the image corresponds to one single letter in the transcription. While Kashida³ is represented in the transcript and each human un-readable letter is substituted by a "?", and vowel diacritics are not included.



FIG. 3.6. Manuscript sample from the IESK-arDB database, and the equivalent machine-readable text with one-to-one line correspondences.

3.1.3 Letters Frequency Analysis

Language statistics are the foundation upon which different language models can be built [86]. Word-based language models or/and character-based language models are statistical models that provide prior knowledge about words or letters occurrences, respectively. Hence, we conducted letter frequency analysis on our database (IESK-arDB). In general, the average length of 4.3 letters of Arabic word

³Kashida is an Arabic text justification symbol used to increase spaces between connected letters.

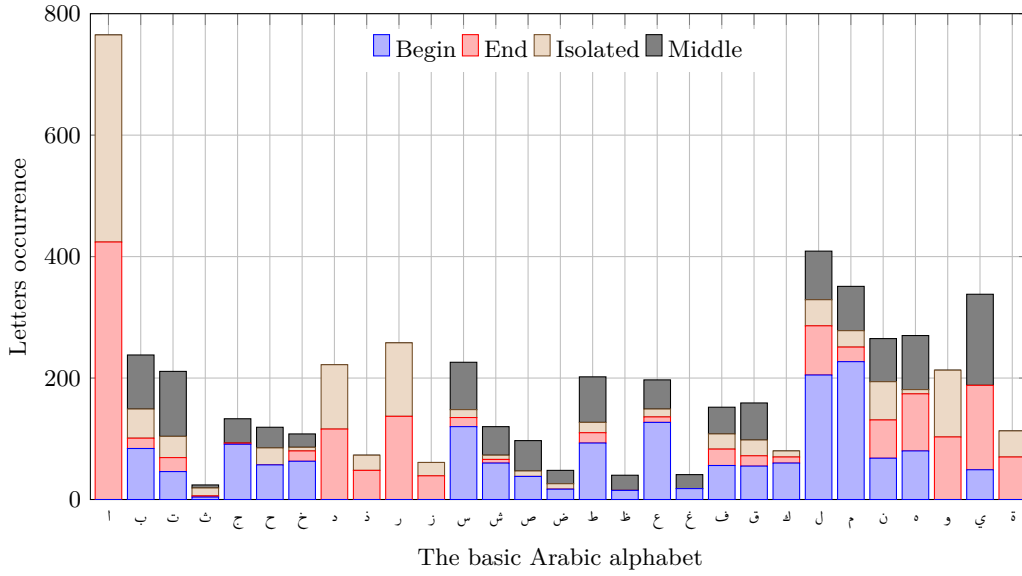


FIG. 3.7. The frequency distribution of Arabic letter in IESKarDB, sorted according to the alphabet sequence.

is relatively shorter compared to other scripts such as English with 5.1 letters, and German 6.26 letters [87, 88]. According to the same references, the most frequent letters are Alef "ا" 17.5%, Lam "ل" 7.3%, Yeh "ي" 6.5%, and the less frequent are Tha "ث" 0.2%, Zah "ظ" 0.4%, and Dhad "ض" 0.6%. Our database retains almost the same characteristics, where the word average length is about 4.4 letters, and the most appearing letters are Alef "ا" 16.9%, Lam "ل" 12.3%, and Yeh "ي" 6.6%. Less frequent, however, are Zah "ظ" 0.2%, Gian "غ" 0.33%, and Dhad "ض" 0.45%.

In this context, it is worth mentioning that frequency analysis will include only the letter basic form, and if a letter appeared in any modified form, it will be counted as appearing in the basic form. Hence, the letter Alef "ا" modified forms (أ, إ, آ) will be counted as Alef "ا". The modified forms such as (و, ؤ) and (ي, ئ) will be counted only as Waw (و) and Yeh (ي), respectively. FIG.3.7 shows the frequency distribution of the basic letters in IESK-arDB with their respective occurrence shapes (Begin, End, Isolated, Middle), and FIG.3.8 illustrates the letters distribution of IESK-arDB compared to the letters distribution in huge corpora of text, that contains 1,297,259 words or 5,122,132 letters [4]. A normalized Chi-square test shows that letter frequencies in both sources are nearly following the same distribution with a goodness fit value of $\chi^2 = 0.980$. These results prove that the IESK-arDB database is unbiased and follows the common distribution of the language model.

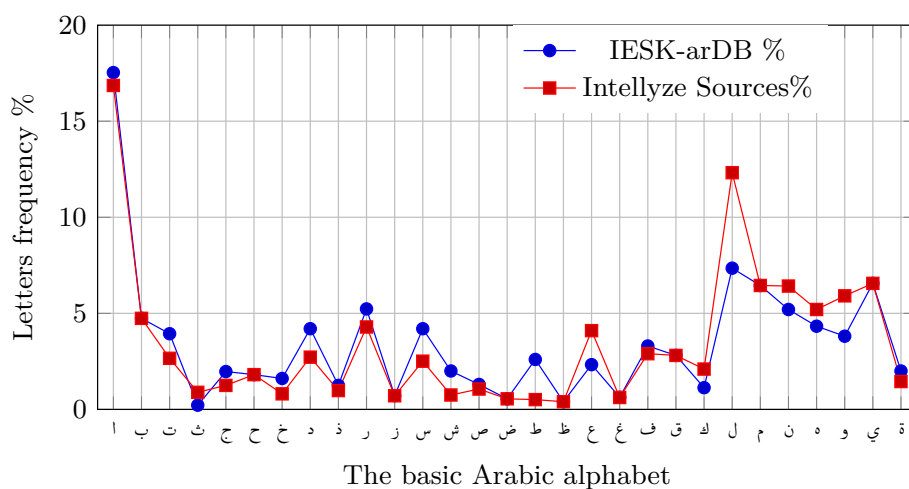


FIG. 3.8. The letters frequency in IESK-arDB compared to the letters frequency in the sources used in [4].

3.2 Handwriting Synthesis

Training, testing, or benchmarking offline handwriting recognition methods, require broad, comprehensive and well annotated databases. On the one hand, creation of such databases is usually very manpower, time, and computational demanding process, but lack of such resources is still among the top hindrance that slowdown the research in this field, and this is obvious in case of holistic based recognition methods regardless of underlying script [84]. This is because of the fact that an efficient holistic recognition, requires collection of a relatively big number of handwritten samples for each vocabulary of the lexicon of the intended application area. To overcome this problem, it would be very helpful, if the process of human handwriting can be imitated. So sufficient artificial handwriting samples can be automatically generated, and this process is called handwriting synthesis [89]. In this context, we developed a system that generates synthetic handwriting sample images from UTF-8 and/or Unicode text, along with their ground truth information files [84,90,91]. The generated data contains line-by-line aligned digital text, sub-word bounding box coordinates, and coordinates of border points between connected letters. We collected about 28000 online handwritten samples and used them to calculate an ASM model for each letter in each form, ending up with a total of 104 models. Further, each ASM is manipulated to generate a broad variety of different handwritten shapes of the corresponding letter. To make synthesized

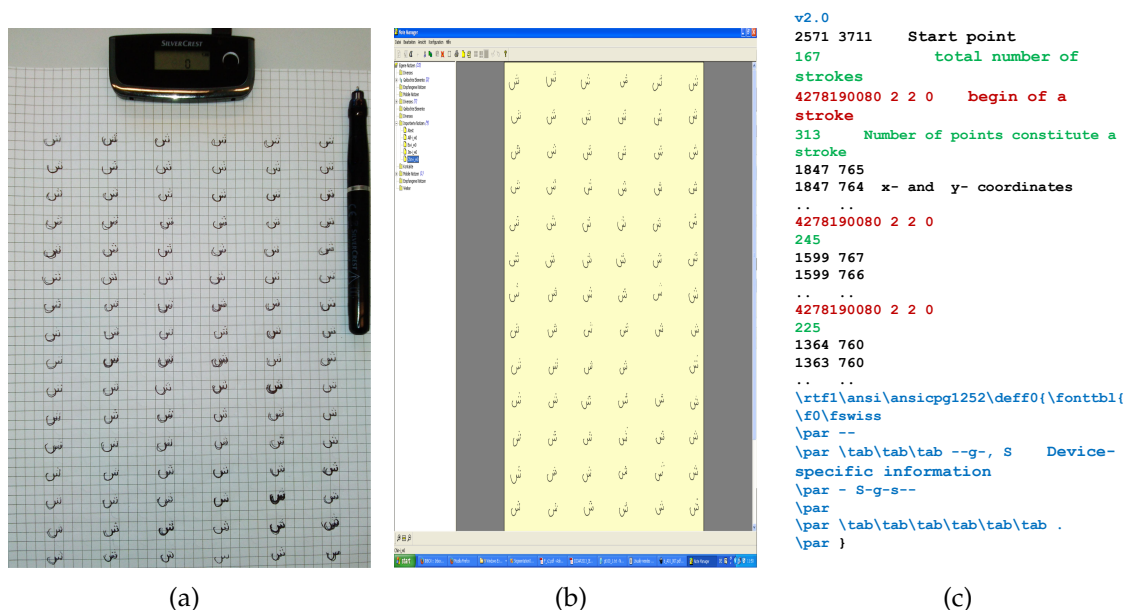


FIG. 3.9. On-line data acquisition: (a) Acquisition system (the base unit and the digital pen), (b) screen-shot of system interface, and (c) an example illustrates the structure of the ground truth information file.

samples look authentic as much as possible, affine transforms are used to add human-like handwriting artifacts such as skew, slant, etc. To smooth the composed text, B-Spline is employed for interpolation. In the following sub sections we will explain each step in more details.

3.2.1 Online Data Acquisition

For online samples' collection, we used a system produced by *SliverCrest* [92], that consists of a wireless digital pen (DGP1000), and a base unit that should be clipped onto the top of a paper or a block of papers, as shown in FIG.3.9 (a). FIG.3.9 (b) shows the screen-shot of the acquisition software. While writing on normal paper the base unit detects the pen strokes trajectories and saves their corresponding coordinates as well as a vector graphic image of the writing. Using the trajectory points of a letter, we created an equivalent ground truth information file that contains, the total number of strokes, the beginning of each stroke, and a time-stamp, as depicted in FIG.3.9 (c). For every shape, a letter may appear in, at least 50 samples are collected from each writer that amount to a total of 28046 samples altogether. These samples were eventually aligned and used to construct a model for each individual shape.

3.2.2 Modeling and Synthesizing of Letter Shapes

ASMs are statistical models that accommodate the variability in shape and appearance over a set of training samples for a given shape class [93]. For the class shape modeling, ASM algorithm requires a set of training samples landmarked with a constant (optimized) number of representative points that are usually manually imposed along the shape contour.

Giving the high variability of handwriting, manual landmarking is proved to be time-consuming, laborious, and expensive. Hence, benefiting from the fact that we can access the x and y coordinate of trajectories as a function of time, we combined the local extrema in the curvature and local extrema in the velocity, in order to automatically generate n landmark points $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$. Even though, n can be optimized for each individual class, empirically, we found that $n = 25$ is sufficient to represent even the most complex letter shape as shown in FIG.3.10 for the letter "س". The landmark points (x_i, y_i) for a single training sample can be represented as $2n$ element vector \mathbf{x} , where

$$\mathbf{x} = (x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n)^T \quad (3.1)$$

To build an ASM for a letter class we used a set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$ of vectors \mathbf{x} , which in turn forms a distribution of shape landmark points in the $2n$ space. Intuitively, k can be optimized for each individual class, where k is the number of vectors in each letter class. However, $k = 50$ is experimentally found to be the most cost-effective value. To synthesize more samples similar to the ones in \mathbf{X} , we should first start by modeling the distribution of the landmarks points of \mathbf{X} . To accomplish this task, we started by applying the well known Principal Component Analysis (PCA) on the data, which is an effective technique for both dimensionality reduction and data modeling. To apply PCA on the data, the data covariance matrix \mathbf{S} is computed as.

$$\mathbf{S} = \frac{1}{k-1} \sum_{i=1}^k (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \quad (3.2)$$

Using \mathbf{S} , the set \mathbf{W} of eigenvectors as well as their corresponding vectors λ of eigenvalues can be straightforwardly computed. Due to the fact that eigenvalues are actually representing the variance around the mean $\bar{\mathbf{x}}$ of the data along the direction of the corresponding eigenvector, synthesized sample of a given class can

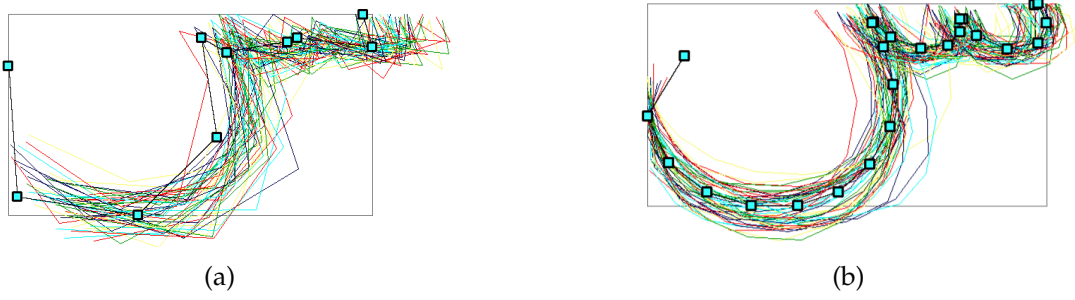


FIG. 3.10. Samples used to build Active Shape Models (ASMs) for the letter Sin (س): (a) samples of training trajectories with 10 landmarks points, vs. , (b) with 25 landmarks points.

be approximated by a linear combination of $\bar{\mathbf{x}}$, and the eigenvectors \mathbf{w} of the class. Furthermore, to guarantee that generated samples are similar to the samples used in the training of the class, a variable c is used as a parameter in Eq.3.3 to represent the variations of the eigenvalues in a pre-specified range (± 2).

$$\hat{\mathbf{x}} = \bar{\mathbf{x}} + \sum_{j=1}^k c_j \mathbf{w}_j, \quad \text{where } c_j \in [\pm 2\sqrt{\lambda_j}], \quad \lambda \in \boldsymbol{\lambda}, \quad \text{and } \mathbf{w} \in \mathbf{W} \quad (3.3)$$

3.2.3 Synthesizing of Handwritten Words

The input to the handwritten word synthesizing process is a sequence of Unicode characters representing the digital word or piece of digital text. Since we built ASMs only for the standard set of the alphabet, we, firstly, start by substituting any derived shape that may appear in a digital word with the basic letter shape. So that the derived letter shapes such as (ل, ل, ل) will become (ل), and (و, و) will be (و), and etc. Afterwards, the relative position (Isolated, Begin, Middle, End) of the letter in the word is determined, and eventually the corresponding ASM will be loaded and used to generate a trajectory similar to the letter shape. After creating trajectories for each individual letter in the Unicode sequence, trajectories are then aligned and juxtaposed in a sequence.

To compose a word or a sub-word, the trajectory of a letter (left connect-able letters) is connected to the subsequent trajectory by simply translate the coordinates of the trajectory on the left by a distance $(\mathbf{e} - \mathbf{b})$, where \mathbf{e} is the coordinates of the leftmost point of trajectory on the right and \mathbf{b} is the coordinates of rightmost point of the trajectory on the left. The estimation of the vertical position (y coordinates)

Table 3.1. Effects incorporated into the synthesized system.

Stretching	Slant	Size	Skew	Position	Connection
نيويورك	نيويورك	نيويورك	نيويورك	نيويورك	نيويورك
نيويورك	نيويورك	نيويورك	نيويورك	نيويورك	نيويورك
نيويورك	نيويورك	نيويورك	نيويورك	نيويورك	نيويورك

relative to the baseline usually depends on the shape of the letter, we computed the average μ and the variance σ of the distance between the baseline and the center of a letter from the manually annotated baseline for the aforementioned handwritten word's database, and then translate the sub-word y-coordinates by $\mathcal{N}(\mu, \sigma)$. Finally, the distance between sub-words in the same word is determined using a parameter that may take negative values to represent the vertical overlapping of the word sub-words.

In our approach, ASMs are mainly dedicated to introduce local shape variation within a letter class, in order to add needed shape variations that may occur on the level of sub-word or a word, various types of affine transforms such as scaling, translation, shearing, and rotation are used. Table 3.1 demonstrates the various types of effects that we incorporated into our system, in order to demonstrate the impact of some of the variability sources that may alter the shape of a handwritten letter, sub-word, or/and a word. The synthesis system is extended to process complete paragraphs of digital text, which makes it capable of producing pages of handwritten synthesized text. FIG.3.11 (a) shows a paragraph of digital text, and FIG.3.11 (b) a possible synthesized version that it may appear in.

3.3 Conclusion

To sum up, this chapter presented the IESK-arDB, a new multi-propose offline Arabic handwritten database. The database is free of charge and online available for research purposes. To facilitate research efforts of segmentation based recognition of handwritten Arabic words, the database contains more than 6000 word images saved in a PNG format with three different versions, namely, gray, binary, and



FIG. 3.11. Paragraph synthesizing: (a) A digital paragraph taken from the Arabic site of the Deutsche Welle (<http://www.dw.de/>), (b) the synthesized handwriting corresponding to the text marked in yellow.

thinned binary. Additionally, for each word image, an XML ground truth file that contains all needed data is included; most important, however, is the segmentation information. A letter frequency analysis showed that the collection exhibits letter frequencies similar to that of large corpora of digital text, which proof usefulness of the database. To the best of our knowledge, IESK-arDB is still the only free database that offers a collection of Arabic handwritten manuscript images, along with Unicode transcription for each manuscript image that is line-wise aligned. The main purpose of this data is to leverage research in the field of recognition of handwritten text in historical documents, in order to allow searching and mining historical documents for information. The collection contains 285 pages taken from two medieval theological works, images saved in PNG format, and each is manually transcribed in txt files.

Lack of extensive Arabic handwriting databases is usually regarded as one of the main reasons that hinders fast progress in the field of handwriting recognition [94]. This can be attributed to the considerable time and effort needed for gathering, scanning and ground truthing. In order to automate the process of handwriting samples generation, we presented an approach for handwriting synthesis. To build an ASM for each letter shape, an online acquisition system consists of a base unit, and a wireless pen is used to collect handwritten letter samples from several writers. Samples of each shape are, first, aligned altogether and then their average shape, matrix of Eigenvectors, and vector of Eigen values are all used to compute synthesized instance from the letter shape class. In total, 104 different ASMs are

constructed to represent the letters basic shapes as well as the letters conditional shapes. Moreover, to make synthetic results looks as genuine as possible, various types of affine transforms are used to add structural noise to both the letter level and the sub-word/word level.

Segmentation of Arabic Handwriting

THROUGHOUT, this research work, two handwriting segmentation-related problems have been investigated. The first is text line segmentation of handwritten Arabic, and the second is the problem of handwritten words segmentation. Obviously, text line segmentation is an essential and inevitable task that must precede any word segmentation process. Thanks to their systematic layout and the relatively wide space between text lines, for type-written and printed documents, the problem is widely believed to be solved [95]. In case of handwritten documents, however, line segmentation is a challenging task and still considered as an open research problem that needs a lot of research to be solved. Among the main impediments that prevent an efficient solution are [96]:

- Touching and overlapping of ascenders and descenders strokes of consecutive text lines. FIG.4.1(a) and FIG.4.1(b) illustrate these two issues, respectively.
- The skew distortion that may affect a text line in a page, or even may vary along the same text line. FIG.4.1(c) shows an example for this problem.
- Extensive use of dots and diacritical marks in Arabic script that usually may appear above or below a letter, and that often vertically overlapping. This is especially true in case of historical manuscripts. Besides the previous impediment, FIG.4.1(c) simultaneously demonstrates the usage of dots and diacritics in a manuscript.

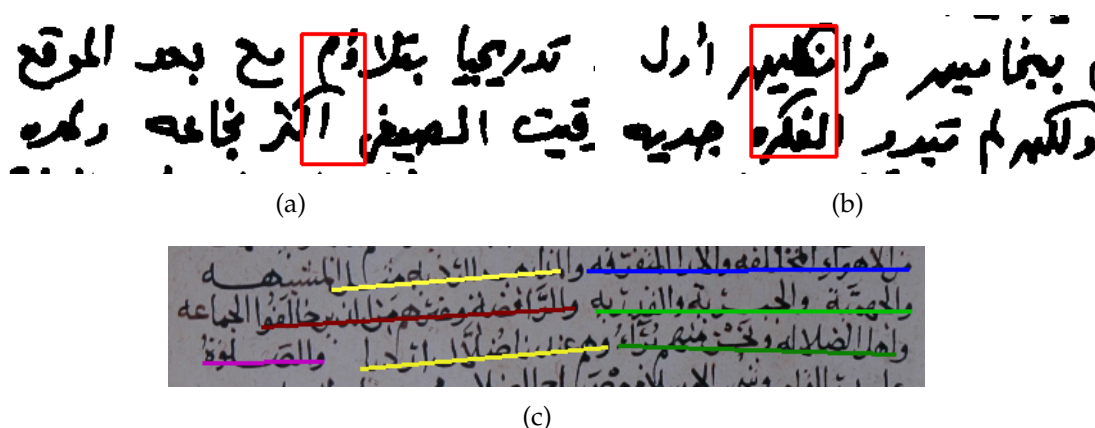


FIG. 4.1. Issues that complicate text line segmentation: (a) Touching of components, from two consecutive lines, (b) vertical overlapping of components, (c) various skew degrees across the whole page and also along the same line.

The first part of this chapter is dedicated to the problem of text line segmentation, where we will explain the problem in more details, discuss some available approaches, and ultimately illustrate our proposed solution. In general, our approach is a projection profile based technique that starts by partitioning the page into vertical strips then calculating the projection profile for each strip. Then we analyze the projection profile to elect the most probable horizontal line that separate two consecutive text lines

As for the crucial issue of handwritten word segmentation; typically, research efforts devoted to solve the problem of optical character recognition follow two broad approaches [33]. The first approach is adopting an explicit segmentation paradigm, in which a word is considered as a collection of smaller sub units such as letters or even primitive strokes, and hence the border of those sub units in the given word should be identified prior to any feature extraction process; these approaches are usually called segmentation- based or analytic-based. The other approach follows a holistic-based strategy, in which features are extracted from the whole word image, and as a result the segmentation is completely avoided. Nowadays OCRs' systems built upon segmentation-free (holistic) approaches are put successfully into service in a number of application areas such automatic reading of postal addresses and bank checks, processing documents such as forms, etc. [34]. Applications of such systems are limited to fields with a small vocabulary and cannot be used successfully in unconstrained OCR environments. This limitation

is due to the fact that the segmentation process is completely bypassed, which is an intuitively pre-requisite operation to reduce the infinite domain of possible words into a limited number of classes (graphemes or characters), that can be accommodated and processed [9,35]. Given the importance and the difficulty of the segmentation problem, solving such problem would be a great achievement in the field of OCR applications [67,68]. Hence, the chapter's second contribution is the methodology for segmentation of handwritten Arabic words based on topological features. The methodology starts by pre-processing steps such as small holes filling and smoothing the outcropped pixels, so the limitations of the extraction stage can be compensated. Then to reduce the extreme variability of handwriting, normalization issues such as Slant- and Skew- correction are considered. The segmentation then conducted through a dual-phase procedure. In the first phase, a connected component analysis is performed to resolve sub-words overlapping. Then, topological features based segmentation is carried out to segment the word into identifiable units representing their constituent characters.

4.1 Segmentation of Text Lines

Although many methods have been proposed in the literatures to solve the problem of text line segmentation, they are either designed for a specific kind of documents, or they do not propose efficient and effective solutions [95]. In this work, we were searching for an algorithm that can be applied regardless of the nature of the concerned document, and also performs the task in a reasonable time with the minimum necessary computation costs. Before delving into the details of our approach, we are going to briefly explain the main paradigms that the majority of published approaches are following.

4.1.1 Hough Transform based Approaches

Typically, HT is an important technique to detect line in images. To use the technique in the text lines identification in a document image, a set of representative points is extracted from the image and used as input for the transform [96,97]. Usually, two different types of points are used, the center of gravity points and the connected components local minima points. Since no assumption is needed regarding the

direction of the text lines, the main advantage of HT based approaches are their capability to find even text lines with different direction within the same page.

Disadvantages of the technique, however, are (i) handwritten text lines are often fluctuating around the horizontal axis resulting in imperfectly aligned points (either center of gravity or local minima points) , hence, several approaches are proposing solutions to form alignment of points prior to apply HT, which leads to costly and unsatisfactory results. (ii) Insensitivity to the horizontal distance between input points along lines, i.e., points that may be located away from each other along a line, are still considered as a text line, which is highly unlikely. (iii) Theoretically, HT applies no assumptions regarding text line directions, which is proved to be impractically in terms of computational costs.

4.1.2 Smearing based Approaches

Although smearing based methods are capable of processing document images that contain graphics, pictures, as well as text, their applications, however, are limited to printed, type written, or skew free documents [98]. This is because smearing techniques, in general, are highly sensitive to the document layout [99]. Smearing methods are usually applied to binary images, where ("1") representing background pixels and ("0") are the object's pixels. The basic algorithm works by transforming a binary sequence to another binary sequence according to some rules and an appropriate threshold, that lead to filling gaps between foreground pixels with black pixels, and as a consequence connecting foreground components. For each image, the algorithm is applied twice, one in the horizontal direction and the other in the vertical direction producing two different binary images. Then the two images are combined together using the logical operation AND to generate one image. For further enhancement, the algorithm is applied again on the resulted image using a shorter threshold and in the horizontal direction only. Finally, the bounding boxes of the connected components in the smeared image are considered to be representative of text lines.

Generally, and as previously mentioned, smearing based approaches are showing weak performance in case of slanted and/or skewed writing, which is common characteristic of handwriting.

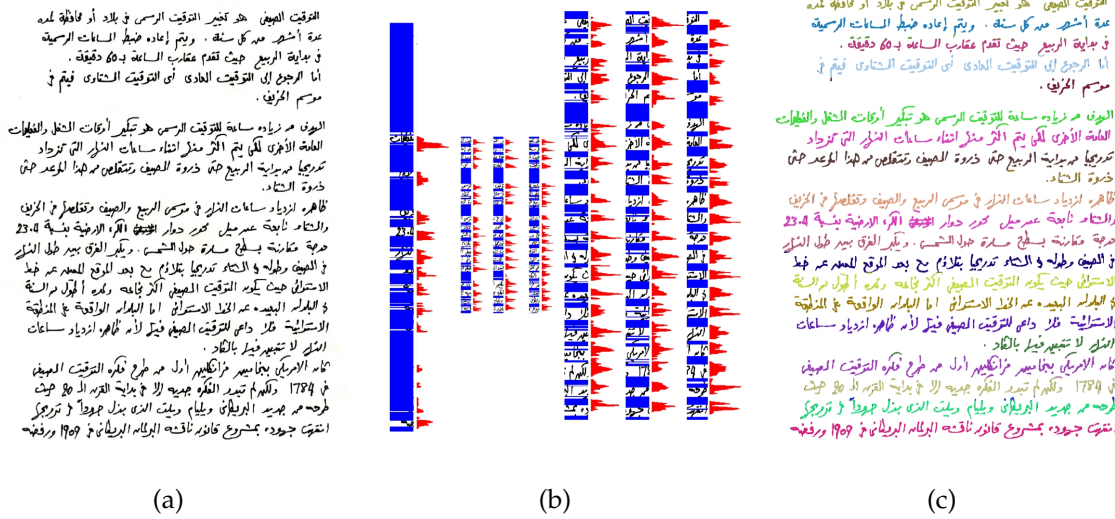


FIG. 4.2. Text line segmentation: (a) Binary skew corrected image, (b) the sub-images, where each sum-image depicted along its horizontal projection profile, (c) the segmented text lines.

4.1.3 The Proposed Approach

This section illustrates an improved method for automatic identification and segmentation of text lines in handwritten documents. The method is basically an improved histogram projection based approaches. In general, projection based text line segmentation is based on a straightforward idea, that is executed by computing the image projection profile along the horizontal axis [100]. Then by analyzing the profile peaks and valleys, the position of most probable border lines is detected. In contrast to smearing approach, projection profile method is usually less sensitive to gaps within the same text line. Moreover, it outperforms HT based approach in case of overlapping and touching connected components.

Although our method can be applied regardless of the underlying script, however, performance is further boosted in case of Arabic handwriting thanks to two reasons. Firstly, Arabic alphabet based scripts contain fewer descending and ascending letters compared to Latin based scripts, which lead to a relatively small number of cases of strokes touching and/or overlapping. Secondly, the text baseline is easy to distinguish, since all letters are usually rising or descending from a horizontal stroke, which is equivalent to a very clear peak in the image horizontal projection profile.

4.1.3.1 Page Skew Correction

Many published works suggesting various techniques for page de-skewing, yet projection based approaches are still proving their usefulness and efficiency in many cases [101]. Therefore, we embrace a projection profile based solution.

Typically, skew is holistically affecting the page and skew angles are usually varying in a finite small range. For practical reasons, we adopted $[\pm 20^\circ]$ as the skew angle range which is empirically found to cover most of the page skewness distortion angles. Then, the concerned image is rotated around its origin with the integer angles in the previously mentioned range producing several images with various skew angles. Eventually, the projection profile for each image is obtained and the corresponding variation is calculated. The profile that exhibit the highest variation is then considered to be equivalent to the projection with the best text line alignment, and the projection angle is consequently used to correct for the skew.

4.1.3.2 Text Lines Segmentation

For text line segmentation, we adopted a top-down approach that starts from the entire document page, and then proceeds by segmenting it down into the individual lines of text¹. Starting with the assumption that the number of touching and/or overlapping components from two consecutive text lines is significantly small compared to the number of non-touching and non-overlapping components. After skew correction, the document page is vertically sliced into a set of sub-images with a width proportional to the original image width, and further the horizontal projection profile for each sub-image is computed. From each sub-image, a set of border-line (instead of text-line) candidates, is generated using an adaptive threshold. The threshold is calculated upon the projection profile, by firstly excluding all zero entries in the profile and then applying a k-means clustering algorithm on the data, where the number of clusters is determined as a function of the profile length. Correspondingly, clusters means are computed and the mean with minimum value is used as a threshold.

By applying a threshold to its respective sub-image profile, a set of local border line candidates is obtained. To form a set of candidates on the level of the document

¹Instead of bottom-up approach that begins with the connected components and progressively joins them to form a text line [100].

Algorithm 4.1: Text line segmentation**Data:** Skew free binary image I .**Result:** $Ridx$, set of row indexes represent borders between text lines.**begin** $W \leftarrow ImageWidth$ $\Upsilon \leftarrow StepWidth$ $idx \leftarrow 1$ **while true do****if** $idx + \Upsilon < W$ **then** $\mathbf{K}_{(i)} \leftarrow$ crop vertical strip of size Υ , beginning at column idx 1 $k_w \leftarrow$ width of $\mathbf{K}_{(i)}$ 2 $\mathbf{p} \leftarrow \sum_{n=1}^{k_w} \mathbf{K}_{(i)}(x_n, y)$ 3 $\mathbf{s} \leftarrow \frac{\text{floor}(\text{SizeOf}(\mathbf{p}))}{\text{ceil}(\text{Log}(\text{SizeOf}(\mathbf{p})))}$ 4 $\mathbf{m} \leftarrow \text{Mean}(\forall s \in \mathbf{s}) \wedge \text{Mean} \neq 0$ 5 $\mathbf{L} \leftarrow k - \text{means}(\mathbf{p}, \text{Size of}(\mathbf{m}))$ 6 $t \leftarrow \min(\text{Mean}(\forall \mathbf{l} \in \mathbf{L}))$ **else** $\mathbf{K}_{(i)} \leftarrow$ crop vertical strip beginning at column idx until last column.

go to 1 until 6

 $false$ $idx \leftarrow idx + \Upsilon$ $i \leftarrow i + 1$ **for** $m \leftarrow 1$ **to** i **do** $Ridx \leftarrow$ All row indices $\in \mathbf{K}_{(m)}$ that contain t pixels at maximum. $Ridx \leftarrow \mathbf{Unique}(Ridx)$

page, we perform an election process among all elements of all local sets. A row index in a document image is marked as a probable border line on the page level, if all rows with the same index in all sub-image profiles; either contains no foreground pixels at all, or contains a small number of pixels determined by a user adjustable tolerance parameter η , that is dedicated to introduce the amount of overlapping and touching of components across the document page. The parameter takes values between $\eta = 0$ (if there is no overlapping/touching strokes) and $\eta = 1$ (in case of a very high number of overlapping/touching strokes), which is especially true for historical documents.

FIG. 4.2, gives an overall illustration of the proposed method, where FIG. 4.2 (a), (b), and (c), are the skew free binary source, the sub-images along with their

corresponding projection profiles, and the result with a $\eta = 0.5$ tolerance value, respectively. Moreover, a detailed description of the approach algorithm is given in Alg. 4.1. As conclusive remarks, even though the proposed method is based on the common projection profile technique, yet, unlike available approaches, projection profile analysis is actually performed to detect the border lines between text lines rather than searching for text lines. This process proved to be relatively less expensive and at the same time robust to various types of handwritten documents.

In FIG. 4.3 and FIG. 4.4, the reader can see two different types of handwritten samples taken from the IESK-arDB database, where FIG. 4.3 (a) and (b), respectively, illustrate the source and the discovered text lines with a zero tolerance value. Considering its complexity, FIG. 4.4, depicts the result when applying the method on a medieval handwritten document with $\eta = 1$ tolerance value.

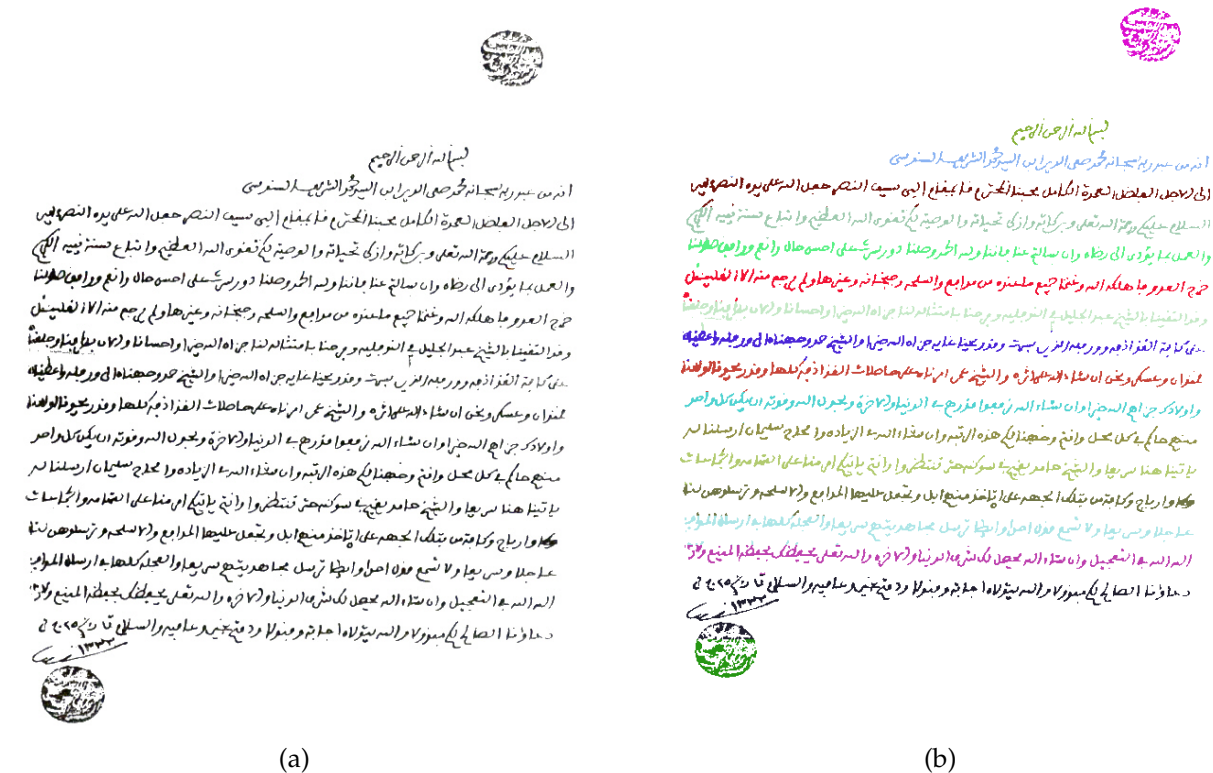


FIG. 4.3. Performance on a handwritten signed and stamped document: (a) A binary image of document written in the begin of the 20th century, (b) the result of the proposed approach with each text lines is individually identified with different colors.



FIG. 4.4. Performance on handwritten historical manuscripts: (a) A source image of a medieval manuscript, (b) the best result obtained at tolerance values $\eta = 1$.

4.2 Segmentation of Arabic Handwritten Words

To segment handwritten words into letters, we propose a two-phase methodology. In the first phase, the issue of pre-processing is considered. In which fundamental pre-processing steps e.g. filtering, binarization, as well as pre-processing issues specific to handwriting are performed.

The segmentation process is taking place in the second phase, which starts by resolving the sub-words overlapping issue that may occur whenever multiple sub-words (in the same word or between two consecutive words) vertically overlapping. Then a heuristic based voting procedure is followed to elect the most probable segmentation point between two letters.

4.2.1 Handwriting Specific Pre-processing

To suppress noisy pixels whilst preserving edges, first a median filter is applied on the handwritten word images. Because of the extraction and binarization processes, issues like smoothing out outcropping pixels and small holes filling should be dealt with. Morphological based operations such as *Closing* and combination of *Opening* and *Reconstruction* are employed respectively to solve for those issues.

Furthermore, to reduce the amount of information to be processed, to the minimum necessary for conducting our segmentation, and to ease the process of extraction of features points, thinning operation is applied on the enhanced binary images according to [42]. Then, offline handwriting specific pre-processing procedures are carried out, which will be detailed in the following sub-section.

4.2.1.1 Skew Correction Approach for Handwritten Arabic Words

Skew correction and baseline detection are proven to be of critical importance for segmentation of handwritten Arabic text. Various techniques have been reported in the literature, each has its pros and cons. HT is one of such methods that are relatively insensitive to noise and tolerates gaps within Arabic words [60]. As for the baseline detection, HT is insensitive to line direction. Consequently, it performs badly when the longest stroke is not parallel to the word baseline.

Another method is based on the Local Minima Regression (LMR) of the word skeleton [102]. Benefiting from the fact that most of the local minima points along the skeleton usually occur on or near of the baseline; the problem of finding the baseline can be reduced to a linear fitting problem of local minima points. Even though LMR is not as accurate as HT, its main advantage is the relatively insensitivity to stroke direction that is not parallel to the baseline. Furthermore, it is being noticed that reducing the Domain of HT's θ parameter according to a priori direction estimation, firstly increases accuracy and, secondly reduces the computational cost. Thus we propose an HT based technique combined with an LMR for baseline estimation, where the LMR is used for the priori estimation of the search domain of HT's θ parameter.

The first step in the proposed technique starts by calculating the fitting line of local minima points according to Eq. 4.1, and Eq. 4.2, then the slope angle α of the fitted line is calculated according to Eq. 4.3

$$y = a + bx, \quad (4.1)$$

where a, b coefficients calculated as follow,

$$b = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}, \quad a = \bar{y} - b\bar{x}, \quad (4.2)$$

where \bar{x} and \bar{y} are the statistical means of x and y coordinates respectively.

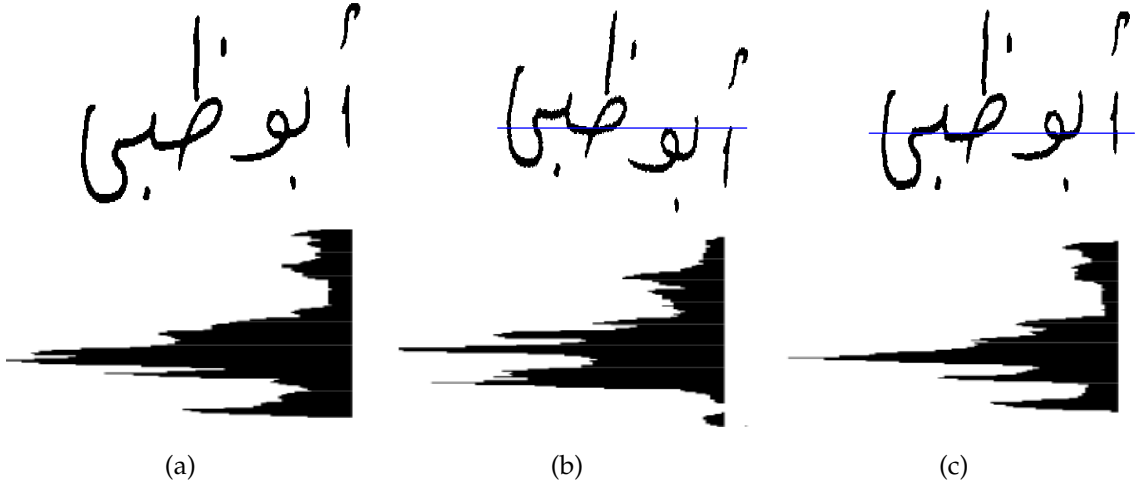


FIG. 4.5. Skew correction and baseline estimation: (a) Original binary image and its corresponding horizontal projection profile, (b) corrected version using LMR only, (c) corrected version using the proposed combination of HT and LMR techniques.

$$\alpha = \arctan(b) \quad (4.3)$$

The second step is to calculate the baseline using the HT, given the θ restricted domain. We first discretize the θ and ρ parameters and then for each point (x_i, y_i) in the image space we calculate $\hat{\rho}$ as stated in Eq. 4.4:

$$\hat{\rho} = x_i \sin \hat{\theta} + y_i \cos \hat{\theta} \quad \forall \hat{\theta} \in [\alpha - \epsilon, \alpha + \epsilon] \quad (4.4)$$

Where ϵ is a constant used to offset the random error that may be produced in the first step. Experimentally, we found that $\epsilon = 10^\circ$ gives the most accurate results. Next, each point in the image space will vote for bins that could have generated it in the hough accumulator A , and votes will be accumulated according to Eq. 4.5

$$A(\hat{\rho}, \hat{\theta}) = A(\hat{\rho}, \hat{\theta}) + 1 \quad (4.5)$$

Finally, $\hat{\rho}$ and $\hat{\theta}$ with the maximum number (global maxima) of votes will be considered as the parameters of the word baseline as shown in Eq. 4.6.

$$\arg \max_{\hat{\rho}, \hat{\theta}} A(\hat{\rho}, \hat{\theta}) \quad (4.6)$$

FIG. 4.5, shows an example of the results, where FIG. 4.5 (a) is the original image, FIG. 4.5 (b) is the skew corrected image and the estimated baseline according to

LMR only. FIG. 4.5 (c) shows the result of the skew correction and the estimated baseline of the word using the proposed technique. The reader can clearly see the improvement.

4.2.2 Segmentation of Handwritten Words into Letters

As mentioned above our segmentation approach conducted in two steps. In the first step, a careful analysis of the x-axis coordinates of the connected components is performed. The result is words images with resolved sub-word overlapping. The second step is to perform topological feature based segmentation for characters representatives. Before detailing our approach, it is helpful to start by recalling some definitions that are thought to be necessary for the clarity of subsequent definitions and notations.

Firstly, let p refer to any foreground pixel in the thinned word image $g(x, y)$, and let $N_8(p)$ denote the 8-neighborhood set of p . Secondly, by examine each $p \in g(x, y)$ a set of feature points are identified, which we call Critical Feature Points (CFPs). CFPs set contains further four subset that are to be defined below:

- i. The first subset is End Points (EP) appearing in blue in FIG.4.6 (a), which contains all pixels with only one pixel in its 8-neighborhood set.

$$EP = \{p | N_8(p) = 1\} \quad (4.7)$$

- ii. The second subset is Branch Points (BP) shown in green in FIG.4.6 (a), which contains all pixels with 3 to 4 pixels in their 8-neighborhood.

$$BP = \{p | N_8(p) = 3 \vee p | N_8(p) = 4\} \quad (4.8)$$

- iii. The third subset is Dot Points (DPs) (appear in cyan), which is the union of the set of all isolated pixels, and the set of pixels that belong to Connected Components (CC) that are less in size than an adaptive threshold T proportional to the estimated character size calculated upon the thinned text image.

$$DP = \{p | N_8(p) = 0\} \cup \{p | p \in CC \wedge size(CC) < T\} \quad (4.9)$$

- iv. The fourth and last subset is the Loop Points (LP) (drawn in red), which are all O_n remained pixels of the thinned text image after performing the flood-fill

algorithm [103].

$$LP = \{p | p \in ff(i(x, y))\} \quad (4.10)$$

Given the aforementioned four subsets, the CFPs set can be defined as the union of all the four subsets. FIG. 4.6 (a), depicts the four different kind of points, and FIG. 4.6 (b), shows a thinned word image with the all possible CFPs.

$$CFPs = \cup\{LP, EP, BP, DP\} \quad (4.11)$$

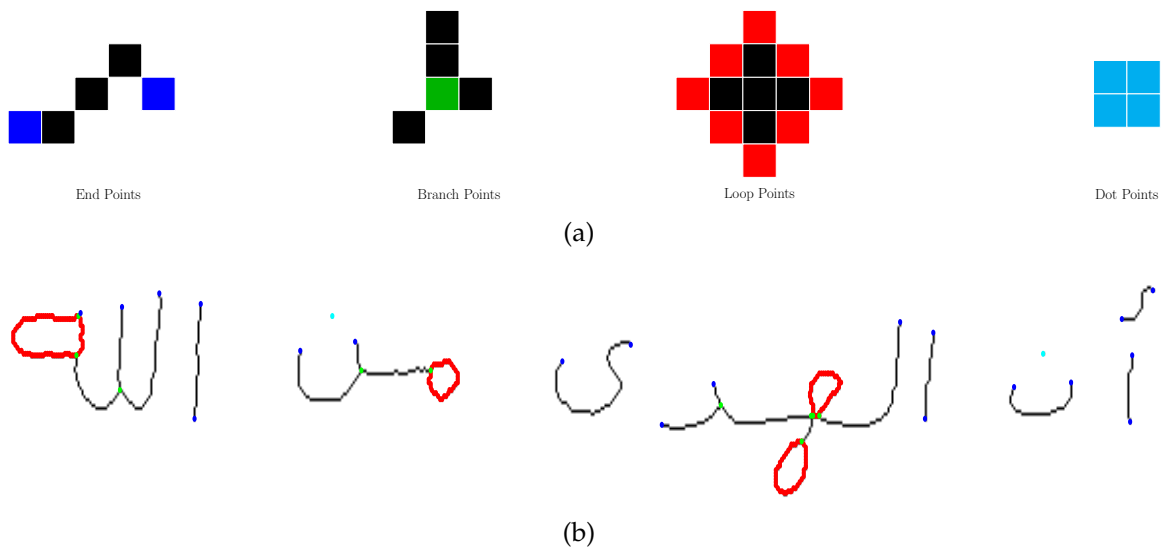


FIG. 4.6. Critical feature points CFPs: (a) Different types of CFPs, (b) a thinned handwriting image with all possible CFPs.

In the subsequent subsection, we will first explain the proposed solution to the problem of sub-words overlapping, and then address the problem of letters segmentation by utilizing the word skeleton CFPs.

4.2.2.1 Resolving Sub-words Overlapping

For resolving the sub-words overlapping, we first find the word baseline as stated above. Then upon finding the baseline, we differentiate between two types of connected components (CC). The first is what we call main CCs, which are all CCs that are intersecting with the baseline y -coordinate. The second are what we call auxiliary CCs, which are all CCs that are not intersecting the baseline y -coordinate. FIG. 4.7 (a) shows an example of word image, where the main CCs are numbered as (1,2,4,6), while the auxiliaries are (3,5,7,8) and the horizontal blue line is representing the word baseline.

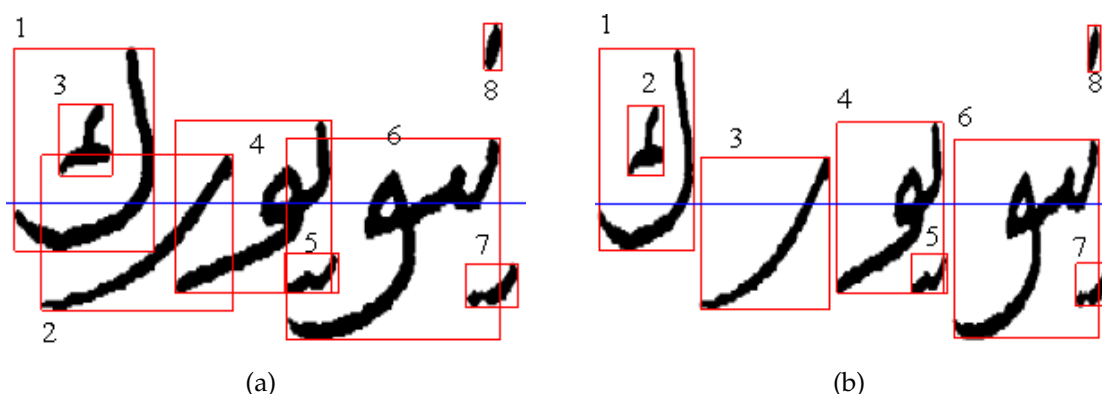


FIG. 4.7. Sub-words Overlapping: (a) Overlapped sub-words within a word baseline, (b) the overlapping free version.

After identifying the main CCs, we conduct a distance analysis on their bounding boxes along the x-axis, in order to identify the baseline overlapped main CCs and their corresponding overlapping distances. In FIG. 4.7 (a) for example main CCs that are overlapping are (1,2), (2,4), and (4,6). Another distance analysis is performed against the auxiliary CCs, so each auxiliary CC can be assigned to its corresponding main CC according to the following rules:

- i. If an auxiliary CC is overlapping only one main CC along the x-axis, then assign the auxiliary to this main CC. So in FIG. 4.7 (a) for example auxiliaries number (8, 7) will be assigned to the main CC number "6".
- ii. If an auxiliary that is above the baseline is completely contained in the bounding box of a main CC, then assign the auxiliary to the main regardless of any main CC that may overlap it along the x-axis. So "3" will be assigned to "1".
- iii. If two main CCs are overlapping an auxiliary under baseline, like in case of "5" that is overlapped both "4" and "6", we calculate the absolute distance along y-axis, between lower bounding box of the auxiliary, and the lower bounding box of the overlapping main CCs, the one with minimal distance wins the auxiliary. So "4" wins "5".
- iv. In case of an auxiliary is above the baseline and overlapping multiple main CCs, the absolute distance along y-axis is measured between its lower bounding

box y -coordinate and the upper y -coordinates of the overlapping main CCs bounding box; and the main CC with the minimum distance wins.

Even though the aforementioned rules resolve almost all the cases, there are some extreme cases where auxiliaries are not overlapping any main CC. In these cases, auxiliary is assigned to the direct next main CC on the left². After assigning the auxiliaries to their corresponding main CCs, we computed the sub-words borders along the x -axis against all its elements (auxiliary CCs and main CCs). The left border of the sub-word bounding box, is then computed to be the farthest left border among all sub-word elements. Likewise, the right border is selected to be the farthest border to the right. Eventually, a final distance analysis is performed against the new sub-word borders and the overlapping is solved by shifting away the overlapped sub-words. FIG. 4.7 (b) shows an examples of the overlapping free version of FIG. 4.7 (a). As a result of this pre-segmentation step, sub-words are separated by enough number of empty columns that make their segmentation relatively an easier process.

4.2.2.2 Segmentation of Letter Representative

After resolving the sub-words overlapping, the proposed approach starts the segmentation of sub-words into their constituent characters representatives. Given that Arabic characters have their boundaries in columns with the minimum number of pixels (only one pixel in the thinned version), our segmentation approach starts by generating a set C of columns indices as candidates for segmentation, where the elements of C are all columns indices within the thinned image $g(x, y)$, containing only one foreground pixel. In our approach, we adopted the segmentation algorithm presented in [23] as the basis of our proposed segmentation algorithm. The algorithm involves a broader set of segmentation candidates instead of using the set of contour local minima only. Using Local minima only as candidates for segmentation is usually resulted in inaccurate segmentation because local minima often occur inside many of Arabic letters main bodies. Hence, we decided to generate a large set of segmentation candidates, and we did not restrict our candidates to only local minima.

²This is due to the fact that Arabic text is written from right to left, and writers usually writing main CC first then auxiliaries. As a result auxiliaries are appearing shifted to the right away from their correspondence sub-words.

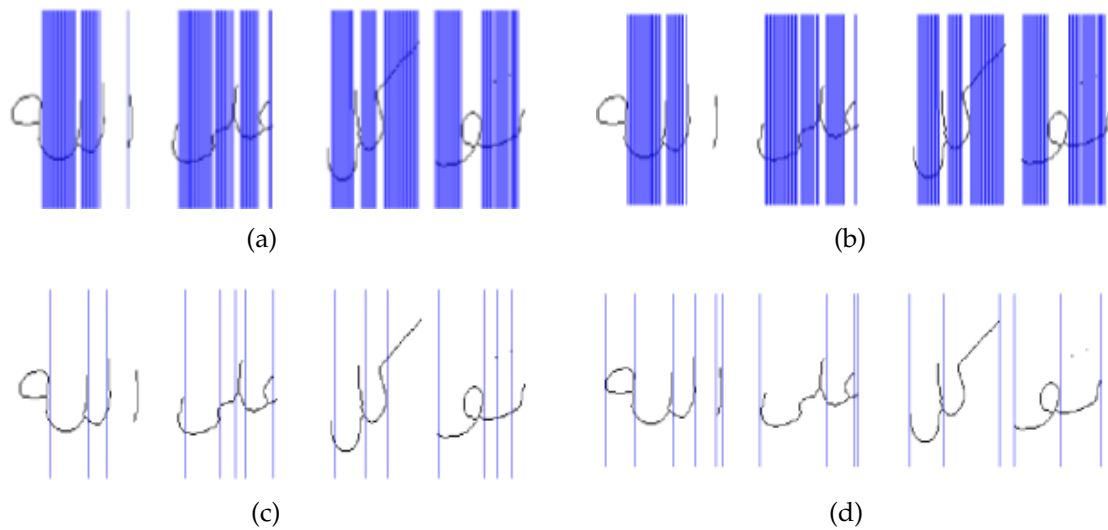


FIG. 4.8. Handwriting segmentation: (a) All possible cut candidates, (b) candidates after excluding column that contains EP, LP, BP or DP, (c) candidates after excluding candidates with the direct left neighbor, (d) the result after applying the proposed heuristic rules.

FIG. 4.8 (a) is depicting the set of all segmentation columns candidates for a handwriting image. The reader can notice that each column containing only one pixel is elected as a candidate. The next step is to exclude from the candidates set all columns that are intersecting with any CFPs point, this is to say that LP, EP, BP and DP columns cannot be in the same time a cut point otherwise we lose meaningful character features. FIG. 4.8 (b) shows the text image after excluding columns candidates that are intersecting CFPs columns. The reader can also see here in FIG. 4.8 (b) that very few candidates are excluded compared with FIG. 4.8 (a). This is because it is quite seldom that an LP or a BP column contains only one pixel, so it will not be chosen as a candidate in the first place.

The next step is to scan the list of candidates, starting from the most right one to the left, electing the left neighbor from each two adjacent segmentation candidates. FIG.4.8 (c), shows the result, in which we can easily notice the significant reduction in the number of candidates for segmentation. The candidates set obtained so far is an important improvement over the previous candidate sets. However, in order to resolve issues such as the over-segmentation in FIG. 4.8 (c) (*the letter ت (TAA), the first letter from the right is over-segmented into three parts*), we formulate four heuristic conditions to increase segmentation accuracy.

To ease the notation of these conditions, we will write m as a subscript of CFP

and CFPs elements to refer to the respective column index. Also, we will use c_i, c_j to refer to any two column indices from the handwriting image, that are chosen to be candidates. Then, the final election process is performed by applying the following conditions on the segmentation candidates:

- i. If there are two consecutive cut candidates and there is no CFPs point in between then delete from the list the one on the right, this condition can be formulated as following;

$$\forall \{c_i, c_j\} \in \{C\} | c_i > c_j, \text{ if } \{CFP_m\} \notin [c_i, c_j] \Rightarrow c_i \notin C \quad (4.12)$$

The result of applying this condition on the candidates depicted in FIG. 4.8 (b), and can be seen in FIG.4.8 (c).

- ii. If there is a BP column or LP column before encountering another candidate then we elect the candidate as a cut point, the notation version of the condition is:

$$\forall \{c_i, c_j\} \in \{C\} | c_i > c_j, \text{ if } \exists (BP_m \vee LP_m) \in [c_j, c_i] \Rightarrow c_i \in C \quad (4.13)$$

This condition is applied simultaneously with condition (i), and its effects can be noticed in FIG. 4.8 (c), where all candidates having BPs or LPs on the left are kept.

- iii. If the direct neighbor on the left is a column that contains a DP point, then delete the candidate from the list. This can be notated as follows:

$$\forall c_i \in C \text{ if } \exists (c_{i+1}) \in DP_m \Rightarrow c_i \notin C \quad (4.14)$$

where DP_m , is the set of columns contain DP pixels. The effect of this condition is visible when comparing FIG. 4.8 (c) and FIG. 4.8 (d), where the first two candidates from the right in FIG. 4.8 (c) are deleted in FIG.4.8 (d), since their direct neighbors to the left are columns contain DPs pixels.

- iv. If the next column contains an end point EP_1 , which is at the same time not an end of stroke, then flow the contour starting from EP_1 down to the left. If another end point EP_2 is encountered (before BP or LP) which is not on the contour and is an end of the stroke, then elect the candidate as a cut point.

This condition solves few cases when the baseline does not intersect all sub-words within a word, leading to a failure in the detection of sub-words vertical overlapping as depicted in FIG. 4.9. It is also worth mentioning that in such cases, we choose to keep candidates at the Begin of the strokes, since the Begin of a stroke is usually more representative than its End.

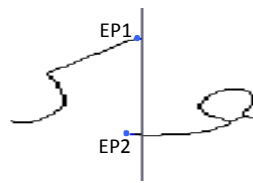


FIG. 4.9. Segmentation of Sub-words vertical overlapping.

Moreover, to mark the start and the end of all letters, we insert segmentation candidates direct before and after each main CC. Finally, each set of pixels between every two segmentation candidates in the binary image, are assumed to represent a letter in the word. To extract the most possible accurate letter image, and also to eliminate isolated pixels (or group of pixels) belonging to neighboring letters that may appear as a result of the crop process, we perform a reconstruction module according to [104], in which, we use those sets of pixels as masks, and their counterparts in the thinned image as a marker. Then we constructed the letter-image and save it, as the result of our segmentation methodology. FIG. 4.10 (a) and (b), illustrate the performance of the suggested approach on several samples taken from two well databases namely the IESK-arDB and the IFN/ENIT. Results prove the efficiency of the approach on a single word as well as on sentences of multiple words. Comparison to other methods and further analysis of the approach will be given in the experimental results chapter.

4.3 Conclusion

The main contribution of this chapter was to propose a solution for the challenging problem of handwriting segmentation, which has different aspects, each can be a research topic on its own. Given the fundamental importance of handwritten text line segmentation, we proposed a solution that is simple, unspecific to the handwriting type, and computation inexpensive. The approach is a top-down

Binary	Thinned	Segmented	Binary	Thinned	Segmented
مؤقت		مؤقت	تطاوينا 7 نوفمبر		تطاوينا 7 نوفمبر
الليلة		الليلة	هتلين		هتلين
انتحاري		انتحاري	القصور		القصور
الانترنت		الانترنت	قبلا ط		قبلا ط
من		من	المنزلة		المنزلة

(a)

(b)

FIG. 4.10. Samples results of the proposed segmentation approach: (a) results of samples taken from the IESK-arDB database. (b) results of samples taken from IFN/ENIT database.

based, begins with a page skew correction then performs a voting process upon a set of horizontal profiles calculated from a set of sub-images cropped from the page image. The approach achieved very good results on modern handwriting as well as on historical manuscripts. The second and most important segmentation issue that we addressed is the segmentation of handwritten Arabic words into their constituent letters.

Unlike in Latin based scripts, in Arabic sub-words/words overlapping issue should be handled first before any segmentation attempt. Thus, we introduced an overlapping resolving module that is carried out before segmentation takes place. This prior procedure is facilitating the letter segmentation process by inserting enough empty columns between overlapped sub-words/words. The segmentation process is then conducted by applying a set of heuristics that is formulated upon a group of pixels called critical feature points. Performance results on samples from two different databases confirm the efficiency of the proposed method. Throughout our research on the problem of Arabic handwriting segmentation, we primarily focused on the issue of word into letters segmentation and page to text lines segmentation. For the segmentation of a text line into words, we relied on solutions presented in [96, 105], which found to perform adequately. Given the presented solutions for the segmentation, in the next chapter, we are going to introduce an unconstrained generative-based recognition methodology for the handwritten Arabic words.

GENERATIVE & DISCRIMINATIVE BASED APPROACHES FOR RECOGNITION OF HANDWRITTEN ARABIC WORDS

GENERALLY speaking, the formulation of the classification problem as a sequence labeling problem significantly improves the system accuracy [106, 107]. This is because of the fact, that unlimited number (theoretically at least) of nearby elements can be used in the process of selecting an "optimal" label for a given element. Such a fact motivated us to approach the problem of offline Arabic handwriting recognition as sequence labeling task. Consequently, the main theme of the current chapter is to introduce probabilistic based classification approaches suitable for labeling sequences of features extracted from offline handwritten Arabic words. In this context, and as we previously discussed in Chapter 2, there are two main paradigms to follow, namely, generative and discriminative classification.

The first part of this chapter, will propose a generative based Hidden Markov Models (HMMs) approach built directly on top of an explicit handwritten Arabic words segmentation module, previously described in Chapter 4. The suggested system uses two sets of sequential features to describe the shape of the segmented unit along two different directions, namely, clockwise and counter clockwise. Unlike previous HMMs based offline handwriting recognition approaches, the main

novelties of our HMMs based approach include the following: (i) rather than relying on an implicit segmentation as a by-product of the decoding process of HMMs, we integrate an explicit segmentation phase into the recognition approach to segment word down into letter representatives, (ii) instead of using traditional sliding windows based features, we extract two sets of sequential shape representative features from the segmented letter¹ after size normalization and thinning, and (iii) for classification, we use a left-to-right banded HMMs equipped with an adaptive threshold model constructed from the trained models of all letters.

To label a sequence of features, in the second part of the current chapter, we present two probabilistic discriminative based classifiers, namely, the linear-chain Conditional Random Fields (CRFs) and its extension the Hidden-state CRFs (HCRFs). Generally, CRFs and HCRFs classification models have been successfully applied in a number of sequence labeling problems, e.g. natural language processing, computer vision, and bioinformatics [108]. This is because, firstly, they make no dependence assumptions among the input data, and secondly they allow to represent complex relations between to-be-predicted labels and the observed data.

As revealed from literature, there is a dearth of prior works that are proposing CRFs based solutions for the problem of offline Arabic handwriting recognition. Therefore, we claim that we present for the first time CRFs and HCRFs based solutions to the problem. Similar to our generative HMMs based approach, the proposed CRFs and HCRFs approaches integrate the explicit segmentation approach described previously in Chapter 2.4 and also to reduce the number of class labels in CRFs and HCRFs models², the letter taxonomization module is adopted.

As a side remark, we point out that the current chapter is solely devoted to describe the proposed methodologies. In the next chapter, the performance analysis of the proposed systems will be described and the results obtained using the generative approach will be compared to those of the discriminative approaches.

¹As a result of erroneous segmentation, non-representative segments may be obtained, so from now on we will use only segment or stroke, interchangeably, instead of segmented letter.

²CRFs and HCRFs, typically use a single exponential model to represent the joint probability of a sequence of labels given an observation sequence. Learning CRFs and HCRFS parameters for models with large number of labels is an intractable problem.

5.1 Sequential Shape Descriptions Features for Arabic Handwriting

Basically, features might be chosen to represent either the stroke external characteristics, i.e. the boundary, or to represent the internal characteristics that are the pixels contained within the stroke's region. External representation is usually more appropriate when focusing on the shape characteristics, whereas internal representation is the better choice when the focus is on characteristics relevant to color or texture [50]. Related literature contains many different types of features that might be geometric (e.g., geometric moments, directional histograms, etc.), structural features (e.g., topological features, Fourier's descriptors, etc.), or space transformation features (e.g., principal component analysis, linear discriminant analysis, etc.) [1]. Due to the fact that HMMs, CRFs, and HCRFs are especially powerful for the task of sequential feature classification, features extracted from a letter or a word image should be sequential or can be easily converted into a sequence that eventually passed to HMMs, CRFs, or HCRFs classifier as input. The most widely adopted sequential features for HMMs, CRFs, and HCRFs handwriting recognition systems, are those extracted using the principle of the so-called sliding-window [30, 31, 109]. Typically, these types of features are sequences of observations extracted by shifting a window along the image of the word from right to left or vice versa.

In case of Arabic handwriting, the sliding-window is shifted with small distance from the right to the left, and for each position a feature vector is extracted [110]. Sliding-window features can be categorized into two different groups, the first group is the pixel-distributions based features which encompass all features describing the distribution of foreground pixels within a given window, e.g., the normalized density of foreground pixels, the number of black/white transitions, etc. The second group is the concavity features that include all features that provide information relevant to the pixel structure arrangement inside the sliding window. When using sliding-window based features, typically, a large number of features are primarily extracted. Then, to get rid of redundant and irrelevant features, feature selection or reduction algorithms are usually needed to allow the recognition process to be computationally tractable. Furthermore, both distribution and concavity sliding-window based features are particularly adequate to represent the image

local patches (local details), rather than capturing shape global characteristics (e.g. contour curvature) which prove to be very important for handwriting recognition. Additionally, being computed based on pixel's connectivity at local image neighborhoods, sliding-window features are sensitive to the stroke width and relatively more prone to multiple handwriting distortions (e.g. slant, skew, etc.). Inspired by the works of [111–113], we propose a robust yet simple approach for extracting two sets of shape descriptor features, that are proper to be used as input for a sequence labeling based recognition system. Besides avoiding the above-mentioned drawbacks of sliding-window features, the proposed features have a number of desirable characteristics such as;

- Less expensive to extract and to process.
- Capturing the letters distinctive shape characteristics.
- Invariant to stroke width, and less sensitive to handwriting distortions.
- Easily converted to vectors of observations suitable for sequence classifiers.

5.1.1 Extraction of Features Descriptors

Since our main objective here is to build a robust and an unconstrained recognition system for handwritten words, our feature extraction approach is built on top of the explicit segmentation methodology detailed in Chapter 4. Providing that writing styles differ greatly with respect to height, width, skew, and slant, feature extraction begins by normalizing the handwritten word against handwriting deformations, i.e. skew and slant. Then, to further minimize the within-class variations, segmented images are size-normalized while preserving the segment aspect ratio. For size normalization, a backward linear normalization method is employed to map the pixels coordinates of all segment images (usually of different size) into a standard plane of fixed $N \times N$ dimension where $N = 64$ is found to be optimal [114]. The core idea of our approach is illustrated in FIG. 5.1, where feature extraction starts by uniformly distributing a set $\mathbb{P} = \{p_1, p_2, \dots, p_m\}$, $p_i \in \mathbb{R}^2$, of m reference points along a rectangle that tightly contains the segment skeleton image (see FIG. 5.1 (a)). Typically m can be any natural number less than or equal to n , where n is the total number of pixels constituting the segment skeleton image. Moreover, in practice,

m should be proportional to the size of the normalized images, therefore $m = 64$ is chosen. Since the main focus of this work is on Arabic handwriting, the first reference point p_1 is positioned on the rectangle's upper-right corner. Additionally, Let $\mathbb{Q} = \{q_1, q_2, \dots, q_n\}$, $q_i \in \mathbb{R}^2$ the set of pixels coordinates constituting skeleton segment. Starting at the reference point p_1 , and in both clockwise and anticlockwise directions, for every $p_j \in \mathbb{P}$, we search for the nearest corresponding pair $q_j \in \mathbb{Q}$ (only one). Upon identifying q_j , and by using p_i as the pole, we estimate the radial distance r_{ij} , the angle φ_{ij} according to Eq. 5.1, and eventually, we exclude the pixel coordinate q_j from the segment pixel set \mathbb{Q} .

$$r_{ij} = \sqrt{(\hat{x}_j - x_i)^2 + (\hat{y}_j - y_i)^2}, \quad \varphi_{ij} = \tan^{-1} \left(\frac{\hat{y}_j - y_i}{\hat{x}_j - x_i} \right), \quad (5.1)$$

where $p_i = (x_i, y_i)$, $q_j = (\hat{x}_j, \hat{y}_j)$

Alg. 5.1, outlines the features extraction process, where each $p_i \in \mathbb{P}$ is assigned one and only one $q_j \in \mathbb{Q}$, and, as a result, two different feature descriptor vectors $\chi_a = (\chi_1, \chi_2, \dots, \chi_{64})$, $\chi_c = (\chi_1, \chi_2, \dots, \chi_{64})$, where $\chi_i = (r_i, \varphi_i)$, are constructed from every (p_i, q_j) pair in anticlockwise as well as in clockwise directions, respectively. In this context, it is also important to mention that computation is performed with the assumption that $n \geq m$, i.e., the number of pixels m in the segment skeleton image are greater than or equal to the number of reference points m .

Algorithm 5.1: Extraction of features descriptors

Data: \mathbb{P} the set of reference points, \mathbb{Q} the set of segment skeleton pixels.

Result: χ

begin

```

   $p_1 \leftarrow \text{Start Point}$ 
  for  $\forall p_i \in \mathbb{P}$  do
    for  $\forall q_j \in \mathbb{Q}$  do
       $q_j^* = \arg \min_{q_j} \|q_j - p_i\|$ 
       $r_{ij} = \|q_j^* - p_i\|$ 
       $\varphi_{ij} = \arctan(q_j^*, p_i)$ 
       $\chi_{ij} = (r_{ij}, \varphi_{ij})$ 
     $\chi += \chi_{ij}$ 
     $q_j^* \notin \mathbb{Q}$ 

```

FIG. 5.1, explains further the proposed feature extraction step, where FIG. 5.1

(a), shows an explicit segmented handwritten word, and FIG. 5.1 (b), visualizes the computation involving the reference points as well as the segment pixels.

To improve the robustness of the recognition system against various writing distortions, we computed the features in two different directions, FIG. 5.1 (c) and (d), are depicting the results along the two directions for the letter ك "KAF" (enclosed in the red rectangle) in FIG. 5.1 (a). To demonstrate the feature descriptors discriminativity, FIG. 5.2 shows the feature profile for letter "ك" (FIG. 5.1 (c)) in the two previously mentioned directions.

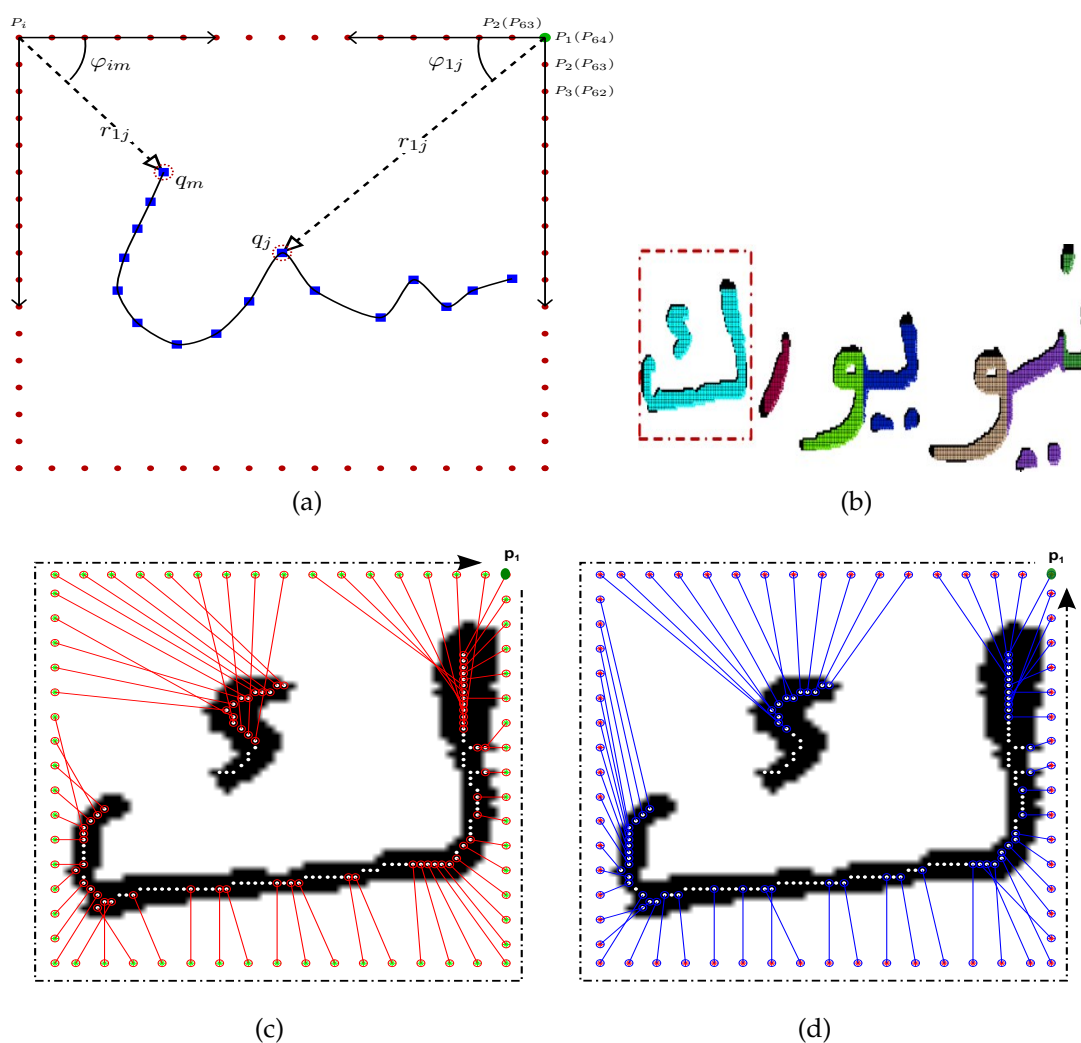


FIG. 5.1. Extraction of features descriptors: (a) Extracted features computation, (b) segmented handwritten word, (c) clockwise features, and (d) anticlockwise features.

5.1.2 Vector Quantization for Feature Sequences

Basically, quantization is the process of mapping a large number of different input values into a far smaller set of discrete values. In case of vectors of data points, most of quantization algorithms work by clustering "nearby" vectors into groups (clusters). A widely used quantization technique is the K-means clustering algorithm, due to its relatively fast convergence property [115]. Its drawback, however, is the sensitivity to initialization step, meaning that the algorithm performance is fully determined by the initial values, thus a reasonable solution can only be achieved through initialization values that lie close to a good clustering solution. To overcome this problem, we adopted a k-means initialization method proposed in [116]. In this method, authors suggest to use the affinity propagation algorithm (AP) to initialize the k-means clustering. AP works initially by assuming that all data points are potential exemplars that iteratively exchange messages until a satisfactory clustering solution is reached.

To estimate the number of clusters that adequately represent a letter without any assumptions about the feature data internal distributions, firstly, the AP's similarity matrix is constructed by calculating pairwise similarity values between each data point $s(\chi_i, \chi_k)$, $\chi \in \mathcal{X}$, where the similarity value quantifies how well χ_k suited to be the exemplar of χ_i . Like in [116], we simply define the similarity function s as the negative squared Euclidean distance between data points. Then, for each letter, the AP is applied twice, once for χ_a and once for χ_c , and the average of the estimated number of clusters is computed. This process is performed for every letter in every form, and the number of clusters (i.e the number of quantization levels) or k of k-means is calculated as the overall average of all clusters of all letters, where $k = 16$ is found to be the optimal number of quantization levels.

Eventually, a k-mean clustering algorithm with $K = 16$ is used to quantize the feature descriptors χ_a and χ_c , and as a result, we obtained two vectors of observation sequences \mathbf{f}_a and \mathbf{f}_c of length 64, containing values of observed quantization level indices, where $\mathbf{f}_a = (f_{a_1}, f_{a_2}, \dots, f_{a_{64}})$, $f_{ai} \in \{1, 2, \dots, 16\}$, representing the clockwise observed feature sequence, and $\mathbf{f}_c = (f_{c_1}, f_{c_2}, \dots, f_{c_{64}})$, $f_{ci} \in \{1, 2, \dots, 16\}$, representing the anticlockwise observed feature sequence.

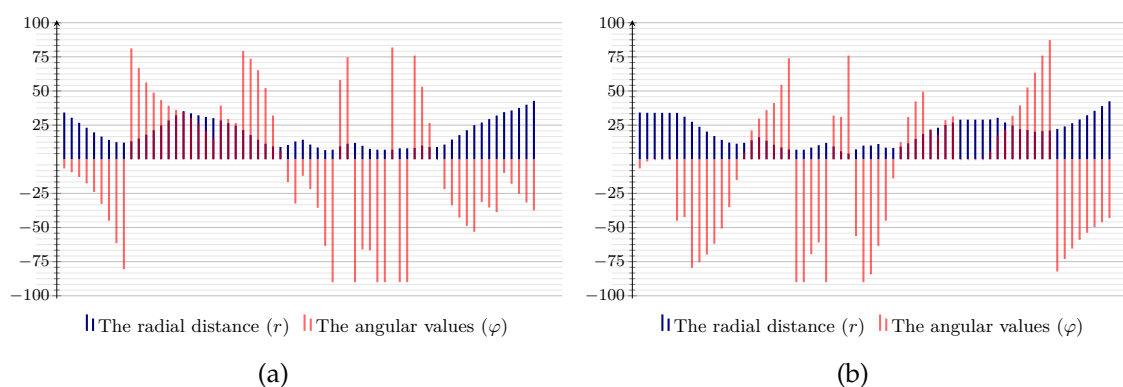


FIG. 5.2. Letter shape descriptor profiles (for the letter in FIG. 5.1 (a)): (a) The profile of clockwise shape descriptor, (b) the profile of anticlockwise shape descriptor.

5.2 HMMs based Recognition of Arabic Handwriting

This section describes in details the proposed recognition approach. Firstly, we show how the letter models are constructed and how the HMMs recognition parameters are initialized, then illustrate the process of building the threshold model and eventually presenting and discussing some recognition results. FIG.5.3, outlines the major elements of the proposed recognition system. Basically, the system is built up of two sub-classifiers corresponding to the different types of features. Each sub-classifier contains an HMMs model for each letter in each form, thereby each sub-classifier consists of 122 different HMMs models. To cope with the errors of segmentation that negatively affects the system performance, a model for each sub-classifier is constructed by ergodically connecting all other models. This model is called threshold model and is dedicated to reject out-of-vocabulary segments.

5.2.1 Shape based Letters Taxonomization

Although, the recognition of handwritten Arabic words is, firstly, reduced (through the explicit segmentation) into the problem of recognizing a segment that may represent a letter in a word. However, the Arabic alphabet contains basically 30 letters and a letter may appear in two to four distinct shape according to its position in a word, resulting in 104 substantially different shapes. A straightforward approach to alleviate this problem, is to taxonomize letter shapes according to very primitive properties. FIG. 5.4, illustrates the adopted taxonomy, which classifies letters according to the number of segment and whether they contain a loop(s) or

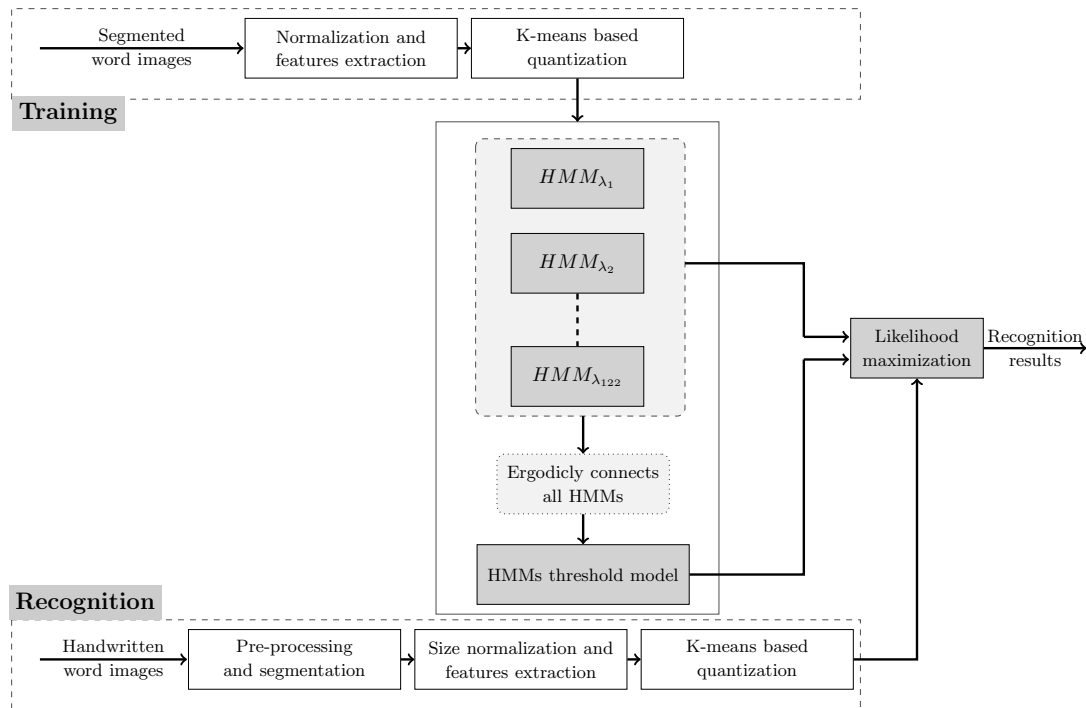


FIG. 5.3. Structure of the proposed HMMs recognition system. The system contains 104 models for letters basic shapes and 18 models for letter shapes that may appear in more than one taxonomy.

not. Thereby, for example, letters such as ح, س, ا, ر, د will be grouped under the first group from the left in FIG. 5.4 (i.e. one segment and no loop group Tax.1), letters such as و, ه, ح, م under the second group (i.e. Tax.2), letters such as ك, ث, ت, ب in the third group (Tax.3), and the fourth group (Tax.4) will contain letters such as غ, ق, ض . By following such a simple pre-classification step, we reduce the space of labels

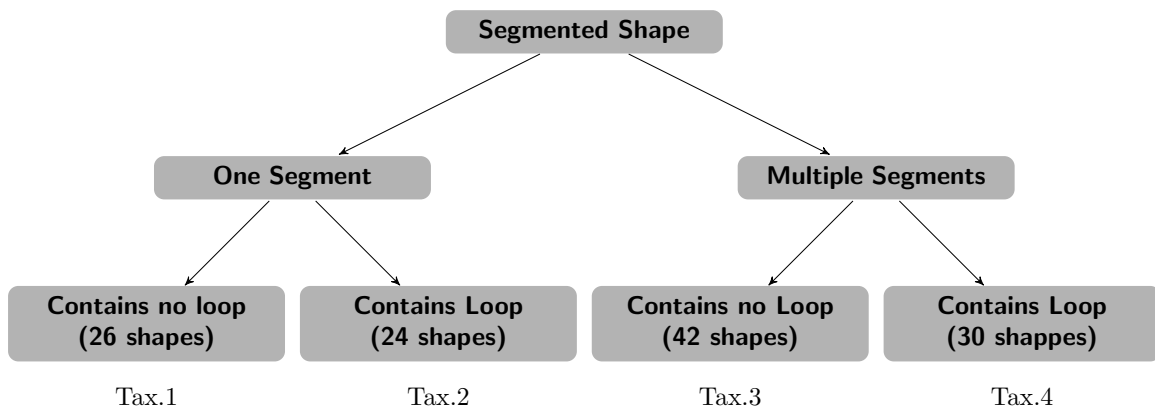


FIG. 5.4. Letters taxonomy, according to the number of segments and the existent of loop(s).

from 122 to 42 at most. Furthermore, notice that the total number of labels is 122

(not 104), because some letters appear twice in two different taxonomies, since they may be either written with or without loop (e.g., μ and ν), or in one or two strokes (e.g., ξ and κ) depending on the personal writing style and/or the font used.

5.2.2 The HMMs Topology and Hidden States Optimization

The first step in building an HMMs model is to specify the model topology and the number of model states, and keep them unchanged throughout the training phase. There are multiple different types of HMMs' topologies, however there is no theoretical framework that can be used to determine the optimum topology. However, when topologies produce similar results, then the simple one is the best, since it involves the fewest number of parameters that need to be optimized. According to [117], the most commonly used topologies in the field of optical character recognition in general, are the banded left-to-right and the left-to-right (Bakis) topologies. To assess and compare the performance of the two topologies, while the number of states is assumed to be globally fixed (i.e. eight states), dedicated HMMs models are built for three randomly chosen letters from each taxonomy. And, since, there are no boosts in performance justifying the use of Bakis' topology (see FIG. 5.5, consequently, the simpler left-to-right banded topology is adopted.

In addition to choosing the proper topology, the number of states (the model's size) for each HMMs model should be carefully determined. Typically, there are two different paradigms that might be followed to select the number of states for HMMs model [118]. According to the first, the number of states should be proportional to the number of strokes within a handwritten word or a letter. Whereas, in the second, the number of the model states is estimated as the average of the length of the observation sequence of the corresponding word or letter.

Due to the fact that we are converting features data into equal-length vectors of observation sequences, estimating the number of states based on the length of observation sequence will be equal to assigning the same number of states to all models, which can be an excessive or insufficient number for the respective letter model. Furthermore, for robust and efficient modeling, typically, letters consisting of multi segments and complex shape characteristics, require models with a bigger number of states, compared to one segment and simple shape letters. Hence, and in contrast to most related works, in which a global fixed number of

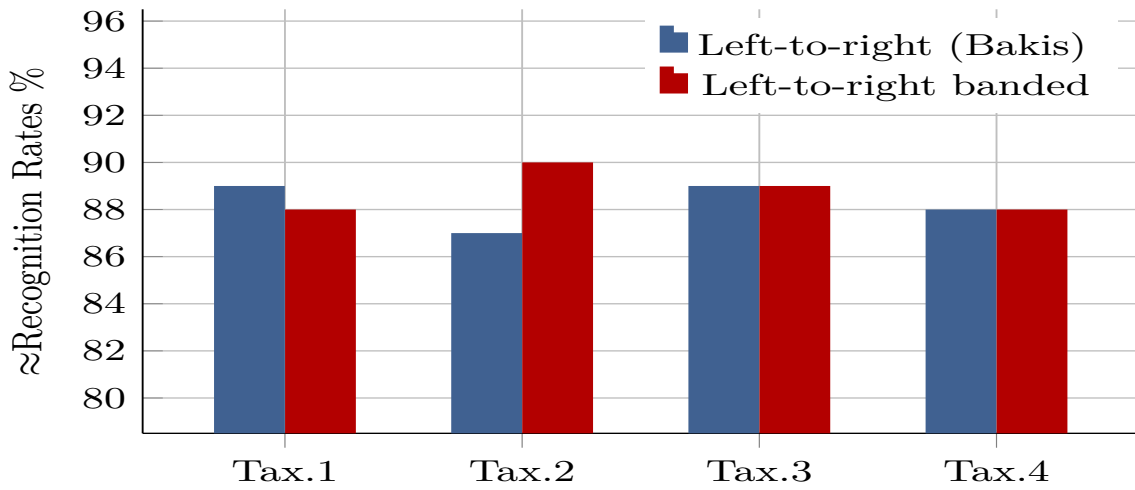


FIG. 5.5. Comparative performance of HMMs based models using Left-to-Right (Bakis) topology vs. using Left-to-Right banded topology.

states is optimized and used to model all letters, we optimized the model size for each taxonomy separately. To estimate the optimal model size for a taxonomy of letters (e.g. multiple segments with a loop taxonomy), for every letter within the taxonomy, eleven different HMMs models are created and trained, starting with a two states model up to twelve states model. Moreover, a smaller portion of data (i.e. 30%) is used for testing and the average recognition rates for all letters for every number of states are calculated. Ultimately, the number of states corresponding to the maximum recognition rate (the average) is selected as the optimal model size for all letters belonging to the considered taxonomy. Accordingly, FIG. 5.6 shows the achieved recognition rates with respect to the different number of states in each taxonomy. Letters under Tax.1 (i.e. one segment and no loop), for example, reach the optimal performance with models of size five states, and this is justifiable given their simple shape characteristics. Whereas, the relatively complex shape characteristics of Tax.3 (multiple segments and no loop) and Tax.4, required models of 10 states size to achieve the best results.

5.2.3 HMMs Parameters Initialization

In addition to optimizing the model topology and the model size, several related works [78, 119, 120] confirm the fact that proper initialization of HMMs parameters (i.e., A , B , and π) is positively affecting the overall system performance. In this context, the matrix of state transitions probabilities A is the first parameters to be

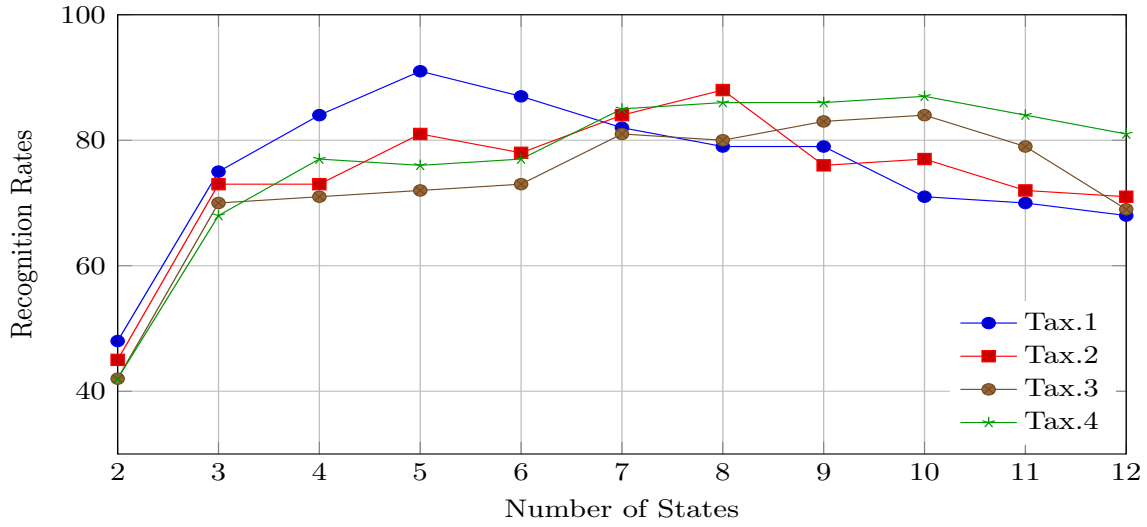


FIG. 5.6. Optimization of the HMMs model size, the best recognition rates for Tax. 1, 2, 3, and 4, are achieved with models of size 5, 8, 10, and 10, respectively.

initialized. Given that we adopted the banded left-to-right topology, Eq.5.2 shows the corresponding mathematical general structure, where the diagonal probabilities represent the self-transitions and the directly above the diagonal are used for the transition probabilities to the subsequent states.

$$A = \begin{pmatrix} a_{11} & 1 - a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & 1 - a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}, \text{ where } a_{ii} = 1 - 1/\frac{L}{N}, \quad (5.2)$$

with L and N are the length of the observation sequence and the model size, respectively. In Eq. 5.2, instead of initializing the transition probabilities with an arbitrary initial guess and subsequently applying the Baum-Welch (BW) optimization algorithm, we choose to initialize the self-transitions probabilities, so they will better represent the state duration which, in turn, will improve the model response for letters with a relatively big number of self-transition, such as ا, ب, ت, ث, ج, and etc. Then the next state transition probability is defined in terms of the self-transition probability.

The second HMMs parameter that should be initialized is the observation symbol probability distribution, which also called emission matrix B , indicating the probability of emission of a symbol (a quantization level) when the model is in

a given state. Since we adopted discrete HMMs to build the letter models, and inspired by the work of [82], we assume that every quantization level has an equal chance to be emitted by any state, hence $b_{ij} \in B$, are assigned an equal probability value, consequently the entire emission matrix is constructed according to the following equation:

$$B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1M} \\ b_{21} & b_{22} & \cdots & b_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N1} & b_{N2} & \cdots & b_{NM} \end{pmatrix}, \text{ where } b_{ij} = 1/M, \quad (5.3)$$

where M is the total number of quantization levels and N is the length of the observation sequence. Finally, the vector of the initial state probability distribution $\pi = \{\pi_i\}$, that holds the probabilities of initial states is defined as follows.

$$\pi = \left(\pi_1 \quad \pi_2 \quad \cdots \quad \pi_N \right)^T, \text{ where } \pi_1 = 1, \text{ and } \forall \pi_i = 0, \quad (5.4)$$

which implies that the process of the training will start for every model from the first state.

5.2.4 HMMs Models Construction

After selecting the topology, optimizing the number of states for each model, and reasonably estimating the parameters of the initial HMMs $\lambda = (\pi, A, B)$, an Expectation-Maximization (EM) based approach³ is used to build the letter models. For each letter, in every shape, a set of training data consists of sequences of observed features (quantization values) $\mathcal{O} = (o_1, o_2, \dots, o_T)$ is used to iteratively calculate better estimation of the initial model parameters. Using \mathcal{O} and λ , the training algorithm Alg. 5.2 begins by computing the BW forward (α) and backward (β) probabilities, where α and β are computed as previously explained in Chapter 2. Then, the EM procedure is performed on α and β probabilities to estimate the probability of various possible state sequences for generating \mathcal{O} , resulting in, two different temporary variables, $\gamma_t(i)$ which is the probability of observing \mathcal{O} in state i at time t given the HMMs intermediate parameters; and $\xi_t(i, j)$ that hold the

³The Baum-Welch algorithm (BW), which is essentially the EM algorithm applied to HMMs.

Algorithm 5.2: Models construction using an EM based approach**Data:** $\mathcal{O} = (o_1, o_2, \dots, o_T)$, observed sequence, $\lambda = (\pi, A, B)$, the initial HMM.**Result:** $\hat{\pi}, \hat{A}, \hat{B}$, the optimized parameters of HMM**begin****Step 1: Initialization** $\hat{\pi}_i \in \pi, \hat{a}_{ij} \in A, \hat{b}_j(t) \in B$ for $1 \leq i, j \leq N$ and $1 \leq t \leq T$ **Step 2: Iterative computation****repeat****Expectation – step :**

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^N \alpha_t(j)\beta_t(j)},$$

 $\xi_t(i, j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}, \quad \forall t, i, \text{ and } j, \text{ where } \alpha \text{ and } \beta$ are the *forward* and *backward* probabilities, respectively.**Maximization – step :**

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^T \gamma_t(j)};$$

$$\hat{b}_j(k) = \frac{\sum_{t=1}^T \gamma_t(j)\delta_{o_t=v_k}}{\sum_{t=1}^T \gamma_t(j)}, \text{ where } \delta_{o_t=v_k} = \begin{cases} 1, & \text{if } o_t = v_k \\ 0, & \text{otherwise} \end{cases}, \text{ and } v_k \text{ is}$$

a quantization level index.

$$\hat{\pi}_i = \gamma_1(i)$$

until convergence

probability of observing \mathcal{O} at t and $t + 1$ in states i and j , respectively, given the HMMs intermediate parameters. Moreover, in the maximization step, γ and ξ are used to refine and update the previous estimation of \hat{a} , \hat{b} , and $\hat{\pi}$. The EM based training procedure is iterated until convergence is reached, or a preset maximum number of iteration is exceeded.

In our system, training process is regarded as converged, if a variable ϵ (calculated according to Eq.5.5) is found to be less than a designated tolerance value ($\epsilon = 0.001$), which indicates minimum changes in the model parameters values. The tolerance variable is used in order to avoid unnecessary computation cost. Alternatively, training is considered as converged, if a maximum number of iterations (i.e. 500) is exceeded.

$$\sum_{i=1}^N \sum_{j=1}^N |\hat{a}_{ij} - a_{ij}| + \sum_{j=1}^N \sum_{m=1}^M |\hat{b}_{jm} - b_{jm}| < \epsilon, \quad \text{where } N \text{ is the model size and } M \text{ number of the quantization levels.} \quad (5.5)$$

size and M number of the quantization levels.

Eventually, and as a result of the training process, a total number of 122 reference models and 122 confirmation models are generated and maintained to be used in the subsequent recognition process.

5.2.5 Threshold Models Construction

As we already discussed in Chapter 4, the high variability nature of handwriting prevents a perfect solution for the segmentation problem. Consequently, segments with almost infinite shape variations may be passed for recognition. Where the HMMs based handwriting recognition will calculate scores indicating how well a given segment matches the different models. As HMMs works by maximizing the segment likelihood over all models, and usually chooses the model with the highest score, there is no way to reject an out-of-vocabulary or a meaningless segment, thereby increasing the probability of insertion errors. As a result of weak matching between meaningful segment and a model, a segment might be wrongly assigned to a model, causing the so-called substitution errors. Hence, the selection of proper threshold values is very critical for the recognition system performance. If a very low value is used as a threshold, then we risk accepting a large number of out-of-vocabulary (i.e. increase in false-positive errors), whereas, if the threshold is set to a very high value, in addition to out-of- vocabulary segments, valid segments may also be rejected (i.e. increase in false-negative errors).

Inspired by the idea of the garbage model in speech recognition [121], and a similar idea for gesture recognition [122], we propose an adaptive HMMs-based solution for the case of offline handwriting, called threshold model, that is used to calculate a likelihood threshold. Since we adopt the left-to-right banded topology, in which each model has two types of transition probabilities, namely self-transition and forward transition. In the context of handwriting recognition, the former represents an integral pattern within the modeled letter shape, whereas the latter typically represents a shared transitional pattern. Due to this trait, we built our threshold model by copying all states of all letter models, yet ergodically connecting them all in one model (FIG.5.7), hence, each state can be reach by all other states. The self-transition probabilities and emission probabilities will retain the same values as in the letter models, whereas the outgoing transition probabilities are

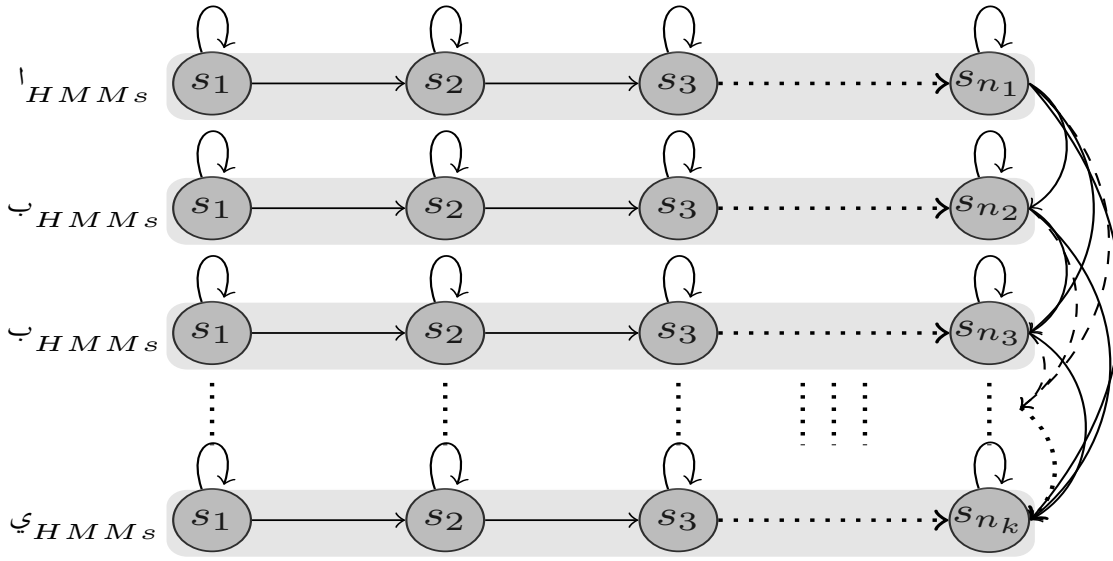


FIG. 5.7. A plain structure of the HMMs threshold model.

updated according to the following equation:

$$a_{ij} = \frac{1 - a_{ij}}{M - 1}, \quad \forall j, i, \text{ and } i \neq j, \quad (5.6)$$

where a_{ij} is the transition probability from state s_i to s_j , and M is the sum of states number over all letters. Adopting the states along with their emission probabilities and self-transition probabilities make them capable of representing any pattern of the modeled segments. Also, the ergodic connectivity allows the model to capture any random combination of sub-patterns that may result from the segmentation process. Interestingly, however, the likelihood of a modeled segment over the threshold model will be always less than that calculated against its dedicated model, since the outgoing transition probabilities are significantly reduced according to Eq.5.6. Therefore, we use the likelihood calculated upon the threshold model as an adaptive threshold, i.e, a segment is assigned a model label, if and only if its likelihood upon any model is greater than that generated from the threshold model.

In general, two different types of threshold models λ_{R_t} and λ_{C_t} have been constructed to be used for recognition with reference models and conformation models, respectively. In the next subsection, we will illustrate how the threshold models and the dedicated models are jointly used to recognize handwriting.

5.2.6 HMMs based Recognition

The proposed HMMs based recognition system is constructed by combining the aforementioned HMMs models at the decision level, as depicted in FIG. 5.8. The classifier consists mainly of two sub-classifiers, reference based classifier which is in turn composed of the dedicated models and the corresponding threshold model, and the confirmation based classifier that is similarly built of the dedicated conformation models and their threshold model.

After the simple pre-classification process (i.e. the shape-based taxonomization) and as an input, the classifier receives simultaneously two different sequences of observations, \mathcal{O}_{Fa} (anticlockwise feature representative) and \mathcal{O}_{Fc} (clockwise representative) that accordingly passed to the respective reference as well as confirmation sub-classifiers, respectively. Assigning a label to an observation sequence is usually regarded as an HMMs evaluation problem. Instead of using the common forward algorithm to solve this problem, we adopt a more efficient and less-expensive alternative, i.e. the Viterbi algorithm [77, 123], which is often used to solve the HMMs decoding problem. Typically, the forward algorithm computes the probability of an observation sequence, given a model over all possible state sequences, whereas using the Viterbi algorithm, the same probability will be estimated only over the single most likely path of states within the model. Using the Viterbi algorithm, we first estimate the most likely sequence of states \mathcal{P} , then for each model, we calculate the probability $P(\mathcal{O}, \mathcal{P}|\lambda)$, where \mathcal{O} and λ are an observation sequence and an HMMs model, respectively.

For reference and confirmation models, the recognition problem is regarded as a scoring problem, where dedicated letter models are competing and the label of the one with maximum probability will be picked as an intermediate recognition result. Before reaching a definitive recognition decision, combinations of intermediate results are performed on two levels. Firstly, an intermediate result is computed from the dedicated models and the corresponding threshold model, then the final result is estimated by combining the intermediate results of anticlockwise and clockwise based models. Strictly speaking, a segmented handwritten stroke can be successfully classified, given the following:

- $\mathcal{O}_{Fa}, \mathcal{O}_{Fc}$ anticlockwise and clockwise observation sequences, respectively,

- λ_R, λ_{Rt} set of reference models and their threshold model, respectively,
- λ_C, λ_{Ct} set of confirmation models and their equivalent threshold model, respectively.

Firstly, we use Viterbi algorithm to compute the following intermediate logarithmic probabilities⁴,

- $\mathcal{L}_R = \log \left(\max_{1 \leq i \leq N_T} [P(\mathcal{O}_{Fa}, \mathcal{P}_{R_i} | \lambda_{R_i})] \right)$, and $\mathcal{L}_{Rt} = \log (P(\mathcal{O}_{Fa}, \mathcal{P}_{Rt} | \lambda_{Rt}))$. \mathcal{L}_R is the highest probability over all reference models, where λ_{R_i} and \mathcal{P}_{R_i} are the involved model and its associated most likely path. \mathcal{L}_{Rt} is the probability of the same observation sequence computed against the reference threshold mode and N_T is the number of models of considered taxonomy.
- $\mathcal{L}_C = \log \left(\max_{1 \leq j \leq N_T} [P(\mathcal{O}_{Fc}, \mathcal{P}_{C_j} | \lambda_{C_j})] \right)$, and $\mathcal{L}_{Ct} = \log (P(\mathcal{O}_{Fc}, \mathcal{P}_{Ct} | \lambda_{Ct}))$. Similarly \mathcal{L}_C and \mathcal{L}_{Ct} are estimated as above except that confirmation models and confirmation threshold model are used instead of their reference counterparts.

Secondly, and because of the fact that confidence measure of the results is typically making the OCR systems more useful in real time applications [124, 125], the definitive recognition results are returned along with confidence values. Given the above estimated probabilities for a handwritten stroke, four different outcomes are expected depending on the following inequalities:

- (i) if $\mathcal{L}_R > \mathcal{L}_{Rt}$ and $\mathcal{L}_C > \mathcal{L}_{Ct}$, where both $\mathcal{L}_R, \mathcal{L}_C$ refer to the same label (i.e letter) in reference as well as in confirmation models, the label will be assigned to the stroke assuming complete confidence.
- (ii) if $\mathcal{L}_R > \mathcal{L}_{Rt}$ and $\mathcal{L}_C > \mathcal{L}_{Ct}$, yet $\mathcal{L}_R, \mathcal{L}_C$ point out to different labels, then the label of higher probability is assigned to the stroke, and a substitution error is reported (i.e, $S_{error} = S_{error} + 1$).
- (iii) if $\mathcal{L}_R \leq \mathcal{L}_{Rt}$ or $\mathcal{L}_C \leq \mathcal{L}_{Ct}$, then the label corresponding to the one with probability higher than that of its own threshold model, will be picked as a recognition result, and an insertion error will be reported (i.e, $I_{error} = I_{error} + 1$).

⁴Computed values can become very small, hence, logarithmic calculation is used to avoid arithmetic underflow errors.

- (iv) if $\mathcal{L}_R \leq \mathcal{L}_{Rt}$ and $\mathcal{L}_C \leq \mathcal{L}_{Ct}$, then the stroke will be rejected and a deletion error will be reported (i.e, $D_{error} = D_{error} + 1$).

As we previously mentioned, the inputs of the recognition system are observation sequences representing the various segmented strokes that constitute a given handwritten word, consequently, the individual's recognition outputs will be combined to altogether to form the recognized word. And the overall recognition confidence will be calculated as follows:

$$conf = 1 - \frac{0.5S_{error} + 0.5I_{error} + D_{error}}{\mathcal{N}} \quad (5.7)$$

where S_{error} , I_{error} are weighted at 0.5, and \mathcal{N} is total number of segmented strokes of the word. FIG. 5.8, outlines the proposed approach using an example of a segmented handwritten word (i.e. New York نیو یورك), where the symbol # is used to represent a rejected stroke (recognition or deletion error). The system output is a sequence of UNICODE letters with an overall confidence value ($conf = 0.86$).

5.3 Linear Chain CRFs based Recognition of Arabic Handwriting

In general, to predict a sequence of letter class labels \mathbf{y} for a given vector of feature observations \mathbf{x} , most of the previous related research works focused on HMMs. In order to reduce the model complexity HMMs assume conditional independence among input data, which consequently reduces the model accuracy [109]. CRFs and its extension HCRFs, are basically introduced to address this shortcoming, where dependencies are assumed among labels without presuming any kind of dependency between observation sequences [126, 127].

In our approach, we initialize with the assumption that letter class labels \mathbf{y} are fully observed, where each $y_i \in \mathbf{y}$ represents a class label of a basic shape a letter may appear in, and by using the anticlockwise based features \mathbf{f}_a as well as the clockwise based features \mathbf{f}_c , two different exponential linear-chain CRFs models are respectively created for each taxonomy, where every letter's "basic" shape under the concerned taxonomy has a corresponding state within the model. Furthermore, the proposed models are built using two different types of feature functions, i.e. transition feature functions $t(y_{i-1}, y_i, \mathbf{x}, i)$ and state/emission feature

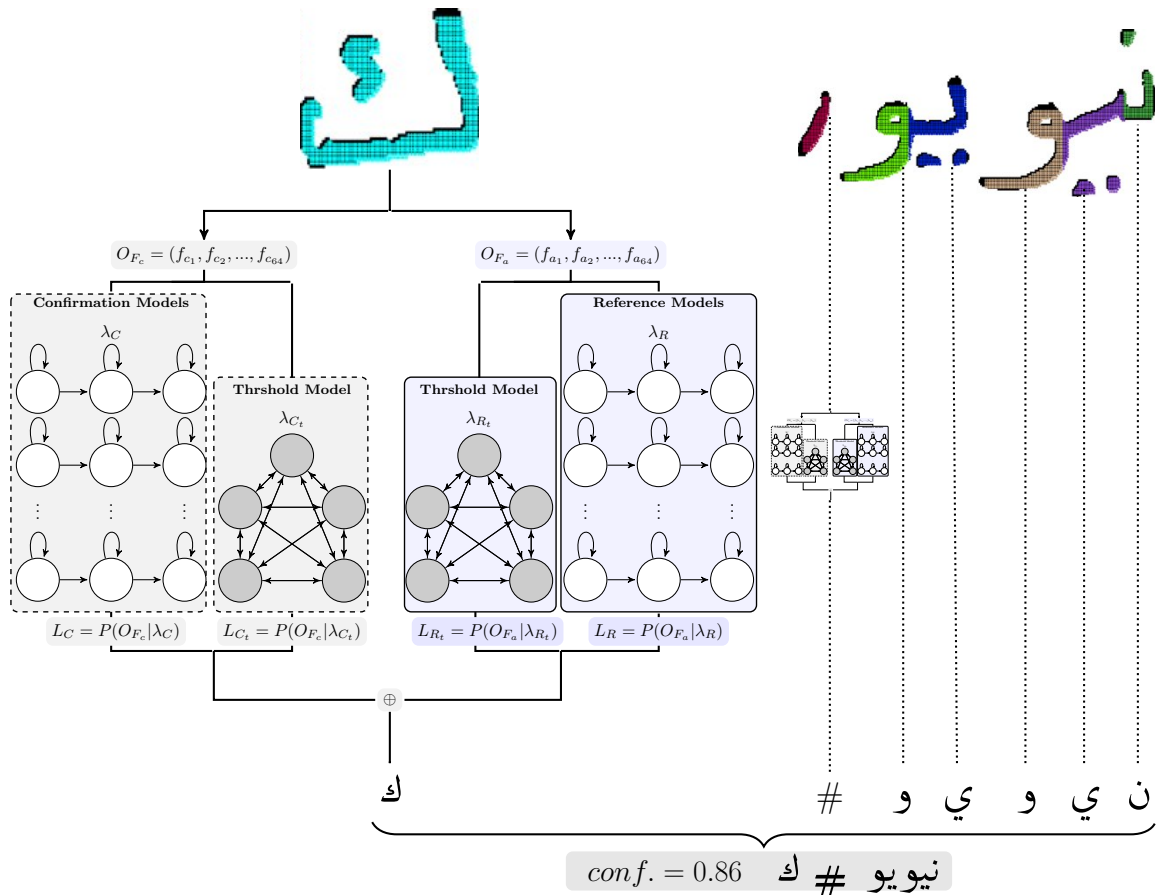


FIG. 5.8. A simplified overview of the HMMs based recognition approach, where \oplus means the combination of the results of anticlockwise and clockwise classifiers for the letter "KAF" (ك). The rest of the letters are classified similarly.

functions $s(y_i, \mathbf{x}, i)$, where these functions take, as input an entire sequence of observations \mathbf{x} , the current position within the sequence i , the current class label y_i , the previous class label y_{i-1} , and output a real-valued number.

Transition functions are typically dedicated to estimate the dependency of neighboring class labels given the value of the current position in \mathbf{x} . The state/emission functions are employed to estimate real values to represent the possibility that the current label emits the current value in \mathbf{x} . In FIG. 5.9, a simplified overview of the structure of the proposed recognition approach is shown, and the difference between transition as well as emission feature functions is illustrated. In our approach, to calculate the sequence overall likelihood, firstly, the transition and emission functions are assigned the weights λ and μ respectively, that are learnt from the training data. Then they are combined together to form a potential function as

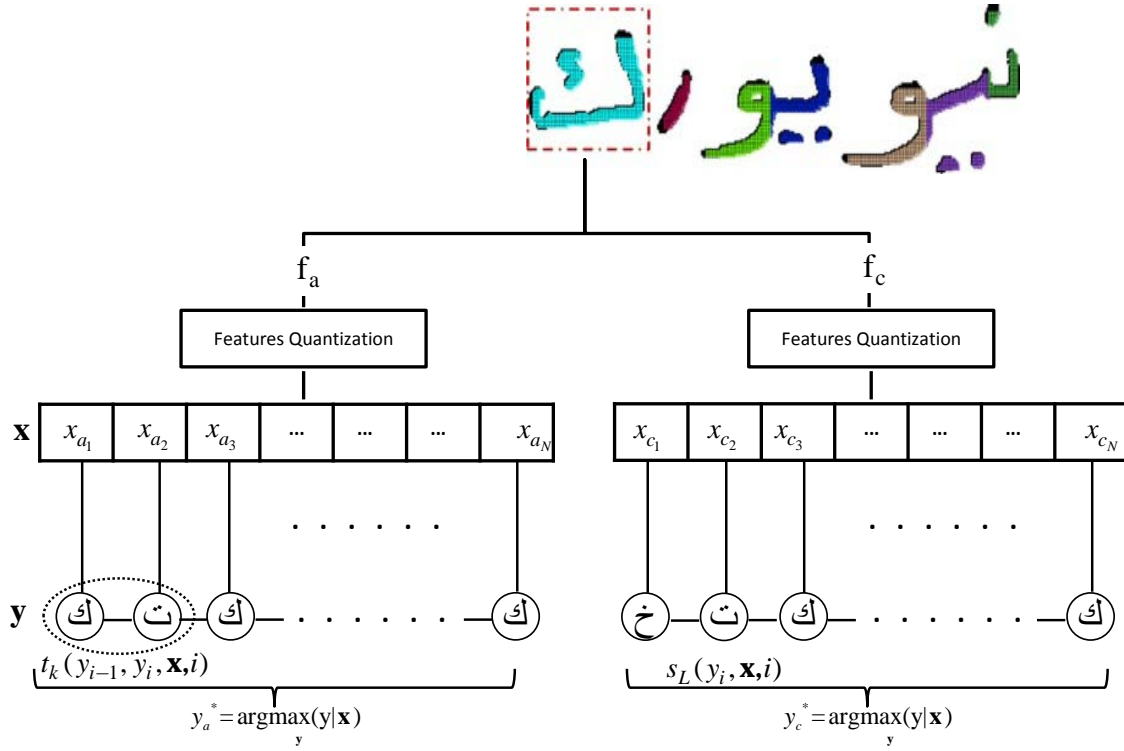


FIG. 5.9. A simplified overview of the proposed linear-chain CRFs based recognition approach.

follows:

$$F_{\theta}(y_{i-1}, y_i, \mathbf{x}, i) = \sum_f \lambda_f t_f(y_{i-1}, y_i, \mathbf{x}, i) + \sum_g \mu_g s_g(y_i, \mathbf{x}, i), \quad (5.8)$$

where $\theta = (\lambda_1, \lambda_2, \dots, \lambda_{N_f}; \mu_1, \mu_2, \dots, \mu_{N_g})$, $\lambda_i \in \boldsymbol{\lambda}$, $\mu_i \in \boldsymbol{\mu}$, and N_f and N_g are the total number of feature functions and state/emission functions, respectively. Finally, to convert the outputs of F_{θ} into proper probabilities, the F_{θ} is summed over all $x_i \in \mathbf{x}$ and the result is exponentiated and normalized as follows:

$$p_{\theta}(\mathbf{y}|\mathbf{x}) = \frac{\exp\left(\sum_{i=1}^n F_{\theta}(y_{i-1}, y_i, \mathbf{x}, i)\right)}{Z_{\theta}(\mathbf{x})}, \quad (5.9)$$

where the normalization factor $Z_{\theta}(\mathbf{x})$ is given by:

$$Z_{\theta}(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left(\sum_{i=1}^n F_{\theta}(y_{i-1}, y_i, \mathbf{x}, i)\right). \quad (5.10)$$

To investigate the effect of dependency range on the performance of the proposed system, and in addition to θ parameter, the potential function F_{θ} can be also parameterized by ω a window size parameter that define the number of previous and

subsequent observations used when predicting a class label at the current position i , (e.g., for a window size ω , the observation from $i - \omega$ to $i + \omega$ will be used to calculate the outputs of F_θ). For each taxonomy, seven linear-chain CRFs models corresponding to seven different window-sizes ($\omega = 0, \omega = 1, \dots, \omega = 6$) are trained. A comprehensive comparison of performance characteristics of different models will be given in the next chapter.

5.3.1 CRFs Parameters Learning

To learn the CRFs feature functions weights θ , we applied the popular gradient ascent algorithm on fully labeled training sequences $D = \{(\mathbf{x}^t, \mathbf{y}^t)\}_{t=1}^T$, where \mathbf{x}^t is a sequence of observed features, \mathbf{y}^t is the corresponding sequence of labels, and T is the total number of training samples. The learning process starts by randomly initializes θ , then for each training sample and for each potential function, the gradient of the log probability with respect to θ is calculated as follows

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_{t=1}^T \left(\sum_{i=1}^n \frac{\partial F_\theta(y_{i-1}^t, y_i^t, \mathbf{x}^t, i)}{\partial \theta} - \sum_{\mathbf{x}} p_\theta(\mathbf{y}|\mathbf{x}^t) \sum_{i=1}^n \frac{\partial F_\theta(y_{i-1}, y_i, \mathbf{x}^t, i)}{\partial \theta} \right), \quad (5.11)$$

where $L(\theta) = \sum_{t=1}^T \log p_\theta(\mathbf{y}^t|\mathbf{x}^t)$, and the first term in the gradient is the contribution of F_θ under the true label, whereas the second term is the expected contribution of F_θ under the current model. The well known L-BFGS algorithm [128], is used for the gradient calculation, and the convergence is assumed to be reached within 300 iterations.

5.3.2 Class Label Prediction

In the proposed CRFs based classification system, a sequence of inputs is recognized by first labeling each element in the sequence through calculating the corresponding optimal Viterbi path under the considered CRFs model. Then, the most frequently occurring class label along the sequence of predicted labels is chosen as an intermediate prediction of the inputs sequence. As discussed in the beginning of this section, the proposed CRFs classifier consists of two sub-classifiers, the first is dedicated to predict labels for f_a features and the other is to predict labels for f_c features. Therefore, we proposed that the ultimate recognition decision is jointly decided by the respective results of the two sub-classifiers. Moreover, and since the number of

the labels to be predicted classes is relatively high (See FIG.5.4), simply picking the most frequent label as the predicted label of the considered sequence, is not enough for a reliable recognition. Thus, we propose a threshold of minimum-occurrence ($\varepsilon_{\bar{L}}$) of a label \bar{L} , to be chosen as the unique global label of the entire sequence. Empirically, the best recognition result is achieved at an average of $\varepsilon_{\bar{L}} \approx 40\%$, hence $\varepsilon_{\bar{L}} = 40\%$ is chosen as the minimum-occurrence threshold.

Given a class label \bar{L}_a that occurs k_a times along a sequence of predicted f_a features, and a class label \bar{L}_c that occurs k_c times along a sequence of predicted f_c features. The proposed CRFs system recognizes a segment by combining the intermediate results of the two subsystems (i.e. f_a and f_c based), as follows:

- (i) If $k_a \geq \varepsilon_{\bar{L}}$, and $k_c \geq \varepsilon_{\bar{L}}$, and $\bar{L}_a = \bar{L}_c$, i.e., both refer to the same class label in f_a as well as in f_c based models, the label \bar{L}_a (or \bar{L}_c) will be assigned to the stroke assuming complete confidence.
- (ii) If $k_a \geq \varepsilon_{\bar{L}}$, and $k_c \geq \varepsilon_{\bar{L}}$, but $\bar{L}_a \neq \bar{L}_c$, i.e., point to different class labels, then the label corresponding to the higher number of occurrences will be assigned to the stroke, and a possibility of substitution error will be reported (i.e., $S_{error} = S_{error} + 1$).
- (iii) If $k_a < \varepsilon_{\bar{L}}$, or $k_c < \varepsilon_{\bar{L}}$, then the label corresponding to the higher number of occurrences will be assigned to the stroke, and a possibility of an insertion error will be reported (i.e., $I_{error} = I_{error} + 1$).
- (iv) If $k_a < \varepsilon_{\bar{L}}$, and $k_c < \varepsilon_{\bar{L}}$, then the stroke will be rejected and a deletion error will be reported (i.e., $D_{error} = D_{error} + 1$).

Furthermore, the recognition of a handwritten word is considered equivalent to the process of recognizing each segment in the given word, where the attached sub values indicating the error possibilities are combined to form a word based confidence score (*conf.*). The confidence score is computed similar to that of HMMs approach stated in Eq.5.7, and predicted labels are mapped to their corresponding Unicode Arabic letters.

5.4 HCRFs for Arabic Handwriting Recognition

It is commonly agreed that models with hidden-state structure usually outperform fully observed ones, since they are more capable of capturing the relevant hidden structure in the given domain [72,77]. To the best of our knowledge, most published approaches for the recognition of offline Arabic handwriting that involve hidden states are using the HMMs. Hence, they inherit the limitations of generative models, as well as, adheres to the Markov's independence assumption among observations. To explore the performance of the probabilistic discriminative models with hidden states on the field of offline Arabic handwriting recognition, we introduce the most recently proposed hidden-state conditional random fields (HCRFs) model to the field [129]. HCRFs is simply an extension of the discriminative fully observed CRFs model, where HCRFs model is equipped with an intermediate set of hidden variables $\mathbf{h} = \{h_1, h_2, \dots, h_n\}$ (between the observations and labels), globally conditioned on the observation vector \mathbf{x} . The hidden variables or the hidden states in our case are devoted to capture assumed hidden patterns of observation values within the observation sequences, which may represent specific shape peculiarities along the segment stroke(s).

To optimize the number of hidden states for each HCRFs model, an approach similar to that employed for the HMMs is adopted, where the number of hidden states are chosen to be proportional to the letter shape complexity⁵. Accordingly, number of the optimized hidden states were found to be 5, 8, 10, and 10, for taxonomy 1 to 4, respectively. Analogous to the formulation of CRFs, HCRFs models, the conditional probability of a class label is given as sequence of observations, as follows:

$$p_{\theta}(\mathbf{y}|\mathbf{x}) = p_{\theta}(\mathbf{y}, \mathbf{h}|\mathbf{x}) = \frac{\sum_{\mathbf{h}} \exp\left(\sum_{i=1}^n F_{\theta}(y_{i-1}, y_i, \mathbf{h}, \mathbf{x}, i)\right)}{\sum_{\mathbf{y}, \mathbf{h}} \exp\left(\sum_{i=1}^n F_{\theta}(y_{i-1}, y_i, \mathbf{h}, \mathbf{x}, i)\right)}, \quad (5.12)$$

where the denominator is a normalization factor similar to $Z_{\theta}(\mathbf{x})$ in Eq.5.9, and the potential function $F_{\theta}(y_{i-1}, y_i, \mathbf{h}, \mathbf{x}, i)$ computes the similarity between a class label, a sequence of observations, and a configuration of the hidden states.

As in CRFs case, to estimate the HCRFs optimal weights θ for a given taxonomy,

⁵Letters of multiple segments or contain loop(s) usually require more states compared to one segment letter.

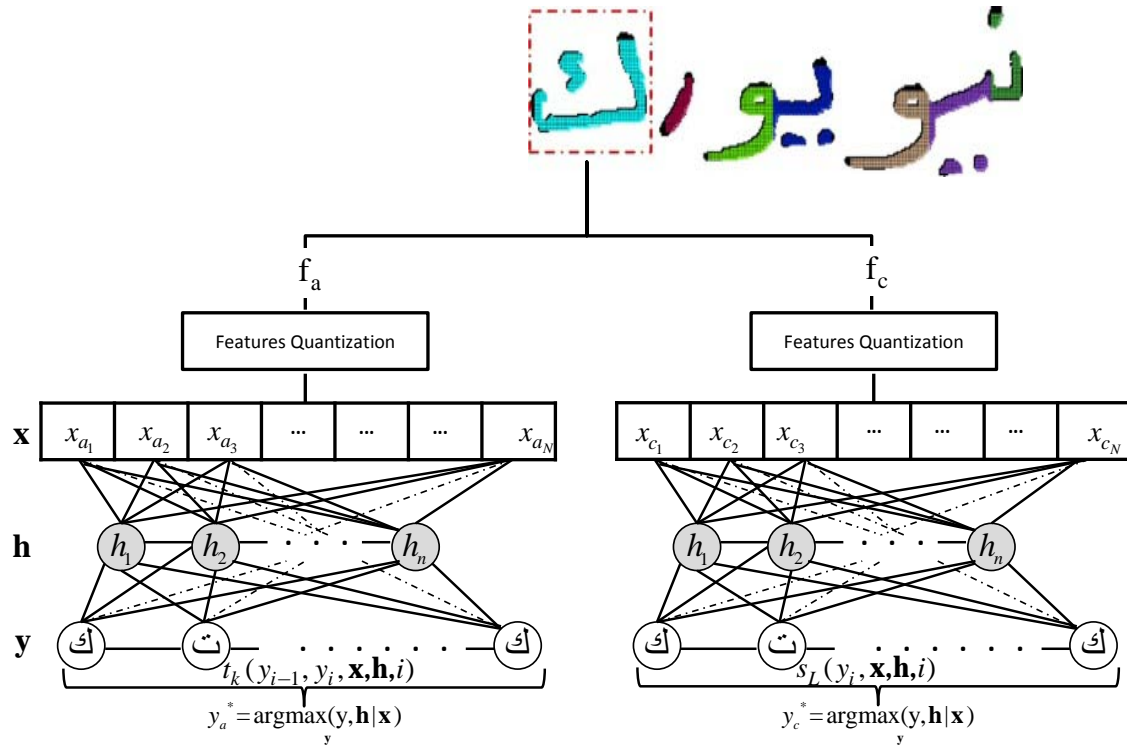


FIG. 5.10. A simplified overview of the proposed hidden-state CRFs (HCRFs.) based recognition approach.

we calculate the gradient ascent of its training samples set $D = \{(\mathbf{x}^t, \mathbf{y}^t)\}_{t=1}^T$ using the L-BFGS algorithm [128], while assuming the corresponding number of hidden states in the taxonomy observation sequences used for training. Furthermore, the convergence is also assumed to be reached within 300 iterations. As a result of the training, seven different HCRFs models are built for each taxonomy, each with different window-size value i.e. (i.e., $\omega = 0, \omega = 1, \dots, \omega = 6$). Moreover, the prediction of the elements of the two different observation sequences and the estimation of the final recognition results are computed just like in the CRFs case. In the next chapter, HCRFs recognition results and performance comparison with CRFs will be presented.

5.5 Conclusion

Based on their theoretical foundations, probabilistic based classification approaches can be categorized into two different categories, namely, generative and discriminative. The objective of this chapter was to introduce probabilistic based classification

approaches capable of recognizing sequences of features extracted from handwritten Arabic words. Instead of using the common sliding window based features preferred when local details are priority, the chapter started by describing new sequential shape description features, that are relatively cost-effective yet capture more discriminative shape characteristics. Then, due to the fact that recognition performance is inversely proportional to the number of classes, letters are grouped into four basic shape categories using some primitive shape characteristics, e.g. number of segments and existence of a loop(s).

As first alternative, the chapter introduced a generative recognition approach using HMMs, in which, an optimized carefully initialized model for each letter in each shape variation is constructed using the EM algorithm. Moreover, an adaptive HMMs based threshold models are also built in order to enhance results by rejecting out-of-vocabulary and meaningless pattern. To further enhance performance, definitive recognition decisions are reached by combining the results of two different types of models and their corresponding threshold models. In order to explore the discriminative recognition alternative, the second part of the chapter was mainly devoted to introduce two relatively recently proposed discriminative based classifiers, namely, the liner-chain CRFs and its extension the hidden-state CRFs (HCRFs). To allow compatible comparisons to the generative approach, both CRFs and HCRFs systems are built on top of the segmentation module used in the HMMs system. And all three approaches, are using the same features and are following the same taxonomization process. Furthermore, the numbers of hidden states for HCRFs models are optimized like in the HMMs case. In the next chapter, the performances of the three classifiers will be evaluated and obtained results will be compared.

Experiment and Recognition Results

THIS chapter is fully dedicated to present and discuss the obtained results of experiments conducted to examine the effectiveness of our different approaches proposed for the problems of text line segmentation, resolving overlapped sub-words, handwritten words segmentation, and the handwriting recognition. To segment a text line, the most probable border-line between every two consecutive text lines is identified as a solution. Samples from two different databases containing pages of modern handwriting as well as historical manuscripts are used for parameters optimization and system performance evaluation. For Arabic alphabet-based scripts, resolving the overlapping of sub-words within a given word is proved to enhance the performance of the subsequent segmentation process. The proposed approach is evaluated on samples drawn from two different datasets. Moreover, in addition to the achieved results the main error types are also reported. The Performance evaluation of the proposed approach for handwritten word segmentation is given in Section 6.3. Experiments are mainly performed on samples from our IESK-arDB database. Additionally to confirm the approach reliability, part of experiments are conducted on the more challenging IFN-ENIT database, where the adopted metric for performance assessment is based on the percentage of pixels matching between the groundtruthed inputs and the system outputs. Furthermore, the approach accuracy, precision, and recall errors are presented. The overall performance of the approach is compared with the performance

of other similar approaches [5,9]. The recognition experiments Section starts by illustrating the experiments setup and the used datasets. For a fair comparison between the proposed recognition approaches, namely, HMMs approach, CRFs approach, and HCRFs approach, all approaches are trained and tested on the same datasets. Besides the achieved recognition results on both the letters level and on the words level, the deletion, substitution, and insertion errors are also reported and discussed. Furthermore, an overall confidence value is computed and attached to the result of the word recognition, indicating the reliability of the obtained results. This chapter concludes with a performance comparison of the three approaches, along with a discussion of the strengths and weakness of each approach.

6.1 Evaluation of Text Lines Segmentation Method

As previously explained in Section 4.1.3, the problem of text line extraction is formulated as a problem of finding the separating line between every two consecutive text lines in a document image. To assess the effectiveness of the proposed approach, samples taken from our IESK-arDB [8] and AHDB [12] databases are used to tune the involved parameters and evaluate the system performance. To the best of our knowledge, these two databases are the only freely available databases that contain pages of handwritten Arabic text, where IESK-arDB contains two different types of handwritten documents (i.e., gray scale modern Arabic handwritten pages of text that contains 13 to more than 18 text lines in each page, and colored pages of handwritten Arabic manuscripts that have from 10 to 30 text lines.). In addition to the common challenges associated to the modern handwriting, historical manuscript handwriting is usually characterized by non-uniform text line skew, letter size variations even within the same text line, and a relatively very high number of overlapping and touching components, which demand a corresponding high tolerance value (i.e., η see Section 4.1.3.2) in order achieve satisfactory results. FIG.6.1 (a) and (d) show a sample of a manuscript page (from our IESK-arDB) and the corresponding result of text line segmentation, where different colors indicate different text lines. FIG.6.1 (b) is an IESK-arDB sample for modern Arabic handwriting and FIG.6.1 (e) is highlighting the detected text lines.

Compared to IESK-arDB, the sample pages of AHDB database are well written

with a wide spaced and relatively fewer words text lines, further it contains very few cases of touching or overlapping components. This fact might explain why the performance of the proposed approach has nearly reached optimal detection rates using samples drawn from AHDB database. FIG. 6.1 (c), presents a monochromatic sample taken from AHDB database and FIG. 6.1 (f) shows the corresponding detected text lines. Given the various degrees of layout complexities, the first type

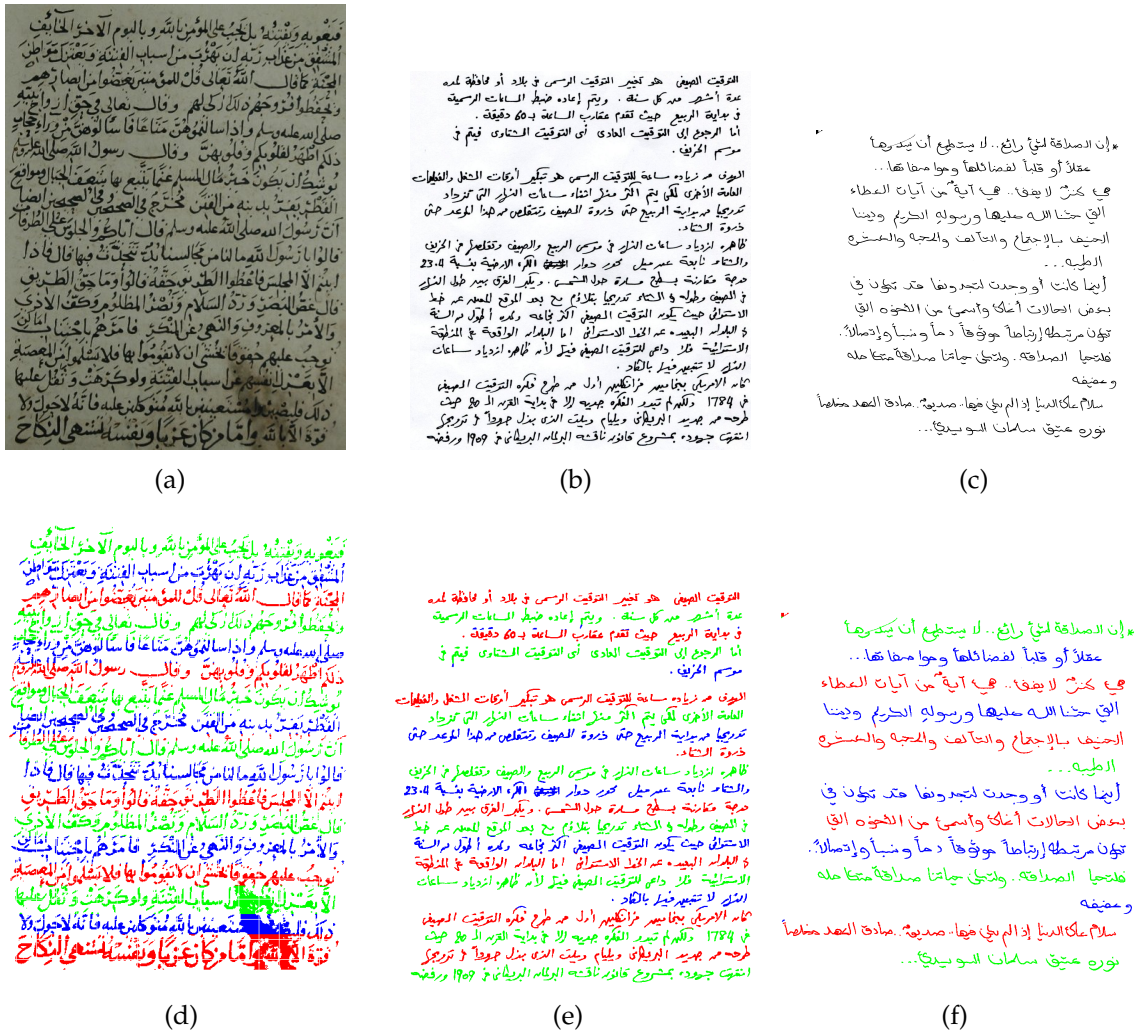


FIG. 6.1. Sample of the images used in experiments: (a) A page from the IESK-arDB manuscript collection, (b) a page of a modern handwriting from the IESK-arDB database, (c) a sample page from the AHDB database. (d), (e), and (f) are the corresponding results of our text line segmentation approach.

of experiments is conducted to optimize the tolerance parameter (i.e. η) and the step-width parameter (i.e., Υ) (see Alg. 4.1), separately, for IESK-arDB manuscript collection, IESK-arDB modern handwriting, and AHDB handwritten page collection. For this purpose, from our IESK-arDB we have used 20 pages of manuscript

collection containing around 500 text lines, 10 pages of modern handwriting with about 200 text lines, and from AHDB database, we have used 15 pages with more than 200 text lines. In contrast to previous parameters tuning experiments, we used a larger number of samples for performance evaluation experiments, where the used data are composed of 50 pages of IESK-arDB manuscript collection containing about 1300 text lines, 20 pages of IESK-arDB modern handwriting with 350 text lines, and from AHDB 30 pages with nearly 400 text lines. All samples are manually annotated by superimpose imaginary lines to separate consecutive text lines. As a performance metric, we adopted an approach based on counting the number of matches between the detected text lines and the ground-truthed text lines [130,131]. According to this approach, a table of match-scores is created where the table entries are computed based on the result of intersecting "ON" pixels of detected text lines with the ground-truth pixels. As stated in [132], assume I is a set of "ON" pixels in a document image, G_i a set of "ON" pixels of a ground-truthed text line i , and R_j is set of the "ON" pixels of a detected text line j . Then the match score between a ground-truth i and a detected text line j is calculated, as follows

$$Match(i, j) = \frac{C(G_i \cap R_j \cap I)}{C((G_j \cup R_j) \cap I)}, \quad (6.1)$$

where C is a simple function counting the number of the corresponding "ON" pixels. Furthermore, in our experiments, a perfect (i.e., one-to-one match) is achieved only if the match-score value exceeded a user predefined threshold t . Moreover, let M be the number of ground truth text lines, N the number of detected text lines, and K the number of perfect matches. A performance metric Γ is defined, as follows

$$\Gamma = \frac{2 \times preRate \times recRate}{preRate + recRate}, \quad (6.2)$$

where $preRate = \frac{K}{M}$ is the precision rate, and $recRate = \frac{K}{N}$ is the recall rate.

Optimization experiments begins by manually discretizing Υ and η (i.e., $\eta = \{0.1, 0.2, 0.3, \dots, 1.0\}$ and $\Upsilon = \{20, 30, 50, \dots, 200\}$). Then, a grid search based method is used to find the optimal values using the optimization dataset, where the matching threshold t is set to 0.95 in both optimization and testing experiments. As a result of optimizing the parameters pair (η, Υ) , it has been found that the values $(\eta = 0.05, \Upsilon = 100)$, $(\eta = 1.0, \Upsilon = 45)$, and $(\eta = 0.1, \Upsilon = 150)$, are delivering the best performance on IESK-arDB modern handwriting, IESK-arDB manuscript collection,

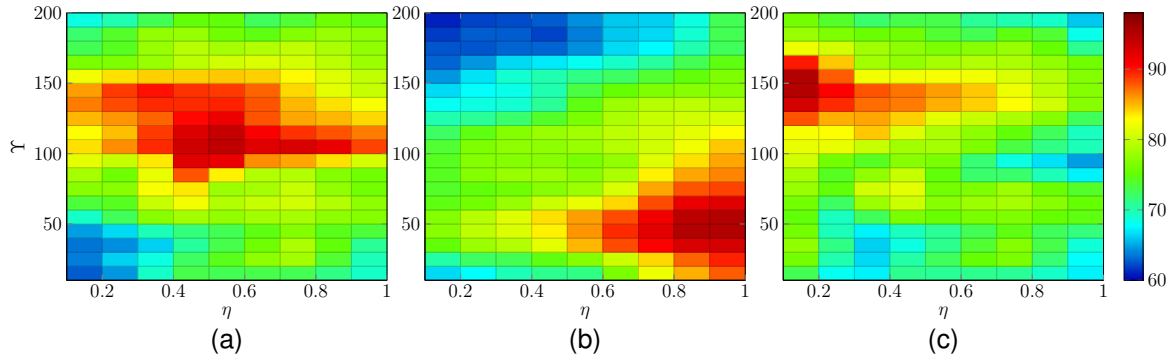


FIG. 6.2. Optimization experiments for η and Υ parameters : (a) Performance rates on IESK-arDB modern handwriting collection, (b) performance rates on IESK-arDB manuscript collection, and (c) performance rates on AHDB handwritten page collection.

Table 6.1. The results achieved using IESK-arDB modern handwriting (HW.) test data, IESK-arDB manuscripts (Manu.) test data, and on AHDB test data, where *GT. lines* are the groundtruthed text line and *Det. lines* are the detected text lines.

Test sets	# <i>GT. lines</i>	# <i>Det. lines</i>	# <i>perfect matches</i>	<i>preRate</i> %	<i>recRate</i> %	$\Gamma\%$
IESK-arDB HW.	350	315	298	85.14	94.60	89.62
IESK-arDB Manu.	1300	1112	988	76.00	88.84	81.92
AHDB	400	381	364	93.25	97.90	95.51

and AHDB collection, respectively. FIG. 6.2 summarizes the obtained experimental results. After obtaining the optimized values, evaluation experiments are carried out, separately, on the test set of each type of handwriting. Table 6.1, summarizes the obtained results on the set of text lines for each handwriting type. Due to the high-quality handwriting images of AHDB, the proposed method achieved a significant high rate of 95.51% using samples from AHDB database. These results compare very favorably with the best results reported in the literature [131, 132].

Even though, the test data of IESK-arDB modern handwriting is freely written text with many cases of letter ascender-descender overlapping/touching, the achieved result of 89.62% are found to be very satisfactory and comparable to results obtained using more complex and expensive methods like in [133]. Due to the fact that historical manuscripts, typically, involve more challenges compared to the ordinary handwriting, we believe that the achieved rate of 81.92% is very promising and impressively confirm the efficiency of the proposed method. Appendix B.1

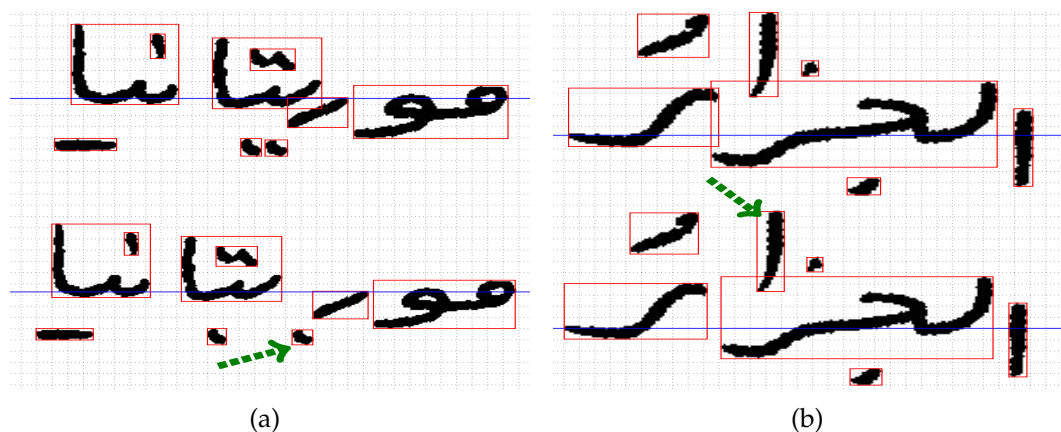


FIG. 6.3. Types of observed errors: (a) Example of Type I error, (b) example of Type II error. The green arrow indicates the affected stroke.

shows successful text line segmentation results on manuscript pages with complex layouts.

6.2 Evaluation of Sub-words Overlapping Resolver

As discussed in Chapter 4, resolving sub-words overlapping is an important prerequisite before conducting word segmentation. To assess the performance of our method proposed in Section 4.2.2.1, samples from the two well-known databases, i.e., IESK-arDB and IFN-ENIT are used. From each database, we choose 100 word images with an average of 2.5 sub-words and a total of 658 sub-words. In the experiments, two different types of errors are observed, the first (Type I) is generated when an auxiliary component is incorrectly assigned to a neighboring main component, that is shifted away from its main component. This type of errors is typical to dense handwritten words, where the centroid of one or more auxiliary components tends to be nearer to the centroid of a neighboring main component than to its own main component. FIG. 6.3 (a) shows an example of such error. Type II error typically occurs when a sub-word is written far from the word baseline, and as a consequence, it is erroneously considered as an auxiliary component rather than a main component. Usually, type II error happens when the letter "ل" (Ailf) (in the isolated form) appears in the middle of a word and written as a short stroke. FIG. 6.3 (b), shows an example of type II error where the letter "ل" is written as a curved sloping stroke above the word baseline.

FIG.6.4 summarizes the performance results obtained on the two data sets taken from IESK-arDB and IFN-ENIT databases, where the proposed method has been executed for each dataset separately. The fact that the IESK-arDB samples are relatively well-written compared to the samples of IFN-ENIT might explain the better performance obtained using IESK-arDB samples. Moreover, it is also

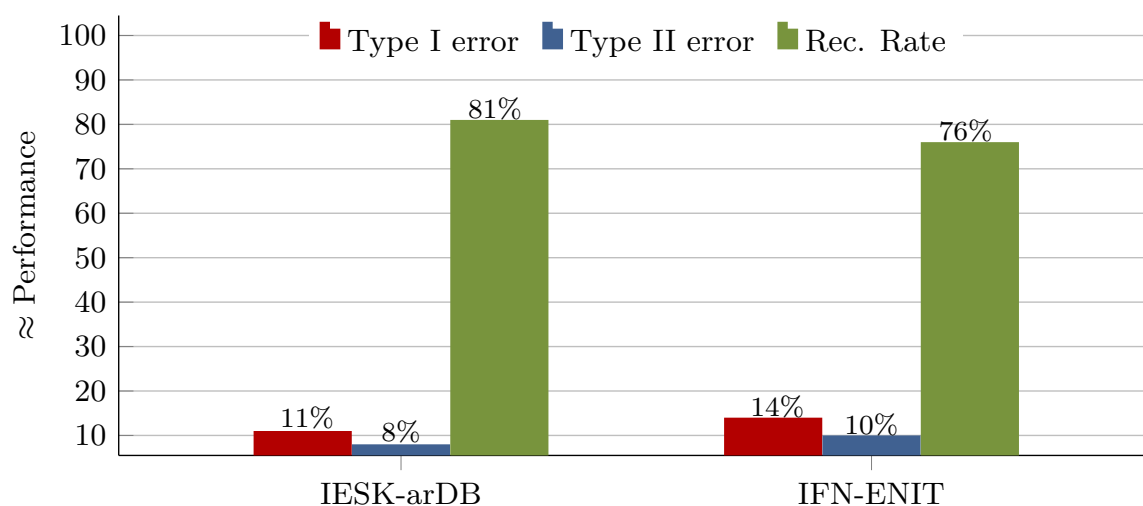


FIG. 6.4. Evaluation results of resolving of sub-word's overlapping: Experiments conducted on two different data sets from IESK-arDB and IFN-ENIT databases.

important to mention that the proposed method is not only limited to one-word images, but rather, it is capable of processing images of multiple words or even images of sentences. FIG. 6.5, shows samples of overlapped sub-words in single words and in sentences and the corresponding overlaps free versions. More samples are presented in Appendix B.3

6.3 Evaluation of the Handwritten Word Segmentation Approach

Since it is the only database that is annotated with segmentation information, we mainly used the IESK-arDB database to assess performance of our handwritten word segmentation approach proposed in Section 4.2.2.2. A total of 600 word images written by 10 different writers are used in the assessment process. To ensure the evaluation reliability, 100 samples from set_a of IFN-ENIT database are selected.

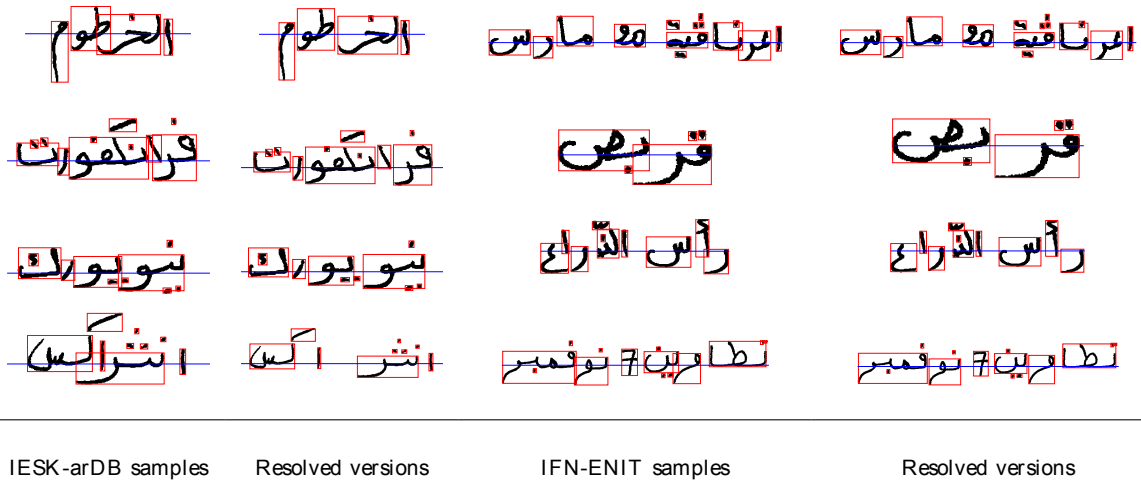


FIG. 6.5. Examples for successfully resolved overlapping, where word images taken from IESK-arDB and IFN-ENIT databases, respectively.

Then, each sample is manually annotated with segmentation information (i.e., letters border coordinates) to test our method performance on them. By examining the published literature and in order to compare with other methods, one notices that very few methods report segmentation results separately from recognition. From these methods, Xiu et al. [5], is chosen for two reasons. Firstly, it is well described allowing us to implement it. Secondly, it is exclusively addressing the same issue. A similar metric to that is used for text line segmentation experiments in Section 6.1, is adopted as a letter segmentation metric. It is based on the percentage of pixels matching between the annotated letter image and a segmentation result. Unlike in text line segmentation, in this case G_i variable denotes a set of all "ON" pixels of an annotated letter within a handwritten word. R_j represents the set of "ON" pixels of a segmentation result. M is the number of letters in the ground-truth word image. N is the number of detected letter representative in the segmented image, and K is the number of perfect matches, where the threshold parameter t is set to 0.85¹.

The match-scores table is created for each pair of annotated handwritten word and its corresponding segmented version, where rows stand for letters in the

¹Letter borders are first detected in the skeletonized version then superimposed on the source, to alleviate the effect of the different pixel numbers between the annotated and the result images, t is optimized to the smallest reasonable tolerance value.

Table 6.2. Letter-based performance comparison of our proposed segmentation approach and the approach proposed in Xiu et al. [5].

Used Approach	Used Database	#GT. words	#GT. letters	#Seg. element	#Perf. match	preRate %	recRate %	Γ %
Our	IESK-arDB	600	2615	2582	1915	73.23	74.16	73.69
	IFN-ENIT	100	456	427	312	68.42	73.06	70.67
Xiu et al.	IESK-arDB	600	2615	2692	1708	65.31	63.44	64.36
	IFN-ENIT	100	456	482	292	64.04	60.58	62.26

former, and columns represent letter candidates in the later and table entries are the matching scores. Moreover, precision rate (*preRate*), recall rate (*recRate*), and the performance metric Γ are all computed as in the text lines segmentation case in Section 6.1.

Table 6.2 details the validation results of the proposed segmentation approach on the letters level across IESK-arDB and IFN-ENIT databases. Further, it presents the results of Xiu et al. [5] on the same databases, which confirm the superiority of our approach. Notice that *preRate* and *recRate* are first computed on the word level, then all outcomes are averaged. Furthermore, it is relevant to notice that the performance is boosted on IESK-arDB compared to IFN-ENIT, due to the fact that the samples from the former are relatively short and well written. Performance comparison of the proposed segmentation method using two different databases is presented in FIG. 6.6. For IESK-arDB and IFN-ENIT, correct segmentation or complete success is reported in 71% and 64% of cases, respectively, where correct segmentation means that the proposed approach accurately discovers the necessary pixels required to recognize the letter within the considered handwritten word (i.e., 85% pixels intersection).

On the other hand, partial success or segmentation with errors is encountered in 29% and 36% of words for IESK-arDB and IFN-ENIT databases, respectively. In general, a 46.15% of partial success/failure can be referred to the so called over segmentation problem (Over Seg. in FIG. 6.6), which occurs when branch or/and end points appears more than once inside the letter body. As a consequence, the letter main body will be splitted into two or more parts. We noticed that this problem is specific to letters *س* and *ش* (SIEN and SHIEN) when they are written in

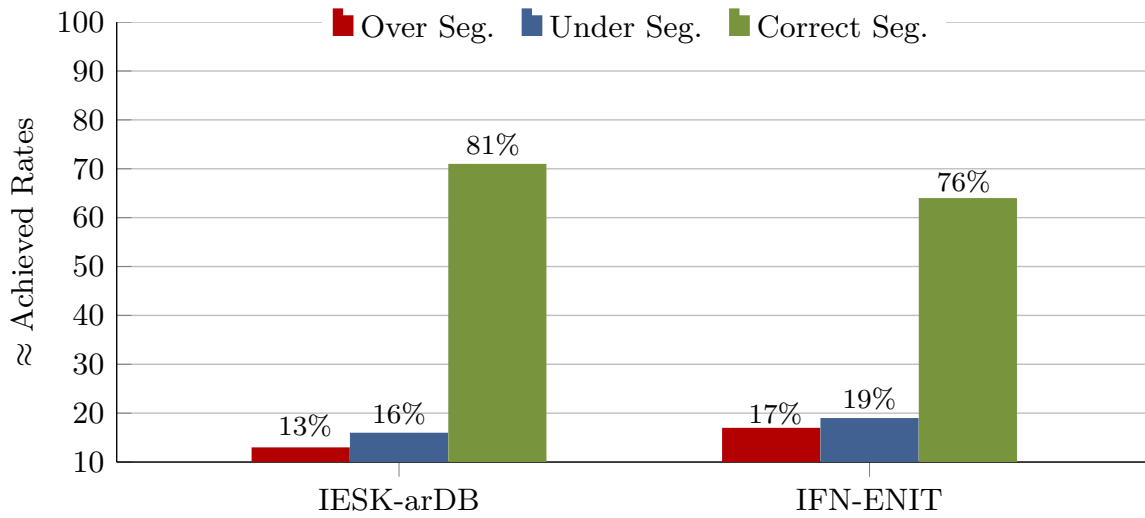


FIG. 6.6. Word based segmentation performance of our proposed approach on IESK-arDB and IFN-ENIT: Over Seg. and Under Seg. stand for the percentage (rounded) of words containing over segmentation and under segmentation errors, respectively. Correct Seg. is the percentage of correctly segmented words.

specific fonts like "Naskh" and "Thulth".

The other 53.84% of partial success cases appear when Critical Feature Points (CFPs), cease to exist between two consecutive letters leading to considering them as a representative of one letter. This problem is so called under segmentation (Under Seg. in FIG. 6.6) and it is specific to cases where the second letter to the left is being a connected \lrcorner (ALF), or a connected \lrcorner (LAM) with sheared distortion angle to the left. Also, it appears occasionally when the letter $\kern-0.25ex\lrcorner$ (KAF) occurs in the middle of two connected letters (e.g., تکب), and the letter KAF's upper part vertically overlapping the previous character on the right. FIG. 6.6 (a) and (b), illustrate the two different types of errors, respectively, where in FIG. 6.6 (a) the letter ش is splitted into two parts as a consequence of a CFPs existence inside the letter main body. And in Fig (b), the letters \lrcorner and $\kern-0.25ex\lrcorner$ are merged into one letter as a result of CFPs absence between them.

We believe that these problems can be addressed, either by expanding the CFPs set to contain more features points (e.g., Local minima points) and then accordingly modify and add heuristic rules, alternatively, they can be solved in subsequent recognition phases (i.e., post-processing phase) where the recognition results can be adjusted against lexicons using text retrieval techniques. Given the



FIG. 6.7. Over and Under segmentation errors: (a) Over segmentation problem where letter ش is erroneously segmented into two parts, (b) under segmentation problem with two letters ل and ك merged into one.

inherent challenge of the segmentation of Arabic handwriting and despite all the aforementioned problems, we believe that the results achieved using the proposed approach compares very favorably with best results reported in the literature. Experiments confirm the validity of our approach and its applicability for single words as well as for short sentences. FIG. 6.8, supports such claim by presenting several examples taken from IESK-arDB and IFN-ENIT database. Appendix B.2, shows more results samples of correct, over-, and under- segmentation.

6.4 Recognition Experiments

Since handwritten words samples of IESK-arDB database are all annotated with segmentation information, we used this database to train the HMMs, CRFs, and HCRFs classifiers. A total of 800 handwritten words containing about 3500 letters are used to build letters models and taxonomies models (in case of CRFs and HCRFs). For testing, the evaluation of the proposed approaches are carried out on two databases. In the evaluation experiments, we used not only 400 words images containing more than 1700 letters from IESK-arDB database, but also a 200 (manually segmentation ground-truthed) word images with about 1000 letters from IFN-ENIT database. The proposed systems are implemented in Matlab and C++, where the Matlab HMMs toolbox and the HCRFs library [14] are used to build HMMs, CRFs, and HCRFs classifiers. All experiments are performed on a Windows 7 professional and a Matlab R2013a installed on an Intel(R) Xenon(R) CPU server machine with 2.67 GHz and 64.0 GB of memory. The recognition performance of the proposed systems are sufficiently evaluated with respect to both letters and words.



IESK-arDB samples Segmented versions IFN-ENIT samples Segmented versions

FIG. 6.8. Examples for successful word segmentation.

6.4.1 Evaluation of HMMs Recognition Performance

Strictly speaking, based on our HMMs recognition approach, proposed first in [120], then refined and documented in Section 5.2.6; a letter image is first assigned to one of four shape-based taxonomies. Then, features are extracted and tested against the corresponding set of HMMs models and the respective threshold model. As a result, a letter might be either recognized with a complete confidence value, recognized with a 50% possibility of a substitution error (S_{error}), recognized with a 50% possibility of Insertion error (I_{error}), or completely rejected and hence a deletion error will be reported. Tables C.1 and C.2 summarize the achieved recognition rates using the three proposed approaches (i.e. HMMs, CRFs, and HCRFs) on IESK-arDB and IFN-ENIT databases, respectively. The results are reported for each letter in each written form, where "B", "E", "I", "M" stand for Begin, End, Isolated, and Middle forms. Using the HMMs approach, and because of their relative distinct shapes and

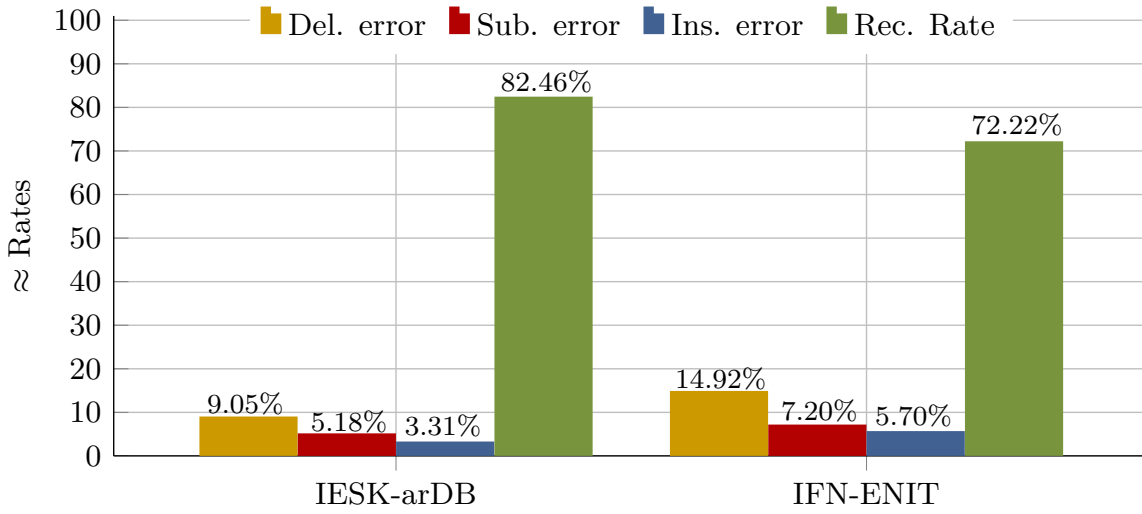


FIG. 6.9. Letter based recognition performance of HMMs: Recognition rates of letters samples from IESK-arDB and IFN-ENIT databases with the amounts of different classification of errors.

the absence of additional dots and diacritics, letters such as (د ر و ه) achieved high recognition rates, compared to letters consisting of multiple strokes (e.g., ت ث ذ ز). This is mainly caused by shifting and/or fusion of dots and diacritics, which significantly increase shape similarity and weaken the features discriminative power. Further, it is observed that the letter position (i.e. the written form) inside a word affects the recognition performance, where, for example, letters in isolation form achieve the highest recognition rate (i.e. 85.39%), while letters written in the middle form achieve the lowest rate (i.e. 79%). FIG. 6.9, presents the overall letter-based performance of the proposed HMMs recognition approach on IESK-arDB and IFN-ENIT evolution sets, and further explains the amount and the type of errors occurring during the classification process. Basically, three different types of errors have been observed. The first is the deletion error, which occurs when a letter is not recognized, because its maximum likelihood is less than that of the threshold model. The second type of errors is so called substitution error, which is generated when a letter is confused with another one. The last error is the insertion error, which is the result of recognizing a non-letter segment (resulting from inaccurate segmentation) as a letter. On the IESK-arDB letter evaluation set (as depicted in FIG. 6.9, an average recognition rate of 82.28% is reached, whereas a 9.24%, 5.18%, and a 3.30% are reported as deletion, substitution, and insertion error rates, respectively. In comparison to results achieved using IESK-arDB, the average results obtained on

Table 6.3. Examples of the inputs and the results of the HMMs based recognition approach. Notice in the third row the third segment (from the right) is rejected hence it is replaced with #. Also the first segment (from the right) in the fourth row is recognized as "ر" with 50% possibility of substitution error.

Inputs	Results	
	Recognized letter	Confidence value (<i>conf.</i>)
مثالين	م ث ل ي ن	$1 - (0.5 \times 0 + 0.5 \times 0 + 0)/5 = 1.00$
الليلة	ال ل ي ل ة	$1 - (0.5 \times 0 + 0.5 \times 0 + 0)/6 = 1.00$
فيلد ط	ق ب # د ط	$1 - (0.5 \times 1 + 0.5 \times 1 + 1)/5 = 0.60$
انتحاري	ر ن ت ح ا ر ي	$1 - (0.5 \times 1 + 0.5 \times 0 + 0)/7 = 0.93$

the IFN-ENIT evaluation set are relatively weaker (i.e., 72.22%, 14.92%, 7.20.28%, and 5.70%, for recognition rates, deletion errors, substitution errors, and insertion errors, respectively), which can be attributed to the fact that letter models are built using only samples from the IESK-arDB. Instead of rejecting or recognizing a handwritten word as whole, in the proposed approach, the word recognition problem is reformulated into tractable sub-problems of rejection or recognition of the letters constituting the considered word. Furthermore, according to Section 5.2.6, a confidence value ($conf \in [0, \dots, 1]$) will be attached to the recognition results to describe how reliable the result is, where $conf = 0$ means a non-recognizable word, while $conf = 1$ implies a complete confidence in the recognition result. The confidence value $conf$ is computed as a function of deletion, substitution and insertion errors (as in Eq. 5.7). Besides being helpful in the assessment of the reliability of the recognition results, attached confidence values can also be used to initiate post-processing procedures (e.g., as spell and grammar correction) that may further improve the obtained recognition results.

In short, the proposed system recognizes a sequence of strokes by mapping each stroke to an Unicode Arabic letter, and as a result generating a sequence of digital letters with an overall confidence value. If, however, a stroke is rejected as being a

meaningless pattern, it will be replaced by # symbol in the result. Table 6.3 gives some examples of handwritten word images along with the Unicode letters and the corresponding recognition confidence values which are computed based on Eq. 5.7. FIG. 6.10, shows the achieved results when the proposed HMMs approach is tested on the evaluation sets of IESK-arDB and IFN-ENIT databases. For simplicity purposes, results are projected into an axis of five different pins of confidence values, each result is sorted by rounding its confidence value to the nearest pin value.

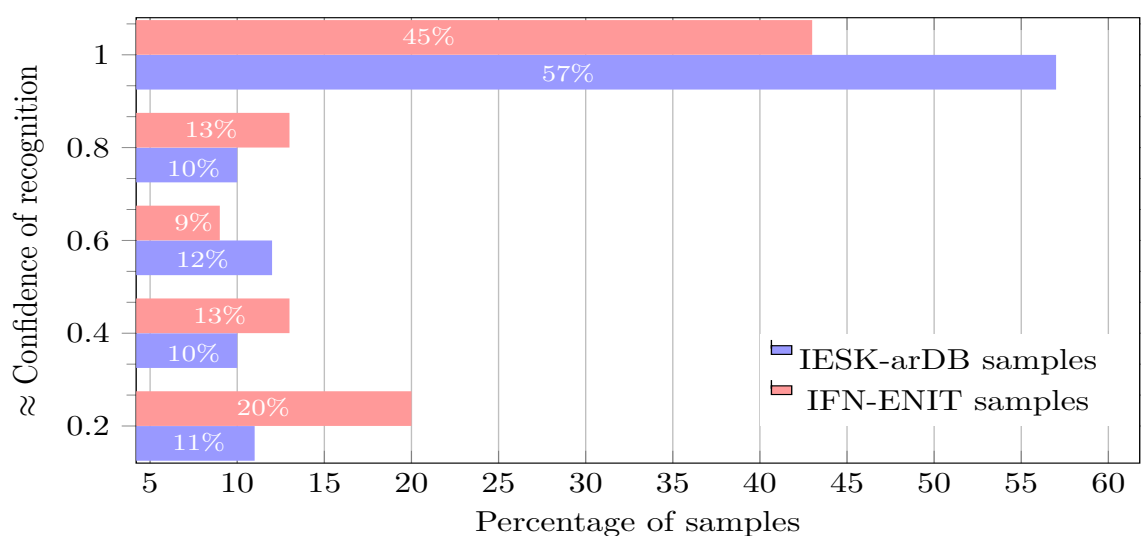


FIG. 6.10. Word based recognition performance using HMMs: Percentage of recognized words with the respective recognition reliability values.

On the words level, recognition associated with confidence values higher than 0.5 are 79% and 67% for samples of IESK-arDB and IFN-ENIT databases, respectively. On the other hand, a low confidence of less than 0.3 is reported for 11% of samples of IESK-arDB and for 20% of IFN-ENIT samples. We believe that such results can be further improved by both optimizing the HMMs model parameters and integrating a spelling checker as post-processing stage.

Finally, it is also important to mention that the relatively poor performance on IFN-ENIT samples is related to several facts, such as, (i) the models are only constructed from the samples drawn from IESK-arDB database, (ii) IFN-ENIT samples consist very often of multiple words with excessive elongation that complicating the segmentation process, and (iii) the variability of IFN/ENIT is higher than that of IESK-arDB, as more writers are involved, and diacritics such as SHADA which is not popular in handwriting, are added to the letter main body.

6.4.2 Evaluation of CRFs and HCRFs Recognition Performance

Instead of building a model for each class as in HMMs, CRFs and HCRFs work by building a single model for all to be predicted classes, where each class is represented through a single state within such a model. Accordingly, and in order to keep the number of states manageable, CRFs and HCRFs models are separately created for each taxonomy. In case of CRFs, a letter form is represented as a single state in the model, whereas HCRFs use a taxonomy specific number of optimized states to model the letter (see Section 5.4). CRFs and HCRFs models are trained and tested on the same data sets used by the HMMs approach, and experiments are performed on systems with the same specifications.

Typically, we conducted experiments to fulfill two purposes, the first is to optimize the parameters of the respective recognition system, and the second is to test the system efficiency. To optimize the window-size parameter for each taxonomy, we train seven CRFs and seven HCRFs classifiers for each taxonomy, where every classifier is trained using different window-sizes ($\omega = 0, \omega = 1, \dots, \omega = 6$). Generally, a 28 CRFs and 28 HCRFs classifiers are built (4 taxonomies \times 7 window-sizes), and by individually evaluating the performance of each classifier, only one CRFs and one HCRFs classifier is selected for each taxonomy. FIG. 6.11 compares the performance of CRFs and HCRFs and shows the effect of modeling the dependency range by using different window sizes (ω) on each letter taxonomy. The experimental results indicate that in case of CRFs, only incorporating the direct neighbors (i.e., $\omega = 1$) will positively affect performance, whereas completely ignoring neighboring elements in computation (i.e., $\omega = 0$), or considering faraway elements (i.e., $\omega > 1$) drastically decrease the system performance. When assessing the performance of the CRFs classifiers using the letter based IESK-arDB test set, the best recognition rates were achieved through classifiers built with $\omega = 1$, hence only classifiers built with $\omega = 1$ are selected as CRFs representatives for the rest of experiments.

As for HCRFs, by using the same test set, the performance improves as ω increases, reaching its peak over all taxonomies at $\omega = 3$, then begins to decay beyond $\omega = 4$. Such a tendency implies that incorporating dependencies positively influence performance, especially when the hidden pattern is also reasonably considered.

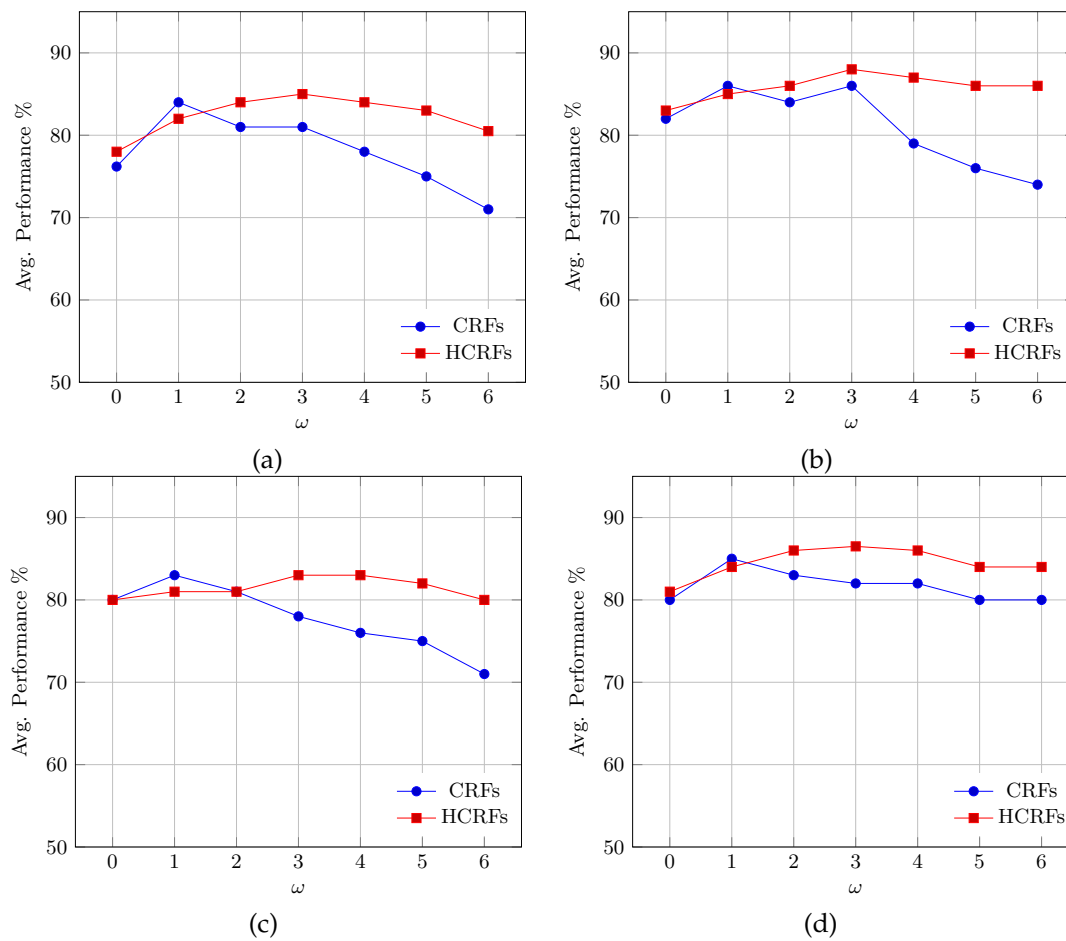


FIG. 6.11. CRFs' and HCRFs' performance with different window sizes (ω) : (a) Performance on Tax.1, (b) performance on Tax.2, (c) performance on Tax.3, and (d) performance on Tax.4

Table 6.4 summarizes the average letter based results achieved using the best-performing CRFs (i.e. with $\omega = 1$) and the best-performing HCRFs (i.e. with $\omega = 3$) on the test dataset. In general, the achieved results confirm the fact that recognition performance is inversely proportional to the number of the classes. Therefore, the lowest recognition rates are reported when testing CRFs and HCRFs classifiers of taxonomy Tax.3, with average rates of 82.10% and 83.64% for both classifiers, respectively. Such modest performance can be explained by the relatively large number and the complex shape characteristics of letters fall under Tax.3. On the contrary to results achieved on Tax.3, CRFs and HCRFs reached their best results of 84.88% and 87.00%, respectively, on the test samples of taxonomy Tax.2, which has the fewest number of class labels. For more insight, FIG. 6.12, details the average recognition rates achieved for each letter's form falling under taxonomy Tax.2 (i.e.,

Table 6.4. Letter based performance of CRFs and HCRs across IESK-arDB and IFN-ENIT database for the four letter taxonomies.

Dataset	CRFs' Rec. rates %				HCRFs' Rec. rates %			
	<i>Tax.1</i>	<i>Tax.2</i>	<i>Tax.3</i>	<i>Tax.4</i>	<i>Tax.1</i>	<i>Tax.2</i>	<i>Tax.3</i>	<i>Tax.4</i>
IESK-arDB	85.83	85.77	83.37	85.24	86.87	87.45	84.73	86.65
IFN-ENIT	83.51	83.99	80.82	82.88	84.93	86.54	82.54	84.76
Avg.%	83.95				85.56			

letters of one stroke with loop), for both IESK-arDB and IFN-ENIT databases. The results achieved using taxonomy Tax.1, Tax.3 and Tax.4, are given in Appendix C.

In general and as expected, letter forms with distinctive shapes such as isolated "ﻮ" and "ﺢ" are recognized efficiently, since it is less likely to be confused with other class labels and also very often perfectly segmented². In FIG. 6.12 (a), as expected CRFs and HCRFs approaches reached their best recognition results (i.e., 92.13% and 93.58%, respectively) on the simple shape of letter "ﻮ". On the other hand, CRFs and HCRFs show their weakest performance on the middle form "ﻮ"³ of letter "ﻊ", where 79.51% and 80.48% are respectively registered for CRFs and HCRFs on samples of IESK-arDB (see FIG. 6.12 (a)), and 77.41% and 79.05% for samples of IFN-ENIT database (FIG. 6.12 (b)). Such tendency can be attributed to the fact that "ﻮ" is usually written with a middle loop that turns it vulnerable to be confused with other letters forms, such as "ﻮ", "ﻮ", and "ﻮ". It is also important to point out that, in average, performance varies according to the considered letter form, where the highest recognition rate of 86.42% and 87.93% are reached on the isolated form for CRFs and HCRFs, respectively. And the lowest rates of 80.79% and 82.48% are obtained on the middle form for both classifiers respectively. This is due to the fact that letters in the isolated form are typically separated by a whitespace from the neighboring letters. This leads to a perfect segmentation, and hence increasing the discriminatively of the extracted features. FIG.6.14 summarizes the obtained results according to the four different handwritten forms, where the HMMs, CRFs, and HCRFs classifiers all reach their best performance on letters in the isolated form.

²Notice that ﺢ is classified under Tax.2, since it is often handwritten with an upper loop.

³The ﻊ middle form ﻮ is always handwritten with a loop.

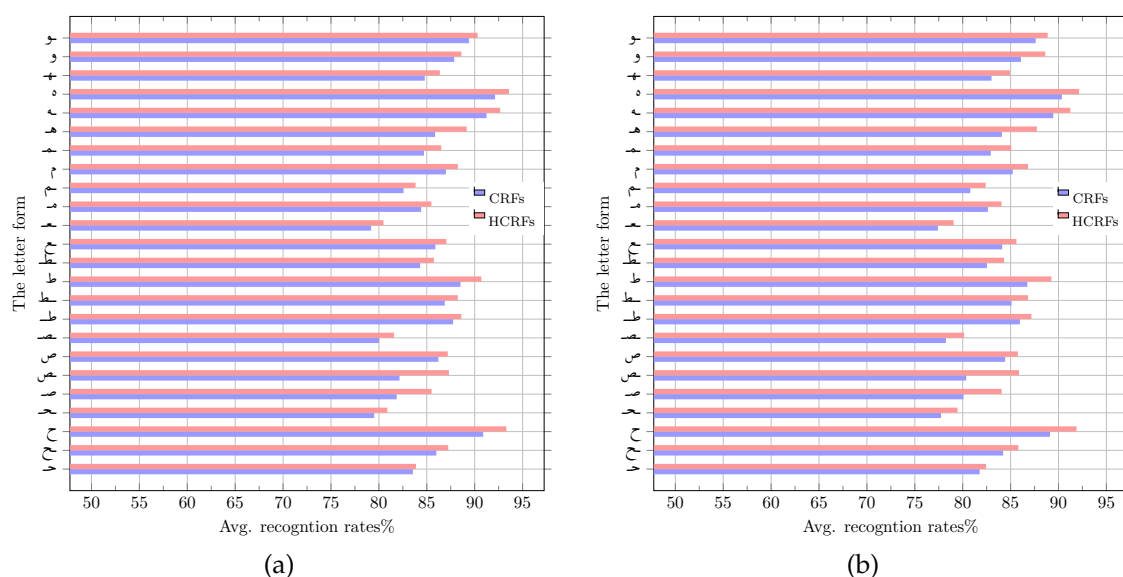


FIG. 6.12. CRFs and HCRFs performance on letter shapes under taxonomy Tax.2: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database.

The evaluation process of CRFs and HCRFs is conducted on the same dataset used in case of HMMs based approach, which, as previously described in Section 6.4, contains 400 words images from the IESK-arDB and 200 words images of the IFN-ENIT. Like IESK-arDB samples, samples of IFN-ENIT are fully ground-truthed and annotated with segmentation information and the equivalent Unicode labels for each segment. As described in Section 5.3.2, and similar to the HMMs approach, word recognition using CRFs and HCRFs is also performed as the result of recognizing each stroke in the considered word. To indicate how confident the recognition system is about the correctness of the final outputs, overall confidence values are calculated according to Eq. 5.7 and returned along with the recognition results. FIG. 6.13 (a) and (b), detail the results achieved using CRFs and HCRFs, respectively.

For simplicity, the obtained results are rounded and projected into five different levels of confidence. When tested on the evaluation set, CRFs system recognized 79% of word images of IESK-arDB and 72% of word images of FIN-ENIT, with confidence values higher than 0.5. On the other hand, 82% and 80% of samples from IESK-arDB and IFN-ENIT, are recognized respectively with confidence values higher than 0.5 using the HCRFs system. Results with confidence values less than 0.5 are considered of low recognition rate (which registered on 21% of IESK-arDB

samples and on 28% of IFN-ENIT samples using CRFs approach, and ,similarly, on 18% and 20% using HCRFs.). Considering the obtained recognition results, we can conclude that the HCRFs recognition approach clearly outperforms the CRFs one, specially on IFN-ENIT samples, as the HCRFs' hidden layer allows the model to be potentially adopted to any unseen handwriting pattern and hence a performance boost of about 8% is observed.

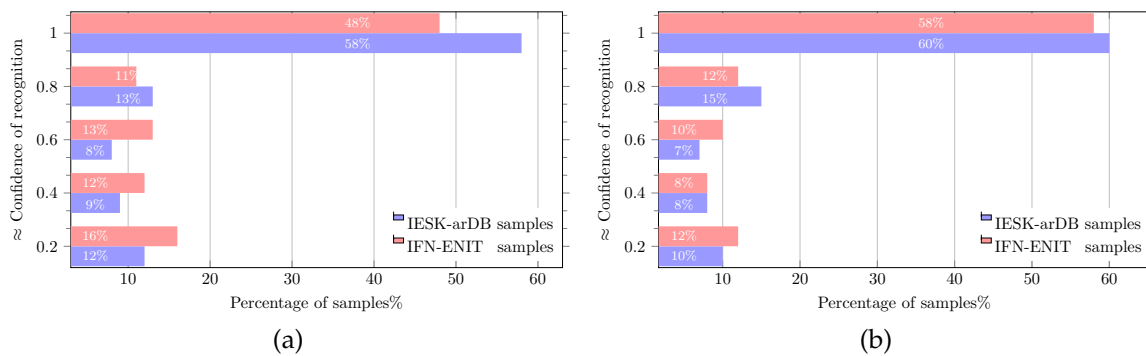


FIG. 6.13. CRFs and HCRFs word based performance on evaluation sets of IESK-arDB and IFN-ENIT databases: (a) CRFs performance, (b) HCRFs performance.

Even though, the HCRFs approach shows strong performance compared to CRFs one, the HCRFs, in general, are very expensive in terms of training costs. FIG.6.14 (b) summarizes the cost in terms of time for the two approaches on all taxonomies, and further indicates that the cost is proportional to the window size ω , the size of the considered taxonomy (i.e. number of different class labels), and to the number of hidden states in case of HCRFs. As an example, the process of training HCRFs model for Tax.3 with window size $\omega = 5$ and 10 hidden states, requires 10 hours in average, while the same process needs about 9 hours for Tax.1.

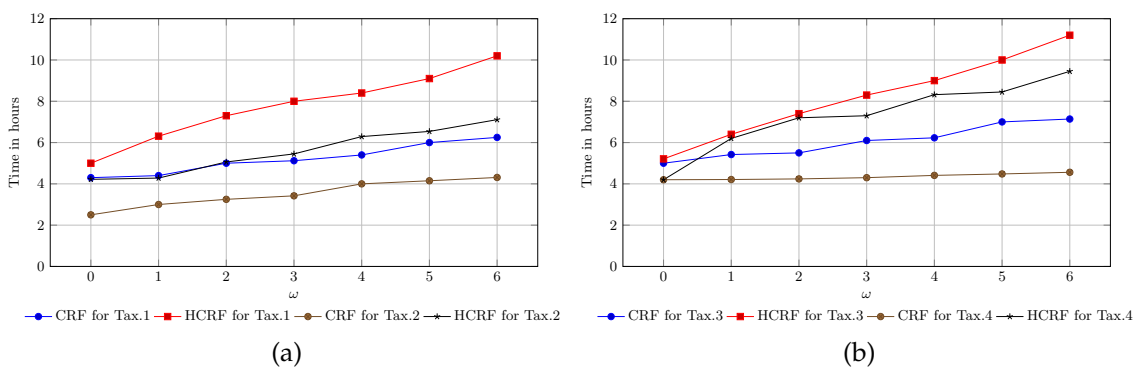


FIG. 6.14. CRFs and HCRFs training cost in terms of time: (a) Training cost of CRFs and HCRFs for Tax.1 and Tax.2, (b) training cost of CRFs and HCRFs for Tax.3 and Tax.4.

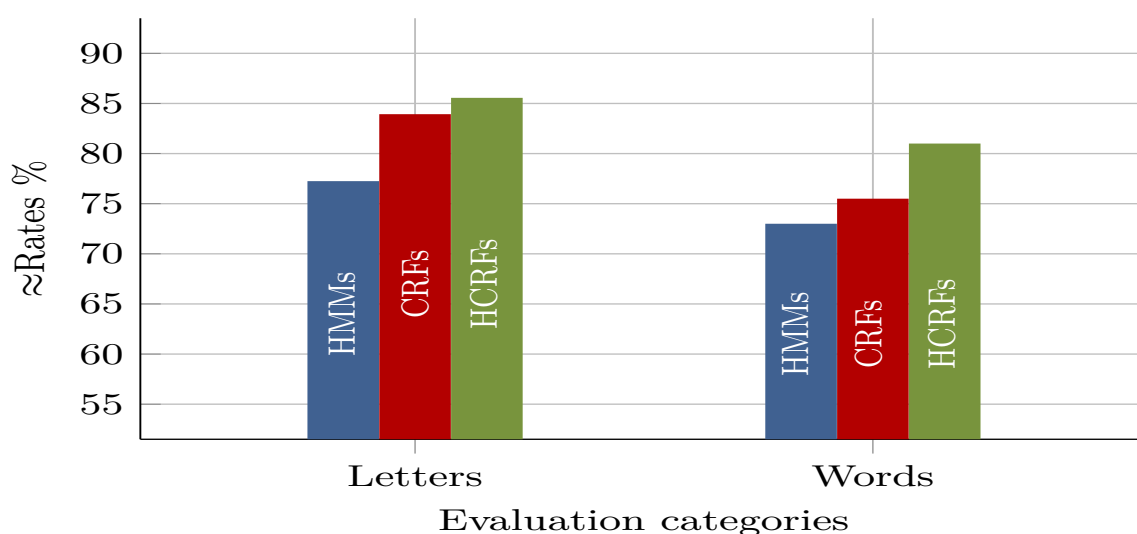


FIG. 6.15. Overall recognition performance: The average performance of the three approaches for recognition of segmented letters, and for recognized words with confidence values greater than 0.5.

6.4.3 HMMs vs. CRFs vs. HCRFs

In order to guarantee a fair comparison among the generative HMMs approach, the discriminative CRFs approach, and the discriminative with a hidden structure CRFs approach, all three approaches are trained and tested using the same training and evaluation datasets described in Section 6.4. Further, the recognition results of the three approaches are given on letters, as well as, on word levels, where the same metric is used in the evaluation process. In this section, we compare the performance of each approach to the rest, highlight the strengths and weakness, and give recommendations for a possible future application of each approach. The first part of FIG. 6.15 summarizes the average recognition rates of the three approaches. Due to the discriminative based training and the modeling of hidden pattern through the hidden states, HCRFs achieved the best performance rates of 85.60%, followed by the discriminatively trained CRFs that achieved 84.0% and the generative HMMs with a performance of 77.30%. The obtained recognition rates, firstly, confirm the efficiency of the discriminative models compared to the generative HMMs, and secondly, indicating the slight improvement in performance when hidden states layer is introduced to CRFs (i.e. HCRFs).

Compared to the results achieved by several approaches in [134], that are participated in the competition of the International Conference on Document Analysis

and Recognition (ICDAR 2009), our results are quite satisfactory and encouraging for two reasons: (i) the competing recognition systems consider only the isolated form of letters, hence only 30 class labels (at maximum) are dealt with, compared to processing 122 class labels in our approaches where each letter form is modeled. (ii) our letter models are built using letter shapes segmented directly from handwritten words rather than from isolated well written letters, which makes our approaches more applicable in realistic situations.

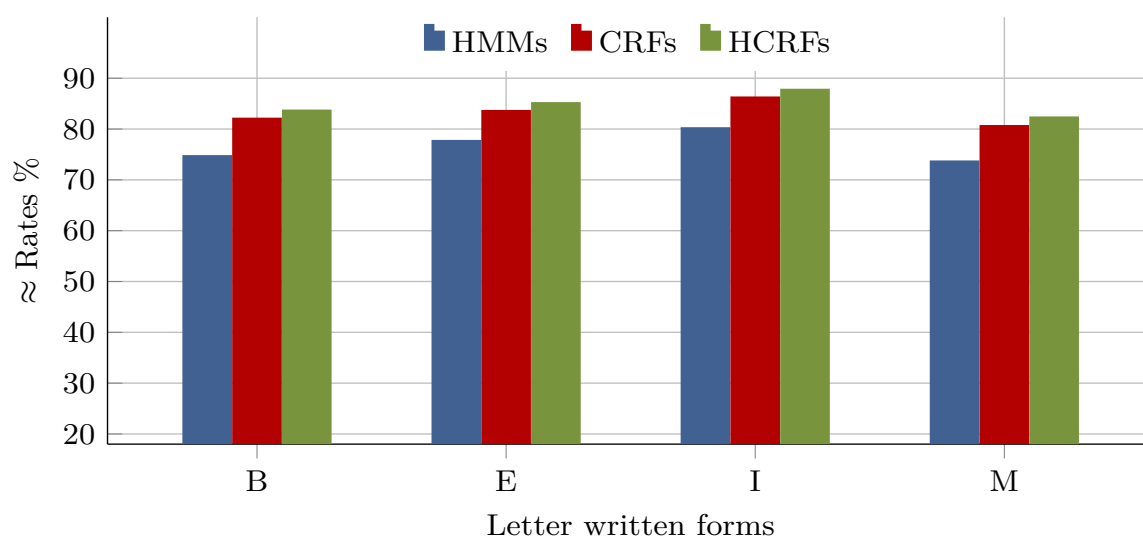


FIG. 6.16. Letter forms based performance: Performance comparison of the three approaches according to the different letter written forms.

By presenting the letter-based recognition results of each approach according to the letter different written forms (i.e. B, E, I, and M), one can easily see the HCRFs superiority, as shown in FIG.6.16. Moreover, the achieved results of 80.40%, 86.41%, and 87.93% for HMMs, CRFs and HCRFs, respectively, on samples in Isolated form (I), prove the relative ease and efficiency of molding letters in this form. Appendices C.1 and C.2, give detailed results using the three classification approaches.

The second part of FIG.6.15, illustrates the percentage of handwritten words that are partially / completely recognized with confidence values higher than 0.5. The three approaches show similar behaviors as in the letter recognition case. The HCRFs classifier attains the best performance of 81%, while CRFs and HMMs achieve performance of 75.5%, and 73%, respectively.

Even though, the word based recognition results are still unsatisfactory, specially the results with confidence values less than 0.8, we believe, the state-of-art spell

correction solutions such as MS spell checker, Google spell checker, or Hunspell for Arabic, can be used to improve performance significantly [135].

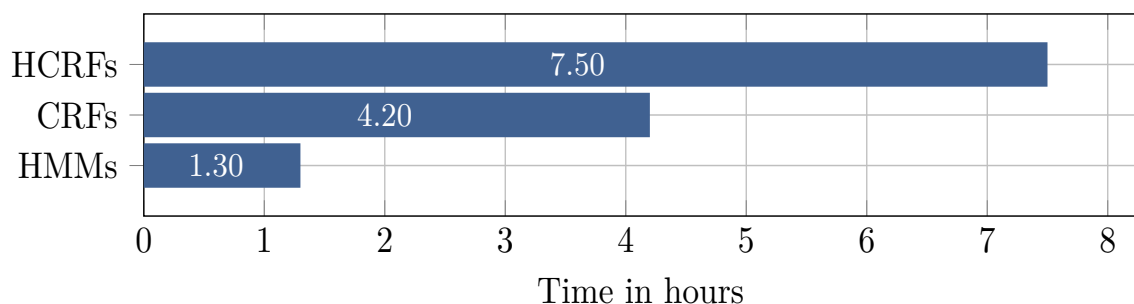


FIG. 6.17. Letter forms based performance: Performance comparison of the three approaches according to the different letter written forms.

However CRFs and HCRFs show a strong performance compared to HMMs, both are very expensive in terms of time and computation costs. FIG. 6.17, shows the average time needed for each approach to model a set of letters under one taxonomy. It is obvious that HMMs approach is significantly time-efficient compared to CRFs and HCRs. Furthermore, the high time costs of HCRFs compared to CRFs, should also be considered and justified by important performance improvements. Finally, we should emphasize the following points, (i) despite the fact that in our experiments, on both HCRFs and CRFs approaches outperform the HMMs, we expect, however, in application-specific small lexicon-based solutions, such as in banking and postal sectors, the gain in performance might not justify the choice of the expensive HCRFs or CRFs. Thus, in such cases we recommend to begin by investigating the performance of a HMMs solution, (ii) for unconstrained solutions with large or unlimited lexicons, and rather than directly deciding for a HCRFs based solution, we strongly recommend to firstly assessing the performance of the reasonably performing and less expensive CRFs based approach.

6.5 Conclusion

This chapter described our experiments conducted on the different proposed approaches. It started by experiments to optimize the parameters involved in the text line segmentation approach. To evaluate the approach, pages of modern and historical handwriting from two different databases have been used. A pixel matching between the groundtruthed images and the detected text lines, is used as a

performance metric. The rates of perfect matches, the precision, and recall are reported. The proposed approach showed satisfactory performance on various types of handwriting, which confirm its usability. Given its importance for the recognition of Arabic handwriting, the solution for the sub-words overlapping problem is also thoroughly tested using a sufficient number of samples from different databases. The achieved results are promising and prove the robustness of the proposed solution. The handwritten word segmentation approach has been validated mainly using samples from the IESK-arDB, where accuracy, precision and recall errors have been reported.

For recognition performance evaluation, HMMs, CRFs, and HCRFs are trained and tested on the same datasets that are drawn from two different databases. Besides the achieved recognition results on both the letters level and on the words level, the deletion, substitution, and insertion errors have been also reported and discussed. Finally, the performance comparison between the three approaches, along with discussing the pros and cons of each approach are given.

Conclusions & Future Perspectives

OPTICAL character recognition (OCR) is one of the most important image analysis tasks, that is carried out in a pipe of different processing steps with an ultimate goal of simulating the human ability to read and understand text. In our research work, the problem of unconstrained OCR for offline Arabic handwriting has been thoroughly investigated, and our contributions are presented in this dissertation. This final chapter is organized in two sections. In the first section, we summarize the thesis and highlight the key contributions. In the second section, possible future research directions for improvement and extension of our work will be suggested.

7.1 Summary and Key Contributions of the Thesis

After the introductory Chapter 1, Chapter 2 is mainly dedicated to help the reader deepen the understanding of the basics of the problem of automatic handwriting recognition. This chapter has adequately reviewed most of the common adopted processing steps from the signal acquisition until the classification, focusing primarily on approaches that we used or improved throughout our work. Furthermore,

special attention is paid to handwriting specific distortions, such as skewed handwritten words and the slanted ascenders or descenders. In this context, we have presented the popular Hough transform (HT) based technique and the contour Local Minima Regression (LMR) technique. These two techniques are combined together to provide an improved solution. As being the prerequisite for any unconstrained recognition OCR system, the bottleneck issue of handwriting segmentation is also addressed. The two different paradigms (i.e., explicit and implicit) of approaching the problem are explained. The chapter concluded by shedding light on the theoretical foundations of the generative Hidden Markov Models (HMMs), the discriminative based Conditional Random Fields (CRFs), and the Hidden-state CRFs (HCRFs).

Lack of extensive Arabic handwriting databases is one of the main reasons that hinder fast progress in the field of handwriting recognition. This can be attributed to the considerable time and effort needed for gathering, scanning and ground-truthing. Chapter 3, has presented our own database the IESK-arDB, a new multi-purpose off-line Arabic handwritten database. The database is free of charge and on-line available for research purposes. To facilitate research efforts of segmentation based recognition of handwritten Arabic words, the database contains about 6000 word images, saved in PNG format of three different types, namely, gray, binary, and thinned binary. Additionally, for each word image, an XML ground truth file that contains all needed data is added. A letter frequency analysis showed that the collection exhibits letter frequencies similar to that of large corpora of digital text, which proves the usefulness of the database. To the best of our knowledge, IESK-arDB is the only free database that offers a collection of Arabic handwritten manuscript images, along with Unicode transcription for each manuscript image that is line-wise aligned. The main purpose of this dataset is to leverage research in the field of recognition of handwritten text in historical documents, to allow searching and mining of historical documents. The collection

contains 285 page images taken from two medieval theological works, images are saved in PNG format, and each is manually transcribed in TXT files.

Furthermore, to automate the process of handwriting samples generation, we also presented an approach for handwriting synthesis. To build an Active Shape Model (ASM) for each letter shape, an on-line acquisition system consisting of a base unit and a wireless pen is used to collect handwritten letter samples from several writers. Samples of each shape are, first, aligned altogether and then their average shape, matrix of eigenvectors, and vector of eigen values are all used to compute synthesized instance from the letter shape class. In total, about 104 different ASMs are constructed to represent the letters basic shapes and the letters conditional shapes. Moreover, to make synthetic results look as genuine as possible, various types of affine transforms are used to add structural noise on both the letter level and on the word level.

The challenging problem of handwriting segmentation has many different aspects, each represents a research topic on its own. Chapter 4 is devoted to introduce our proposed solutions for the problem. Given the fundamental importance of handwritten text line segmentation, we suggested a solution that is simple, unspecific to the handwriting type, and computationally inexpensive. The proposed approach follows a top-down methodology, which begins with a page skew correction, then performs a voting process upon a set of horizontal profiles calculated from a set of sub-images cropped from the source image. The approach has achieved very good results on modern handwriting as well as on historical manuscripts.

The second and most important segmentation issue that we addressed is the segmentation of handwritten Arabic words into their constituent letters. Unlike Latin based scripts, Arabic sub-words/words overlapping issue should be handled first before any segmentation attempt. Thus, we introduced an overlapping resolving module that is carried out before segmentation takes place. This prior procedure facilitates the letter segmentation process by inserting enough empty

columns between overlapped sub-words/words. The segmentation process is then conducted by applying a set of heuristics that is formulated upon a group of pixels called critical feature points. Performance results on samples from two different databases confirm the efficiency of the proposed method.

Based on their theoretical foundations, probabilistic based classification approaches can be categorized into two different categories, namely, generative and discriminative. The main objective of Chapter 5 was to introduce probabilistic classification approaches that are capable of recognizing sequences of features extracted from handwritten Arabic words. Instead of using the commonly used sliding window based features (which are preferred when local details are a priority), in the chapter we begin by describing new sequential shape description features that are relatively cost-effective, yet able to capture more discriminative shape characteristics. Then, due to the fact that recognition performance is inversely proportional to the number of classes, letters are grouped into four basic shape categories using some primitive shape characteristics, e.g. number of segments and existence of a loop(s).

As a first alternative, we have proposed a generative based recognition approach using HMMs, where a model for each letter in each shape variation is constructed using EM algorithm. Moreover, an adaptive HMMs based threshold models are also built in order to enhance results by rejecting out-of-vocabulary and meaningless pattern. To further enhance performance, definitive recognition decisions are made by combining the results of two different types of models and their corresponding threshold models. In order to explore the discriminative based recognition alternative, the second part of the chapter has been mainly devoted to introduce two relatively recent discriminative based classifiers, namely, the linear-chain CRFs and its extension the hidden-state CRFs (HCRFs). For a fair comparison with the HMMs system, both CRFs- and HCRFs- systems are built on top of the same segmentation module and used the same set of features. Furthermore, the numbers of the hidden

states for HCRFs models are optimized as in the HMMs case.

Chapter 6 has described the experiments that have been conducted using our proposed approaches. It commences with experiments to optimize the parameters involved in the text line segmentation approach. To evaluate the approach, pages of modern and historical handwriting from two different databases are used. A pixel matching between the ground-truthed images and the detected text lines is used as a performance metric. The rates of perfect matches, the precision, and recall are also reported. The proposed approach showed satisfactory results on various types of handwriting which confirm its usability. Given its importance for the recognition of Arabic handwriting, the solution for the sub-words overlapping problem is also thoroughly tested using enough number of samples. The achieved results are promising and confirm the robustness of the proposed solution. The handwritten word segmentation approach is validated using samples from the IESK-arDB, where accuracy, precision and recall errors are all reported.

For recognition performance evaluation, HMMs, CRFs, and HCRFs are all trained and tested on the same datasets drawn from two different databases. Besides the achieved recognition results on both the letters level and on the words level, the deletion, substitution, and insertion errors are also reported and discussed. Finally, the performance comparisons of the three approaches along with a discussion of pros and cons of each approach are given.

7.2 Future Perspectives

Since our main motivation is to come up with unconstrained solutions for the problem of OCR for offline Arabic handwriting, the focus of our work has been directed toward segmentation based recognition solutions.

Even though our segmentation approach results are quite satisfactory, future work may investigate issues like, expanding Critical Feature Points (CFPs) set by

adding more topological features and more heuristics. Furthermore, we believe that prior context information (e.g., letter frequency distributions, n-Gram Statistics, etc.) can be employed to save unnecessary computation efforts and avoid many segmentation mistakes. Also a cyclic segmentation recognition based approach is expected to deliver better results where in such approach the segmentation decision will be confirmed first by a successful recognition. Alternatively, to avoid the high error tendency of the segmentation based approaches and also the constrained nature of holistic based approaches, sub-words based approaches will be investigated for the recognition handwritten words in future work.

Moreover, we think that more complex pre-classification steps are achievable by including diacritics, since several Arabic letters sharing the same basic shape and are only distinguishable through diacritics. Therefore, a further reduction in the number of class labels is a future improvement idea.

On feature extraction and prediction levels, using different types of features, classifiers, better optimization of involved models parameters, and text retrieval techniques integration are all possible future improvements. Also, a more complex CRFs based algorithm, namely, Latent-Dynamic Conditional Random Fields (LDCRF), has demonstrated a strong performance in fields such as NLP and bioinformatics [30]. Therefore, we plan to investigate its performance in the field of offline Arabic handwriting recognition.

Bibliography

- [1] M. Cheriet, N. Kharma, C.-l. Liu, and C. Suen, *Character Recognition Systems: A Guide for Students and Practitioners*. Wiley-Interscience, 2007.
- [2] Y. Ding, F. Kimura, and Y. Miyake, "Accuracy improvement of slant estimation for handwritten words," in *Proc. 15th ICPR*, pp. 527–530, 2000.
- [3] T. COHN, *Scaling Conditional Random Fields for Natural Language Processing*. PhD thesis, Department of Computer Science and Software Engineering, 2007.
- [4] M. Madi, "A study of Arabic letter frequency analysis." WWW page, 2011.
- [5] P. Xiu, L. Peng, X. Ding, and H. Wang, "Offline handwritten arabic character segmentation with probabilistic model," in *Proceedings of the 7th International Conference on Document Analysis Systems, DAS'06*, (Berlin, Heidelberg), pp. 402–412, Springer-Verlag, 2006.
- [6] A. J. Sellen and R. H. R. Harper, "The myth of the paperless office," 2001.
- [7] U. U. Ungor, *The Making of Modern Turkey - Nation and State in Eastern Anatolia, 1913-1950*. New York, London: OUP Oxford, 2012.
- [8] M. Elzobi, A. Al-Hamadi, Z. Al Aghbari, and L. Dings, "Iesk-aradb: a database for handwritten arabic and an optimized topological segmentation approach,"

- International Journal on Document Analysis and Recognition (IJ DAR)*, pp. 1–14, 2012.
- [9] L. Lorigo and V. Govindaraju, "Segmentation and pre-recognition of arabic handwriting," in *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, (Washington), pp. 605 – 609 Vol. 2, aug.-1 sept. 2005.
- [10] V. Märgner and H. El Abed, "Databases and competitions: strategies to improve arabic recognition systems," in *Proceedings of the 2006 conference on Arabic and Chinese handwriting recognition, SACH'06*, (Berlin, Heidelberg), pp. 82–103, Springer-Verlag, 2008.
- [11] S. Srihari, H. Srinivasan, P. Babu, and C. Bhole, "Handwritten arabic word spotting using the cedarabic document analysis system," in *Proc. Symposium on Document Image Understanding Technology (SDIUT-05)*, College Park, MD, pp. 123–132, 2005.
- [12] S. Al-Ma'adeed, D. Elliman, and C. Higgins, "A data base for arabic handwritten text recognition research," in *Frontiers in Handwriting Recognition, 2002. Proceedings. Eighth International Workshop on*, pp. 485–489, 2002.
- [13] Y. Al-Ohali, M. Cheriet, and C. Suen, "Databases for recognition of handwritten arabic cheques," *Pattern Recognition*, vol. 36, no. 1, pp. 111 – 121, 2003.
- [14] S. M. Strassel, "Linguistic resources for arabic handwriting recognition," in *Proceedings of the second International Conference for Arabic Handwriting Recognition*, 2009.

- [15] F. Slimane, R. Ingold, S. Kanoun, A. Alimi, and J. Hennebert, "Database and evaluation protocols for arabic printed text recognition," Tech. Rep. 296-09-01, University of Fribourg, Department of Informatics, 2009.
- [16] S. Schlosser, "Erim arabic database," *Document Processing Research Program, Information and Materials Applications Laboratory, Environmental Research Institute of Michigan*, 1995. On-line reference http://documents.cfar.umd.edu/resources/database/erim_Arabic_DB.html.
- [17] R. Farrahi Moghaddam, M. Cheriet, M. M. Adankon, K. Filonenko, and R. Wisnovsky, "Ibn sina: a database for research on processing and understanding of arabic manuscripts images," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, DAS '10*, (New York, NY, USA), pp. 11–18, ACM, 2010.
- [18] S. Mozaffari, H. El Abed, V. Maergner, K. Faez, and A. Amirshahi, *IfN/Farsi-Database: A Database of Farsi Handwritten City Names*. 2008.
- [19] M. Ziaratban, K. Faez, and F. Bagheri, "Fht: An unconstraint farsi handwritten text database," in *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, (Catalonia, Spain), pp. 281–285, july 2009.
- [20] M. Elzobi, A. Al-Hamadi, L. Dinges, and B. Michaelis, "A structural features based segmentation for off-line handwritten arabic text," in *I/V Communications and Mobile Network (ISVC), 2010 5th International Symposium on*, pp. 1–4, Sept 2010.
- [21] H. Almuallim and S. Yamaguchi, "A method of recognition of arabic cursive handwriting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, pp. 715–722, September 1987.

- [22] B. Bushofa, "Segmentation and recognition of arabic characters by structural classification," *Image and Vision Computing*, vol. 15, no. 3, pp. 167–179, 1997.
- [23] A. A. Atici and F. T. Yarman-Vural, "A heuristic algorithm for optical character recognition of arabic script," *Signal Processing*, vol. 62, no. 1, pp. 87 – 99, 1997.
- [24] I. S. I. Abuhaiba, M. J. J. Holt, and S. Datta, "Recognition of off-line cursive handwriting," *Computer Vision and Image Understanding*, vol. 71, no. 1, pp. 19 – 38, 1998.
- [25] M. Dehghan, K. Faez, M. Ahmadi, and M. Shridhar, "Handwritten farsi (arabic) word recognition: a holistic approach using discrete hmm," *Pattern Recognition*, vol. 34, no. 5, pp. 1057 – 1065, 2001.
- [26] M. Pechwitz and V. Maergner, "Hmm based approach for handwritten arabic word recognition using the ifn/enit - database," in *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*, pp. 890 – 894, aug. 2003.
- [27] R. Al-Hajj Mohamad, L. Likforman-Sulem, and C. Mokbel, "Combining slanted-frame classifiers for improved hmm-based arabic handwriting recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, pp. 1165 –1177, july 2009.
- [28] P. Dreuw, G. Heigold, and H. Ney, "Confidence- and margin-based mmi/mpe discriminative training for off-line handwriting recognition," *Int. J. Doc. Anal. Recognit.*, vol. 14, pp. 273–288, Sept. 2011.
- [29] M. Elzobi, A. Al-Hamadi, L. Dings, and S. El-Etriby, "Crfs and hcrfs based recognition for off-line arabic handwriting," in *Advances in Visual Computing - 11th International Symposium, ISVC 2015, Las Vegas, NV, USA, December 14-16, 2015, Proceedings, Part II*, pp. 337–346, 2015.

- [30] X.-D. Zhou, D.-H. Wang, F. Tian, C.-L. Liu, and M. Nakagawa, "Handwritten chinese/japanese text recognition using semi-markov conditional random fields.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 10, pp. 2413–2426, 2013.
- [31] S. Feng, R. Manmatha, and A. McCallum, "Exploring the use of conditional random field models and hmms for historical handwritten document recognition," in *the Proceedings of the 2nd IEEE International Conference on Document Image Analysis for Libraries (DIAL)*, pp. 30–37.
- [32] S. Mori, H. Nishida, and H. Yamada, *Optical Character Recognition*. New York, NY, USA: John Wiley & Sons, Inc., 1st ed., 1999.
- [33] V. Lavrenko, T. Rath, and R. Manmatha, "Holistic word recognition for handwritten historical documents," in *Document Image Analysis for Libraries, 2004. Proceedings. First International Workshop on*, (New York), pp. 278 – 287, ACM, 2004.
- [34] T. Steinherz, E. Rivlin, and N. Intrator, "Offline cursive script word recognition a survey," *International Journal on Document Analysis and Recognition*, vol. 2, pp. 90–110, 1999.
- [35] B. Yanikoglu and P. A. Sandon, "Segmentation of off-line cursive handwriting using lunar programming," *Pattern Recognition*, vol. 31, pp. 1825–1833, 1998.
- [36] M. LIWICKI and H. BUNKE, "Handwriting recognition of whiteboard notes studying the influence of training set size and type," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 21, no. 01, pp. 83–98, 2007.
- [37] M. R. Gupta, N. P. Jacobson, and E. K. Garcia, "{OCR} binarization and image pre-processing for searching historical documents," *Pattern Recognition*, vol. 40, no. 2, pp. 389 – 397, 2007.

- [38] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146–168, 2004.
- [39] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, pp. 62–66, Jan. 1979.
- [40] J. Sauvola and M. Pietikinen, "Adaptive document image binarization," *PATTERN RECOGNITION*, vol. 33, pp. 225–236, 2000.
- [41] W. Niblack, *An Introduction to Digital Image Processing*. Birkerød, Denmark, Denmark: Strandberg Publishing Company, 1985.
- [42] L. Lam, S.-W. Lee, and C. Suen, "Thinning methodologies—a comprehensive survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 9, pp. 869–885, 1992.
- [43] M. Wienecke, *Videobasierte Handschrifterkennung*. PhD thesis, Bielefeld University, 2003.
- [44] T. Ha and H. Bunke, "Image processing methods for document image analysis," in *Handbook of Character Recognition and Document Image Analysis* (H. Bunke and P. Wang, eds.), pp. 1–47, World Scientific, 1997.
- [45] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Commun. ACM*, vol. 27, pp. 236–239, Mar. 1984.
- [46] F. Stentiford and R. Mortimer, "Some new heuristics for thinning binary hand-printed characters for ocr," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-13, no. 1, pp. 81–84, 1983.
- [47] C. Neusius and J. Olszewski, "A noniterative thinning algorithm," *ACM Trans. Math. Softw.*, vol. 20, pp. 5–20, Mar. 1994.

- [48] P. D. Burns and D. Williams, "Identification of image noise sources in digital scanner evaluation," 2003.
- [49] A. K. Jain, *Fundamentals of Digital Image Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.
- [50] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Reading, MA: Addison-Wesley, 1992.
- [51] I. Young, J. Gerbrands, and L. J. van Vliet, *Image Processing Fundamentals*, 2014 (accessed January 23, 2014). <http://www.mif.vu.lt/atpazinimas/dip/FIP/fip.html>.
- [52] E. R. Dougherty and R. A. Lotufo, *Hands-on Morphological Image Processing (SPIE Tutorial Texts in Optical Engineering Vol. TT59)*. SPIE Publications, July 2003.
- [53] G. Kim and V. Govindaraju, "A lexicon driven approach to handwritten word recognition for real-time applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, pp. 366–379, Apr 1997.
- [54] G. Kim, V. Govindaraju, and S. N. Srihari, "An architecture for handwritten text recognition systems," *International Journal on Document Analysis and Recognition*, vol. 2, no. 1, pp. 37–44, 1999.
- [55] L. Lorigo and V. Govindaraju, "Offline arabic handwriting recognition: a survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, pp. 712–724, may 2006.
- [56] I. S. I. Abuhaiba, S. Mahmoud, and R. Green, "Recognition of handwritten cursive arabic characters," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, pp. 664–672, Jun 1994.

- [57] A. Bagdanov and J. Kanai, "Projection profile based skew estimation algorithm for jbig compressed images," in *Document Analysis and Recognition, 1997., Proceedings of the Fourth International Conference on*, vol. 1, pp. 401–405 vol.1, Aug 1997.
- [58] P. Burrow, "Arabic handwriting recognition," Master's thesis, School of Informatics University of Edinburgh, 2004.
- [59] H. Bunke and P. Wang, *Handbook of Character Recognition and Document Image Analysis*. World Scientific, 1997.
- [60] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, pp. 11–15, Jan. 1972.
- [61] M. van Ginkel, C. L. Hendriks, and L. van Vliet, "A short introduction to the radon and hough transforms and how they relate to each other," 2004.
- [62] L. A. Fernandes and M. M. Oliveira, "Real-time line detection through an improved hough transform voting scheme," *Pattern Recognition*, vol. 41, no. 1, pp. 299 – 314, 2008.
- [63] P. Slavik and V. Govindaraju, "Equivalence of different methods for slant and skew corrections in word recognition applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, pp. 323–326, Mar 2001.
- [64] R. Buse, Z.-Q. Liu, and T. Caelli, "A structural and relational approach to handwritten word recognition," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 27, pp. 847–861, Sep 1997.
- [65] C.-L. Liu, "Handwritten chinese character recognition: Effects of shape normalization and feature extraction," in *Arabic and Chinese Handwriting Recognition* (D. Doermann and S. Jaeger, eds.), vol. 4768 of *Lecture Notes in Computer Science*, pp. 104–128, Springer Berlin Heidelberg, 2008.

- [66] C.-L. Liu, M. Koga, H. Sako, and H. Fujisawa, "Aspect ratio adaptive normalization for handwritten character recognition," in *Advances in Multimodal Interfaces ICMI 2000* (T. Tan, Y. Shi, and W. Gao, eds.), vol. 1948 of *Lecture Notes in Computer Science*, pp. 418–425, Springer Berlin Heidelberg, 2000.
- [67] R. Casey and E. Lecolinet, "A survey of methods and strategies in character segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, pp. 690–706, Jul 1996.
- [68] M. Blumenstein and B. Verma, "A new segmentation algorithm for handwritten word recognition," in *Neural Networks, 1999. IJCNN '99. International Joint Conference on*, vol. 4, pp. 2893–2898 vol.4, 1999.
- [69] L. Kang, D. S. Doermann, H. Cao, R. Prasad, and P. Natarajan, "Local segmentation of touching characters using contour based shape decomposition," in *Document Analysis Systems* (M. Blumenstein, U. Pal, and S. Uchida, eds.), pp. 460–464, IEEE, 2012.
- [70] P. R. Cavalin, A. de Souza Britto, Jr., F. Bortolozzi, R. Sabourin, and L. E. S. Oliveira, "An implicit segmentation-based method for recognition of handwritten strings of characters," in *Proceedings of the 2006 ACM Symposium on Applied Computing, SAC '06*, (New York, NY, USA), pp. 836–840, ACM, 2006.
- [71] É. Caillault and C. Viard-Gaudin, "Using Segmentation Constraints in an Implicit Segmentation Scheme for On-line Word Recognition," in *Tenth International Workshop on Frontiers in Handwriting Recognition* (G. Lorette, ed.), (La Baule (France)), Université de Rennes 1, Suvisoft, Oct. 2006. <http://www.suvisoft.com> Université de Rennes 1.
- [72] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," 2001.

- [73] I. Ulusoy and C. Bishop, "Comparison of generative and discriminative techniques for object detection and classification," in *Toward Category-Level Object Recognition* (J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, eds.), vol. 4170 of *Lecture Notes in Computer Science*, pp. 173–195, Springer Berlin Heidelberg, 2006.
- [74] C. M. Bishop and J. Lasserre, "Generative or Discriminative? Getting the Best of Both Worlds," in *Bayesian Statistics 8* (J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, eds.), pp. 3–24, International Society for Bayesian Analysis, Oxford University Press, 2007.
- [75] J.-H. Xue and D. Titterton, "Comment on on discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Neural Processing Letters*, vol. 28, no. 3, pp. 169–187, 2008.
- [76] V. Frinken, A. Fischer, M. Baumgartner, and H. Bunke, "Keyword spotting for self-training of {BLSTM} {NN} based handwriting recognition systems," *Pattern Recognition*, vol. 47, no. 3, pp. 1073 – 1082, 2014. Handwriting Recognition and other {PR} Applications.
- [77] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, Feb 1989.
- [78] M. Elmezain, A. Al-Hamadi, J. Appenrodt, and B. Michaelis, "A hidden markov model-based continuous gesture recognition system for hand motion trajectory," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pp. 1–4, Dec 2008.
- [79] K. Riedhammer, T. Bocklet, A. Ghoshal, and D. Povey, "Revisiting semi-continuous hidden markov models," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pp. 4721–4724, March 2012.

- [80] J. Lafferty, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," pp. 282–289, Morgan Kaufmann, 2001.
- [81] H. M. Wallach, "Conditional random fields: An introduction," tech. rep., 2004.
- [82] M. Elmezain, A. Hamadi, and B. Michaelis, *Hand Gesture Spotting and Recognition Using HMMs and CRFs in Color Image Sequences*. 2010.
- [83] D. Pinto, A. McCallum, X. Wei, and W. B. Croft, "Table extraction using conditional random fields," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, SIGIR '03*, (New York, NY, USA), pp. 235–242, ACM, 2003.
- [84] L. Dinges, A. Al-Hamadi, and M. Elzobi, "An approach for arabic handwriting synthesis based on active shape models," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pp. 1260–1264, Aug 2013.
- [85] T. Nawaz, S. Naqvi, H. Rehman, and A. Faiz, "Optical character recognition system for urdu (naskh font) using pattern matching technique," *International Journal of Image Processing*, vol. 3, no. 3, pp. 92–104, 2008.
- [86] R. Rosenfeld, "Two decades of statistical language modeling: Where do we go from here," in *Proceedings of the IEEE*, p. 2000, 2000.
- [87] G. A. ABANDAH and M. Z. KHEDHER, "Analysis of handwritten arabic letters using selected feature extraction techniques," *International Journal of Computer Processing of Languages*, vol. 22, no. 01, pp. 49–73, 2009.
- [88] Wikipedia, "Frequency analysis — wikipedia, the free encyclopedia," 2014. [Online; accessed 28-March-2014].

- [89] J. Wang, C. Wu, Y.-Q. Xu, and H.-Y. Shum, "Combining shape and physical models for online cursive handwriting synthesis," *Int. J. Doc. Anal. Recognit.*, vol. 7, pp. 219–227, Sept. 2005.
- [90] L. Dinges, M. Elzobi, A. Al-Hamadi, and Z. A. Aghbari, *Synthizing Handwritten Arabic Text Using Active Shape Models*, pp. 401–408. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.
- [91] L. Dinges, A. Al-Hamadi, M. Elzobi, and S. El-etriby, "Synthesis of common arabic handwritings to aid optical character recognition research," *Sensors*, vol. 16, no. 3, p. 346, 2016.
- [92] SilverCrest, "Digital pen for digitally recording handwritten notes @ONLINE," June 2013.
- [93] T. Cootes, E. Baldock, and J. Graham, "An introduction to active shape models," *Image Processing and Analysis*, pp. 223–248, 2000.
- [94] Y. Elarian, H. A. Al-Muhsateb, and L. M. Ghouti, "Arabic handwriting synthesis," First International Workshop on Frontiers in Arabic Handwriting Recognition (<http://hdl.handle.net/2003/27562>), 2011.
- [95] L. Likforman-Sulem, A. Zahour, and B. Taconet, "Text line segmentation of historical documents: a survey," *International Journal of Document Analysis and Recognition (IJ DAR)*, vol. 9, no. 2-4, pp. 123–138, 2007.
- [96] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, "Text line and word segmentation of handwritten documents," *Pattern Recognition*, vol. 42, no. 12, pp. 3169 – 3183, 2009. New Frontiers in Handwriting Recognition.
- [97] M. Feldbach, *Segmentierung und strukturbasierte adaptive Erkennung von Gebrauchsschrift in historischen Dokumenten*. PhD thesis, Otto-von-Guericke Universität Magdeburg, 2006.

- [98] K. Wong, R. Casey, and F. Wahl, "Document analysis system," *IBM Journal of Research and Development*, vol. 26, pp. 647–656, Nov 1982.
- [99] F. Shafait, D. Keysers, and T. Breuel, "Performance evaluation and benchmarking of six-page segmentation algorithms," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, pp. 941–954, June 2008.
- [100] Y. Li, Y. Zheng, D. S. Doermann, and S. Jaeger, "Script-independent text line segmentation in freestyle handwritten documents.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 8, pp. 1313–1329, 2008.
- [101] W. Boussellaa, A. Zahour, H. Elabed, A. Benabdelhafid, and A. M. Alimi, "Unsupervised block covering analysis for text-line segmentation of arabic ancient handwritten document images," in *Proceedings of the 2010 20th International Conference on Pattern Recognition, ICPR '10*, (Washington, DC, USA), pp. 1929–1932, IEEE Computer Society, 2010.
- [102] H. Boubaker, M. Kherallah, and A. M. Alimi, "New algorithm of straight or curved baseline detection for short arabic handwritten writing," in *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition, ICDAR '09*, (Washington, DC, USA), pp. 778–782, IEEE Computer Society, 2009.
- [103] H. Samet and M. Tamminen, "Efficient component labeling of images of arbitrary dimension represented by linear bintrees," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 10, pp. 579–586, Jul 1988.
- [104] L. Vincent, "Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms," *IEEE Transactions on Image Processing*, vol. 2, pp. 176–201, 1993.

- [105] P. Dreuw, S. Jonas, and H. Ney, "White-space models for offline arabic handwriting recognition," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pp. 1–4, Dec 2008.
- [106] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, (Stroudsburg, PA, USA), pp. 1070–1079, Association for Computational Linguistics, 2008.
- [107] N. Nguyen and Y. Guo, "Comparisons of sequence labeling algorithms and extensions," in *Proceedings of the 24th International Conference on Machine Learning, ICML '07*, (New York, NY, USA), pp. 681–688, ACM, 2007.
- [108] C. Sutton and A. McCallum, "An introduction to conditional random fields," 2010.
- [109] T. Ploetz and G. Fink, "Markov models for offline handwriting recognition: a survey," *International Journal on Document Analysis and Recognition (IJ DAR)*, vol. 12, pp. 269–298, 2009.
- [110] L. Likforman-Sulem, R. AlHajjMohammad, C. Mokbel, F. Menasri, A.-L. Bianne-Bernard, and C. Kermorvant, "Features for hmm-based arabic handwritten word recognition systems," in *Guide to OCR for Arabic Scripts* (V. Mrgner and H. El Abed, eds.), pp. 123–143, Springer London, 2012.
- [111] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 509–522, Apr 2002.
- [112] G. Mori, S. Belongie, and J. Malik, "Efficient shape matching using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, pp. 1832–1837, Nov 2005.

- [113] R. Saabni and J. El-Sana, "Keywords image retrieval in historical handwritten arabic documents," *Journal of Electronic Imaging*, vol. 22, no. 1, pp. 013016–013016, 2013.
- [114] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: investigation of normalization and feature extraction techniques," *Pattern Recognition*, vol. 37, no. 2, pp. 265 – 279, 2004.
- [115] D. Dueck, *Affinity Propagation: Clustering Data by Passing Messages*. Canadian theses, University of Toronto, 2009.
- [116] Y. Zhu, J. Yu, and C. Jia, "Initializing k-means clustering using affinity propagation," in *Hybrid Intelligent Systems, 2009. HIS '09. Ninth International Conference on*, vol. 1, pp. 338–343, Aug 2009.
- [117] J. Geiger, J. Schenk, F. Wallhoff, and G. Rigoll, "Optimizing the number of states for hmm-based on-line handwritten whiteboard recognition," in *Frontiers in Handwriting Recognition (ICFHR), 2010 International Conference on*, pp. 107–112, Nov 2010.
- [118] M. Zimmermann and H. Bunke, "Hidden markov model length optimization for handwriting recognition systems," in *Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), IWFHR '02, (Washington, DC, USA)*, pp. 369–, IEEE Computer Society, 2002.
- [119] Y. Al-Ohali, M. Cheriet, and C. Suen, "Introducing termination probabilities to hmm," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 3, pp. 319–322 vol.3, 2002.
- [120] M. Elzobi, A. Al-Hamadi, L. Dings, M. Elmezain, and A. Saeed, "A hidden markov model-based approach with an adaptive threshold model for off-line arabic handwriting recognition.," in *ICDAR*, pp. 945–949, 2013.

- [121] D. Willett, A. Worm, C. Neukirchen, and G. Rigoll, "Confidence measures for hmm-based speech recognition," in *in ICSLP98*, pp. 3241–3244.
- [122] H.-K. Lee and J. Kim, "An hmm-based threshold model approach for gesture recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 21, pp. 961–973, Oct 1999.
- [123] M.-Y. Chen, A. Kundu, and J. Zhou, "Off-line handwritten word recognition using a hidden markov model type stochastic network," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, pp. 481–496, May 1994.
- [124] Wikipedia, "Word error rate — wikipedia, the free encyclopedia," 2014. [Online; accessed 6-January-2015].
- [125] D. Willett, A. Worm, C. Neukirchen, and G. Rigoll, "Confidence measures for hmm-based speech recognition," in *in ICSLP98*, pp. 3241–3244, 1998.
- [126] D. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, (San Francisco, CA, USA), pp. 282–289, Morgan Kaufmann Publishers Inc., 2001.
- [127] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas, "Conditional models for contextual human motion recognition," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, pp. 1808–1815 Vol. 2, Oct 2005.
- [128] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Math. Program.*, vol. 45, pp. 503–528, Dec. 1989.
- [129] A. Quattoni, S. Wang, L. p Morency, M. Collins, T. Darrell, and M. Csail, "Hidden-state conditional random fields," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

- [130] I. T. Phillips and A. K. Chhabra, "Empirical performance evaluation of graphics recognition systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, pp. 849–870, Sept. 1999.
- [131] B. Gatos, N. Stamatopoulos, and G. Louloudis, "Icdar 2009 handwriting segmentation contest," in *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pp. 1393–1397, July 2009.
- [132] N. Stamatopoulos, B. Gatos, G. Louloudis, U. Pal, and A. Alaei, "Icdar 2013 handwriting segmentation contest," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pp. 1402–1406, Aug 2013.
- [133] J. Kumar, W. Abd-Almageed, L. Kang, and D. Doermann, "Handwritten arabic text line segmentation using affinity propagation," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, DAS '10*, (New York, NY, USA), pp. 135–142, ACM, 2010.
- [134] S. Mozaffari and H. Soltanizadeh, "Icdar 2009 handwritten farsi/arabic character recognition competition.," in *ICDAR*, pp. 1413–1417, IEEE Computer Society, 2009.
- [135] K. Shaalan, M. Attia, P. Pecina, Y. Samih, and J. van Genabith, "Arabic word generation and modelling for spell checking," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)* (N. C. C. Chair), K. Choukri, T. Declerck, M. U. Dogan, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, eds.), (Istanbul, Turkey), European Language Resources Association (ELRA), may 2012.

IESK-arDB Database and Handwriting Synthesis

Our IESK-arDB database is still growing to include more samples and to cover various types of handwritten documents. FIG.A.1 presents an example from the IESK-arDB manuscript collection where each page in the collection is transcribed (line-by-line alignment) into Unicode text. FIG.A.2 and FIG. A.3, respectively, show the capability of the proposed handwriting synthesis approach of generating single word images as well as images corresponding to blocks of text.

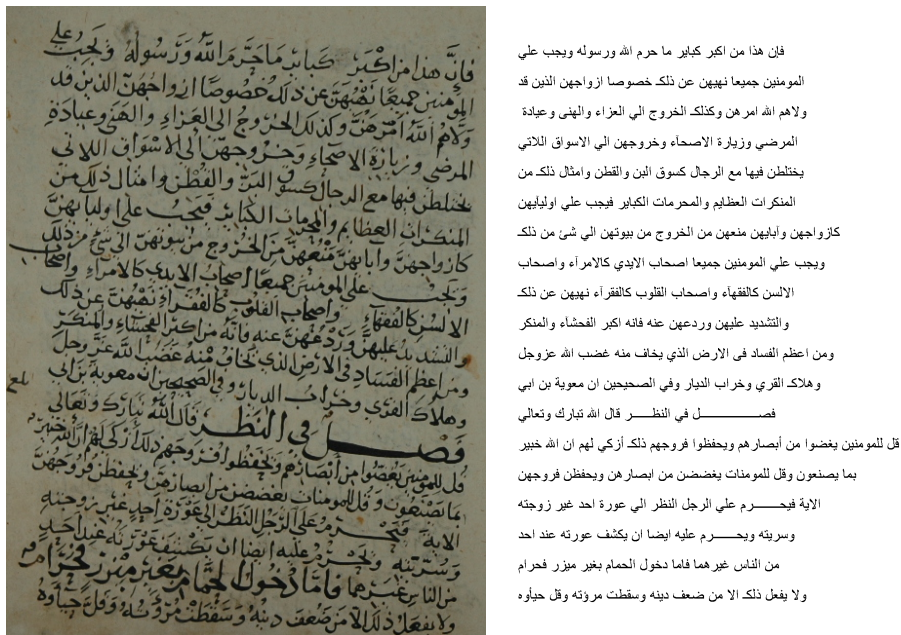


FIG. A.1. A manuscript page and its corresponding transcription.

تعلم علي من
الناس الحقيقة المقبل
البداية المال الوقت

FIG. A.2. Samples of single words synthesizing.

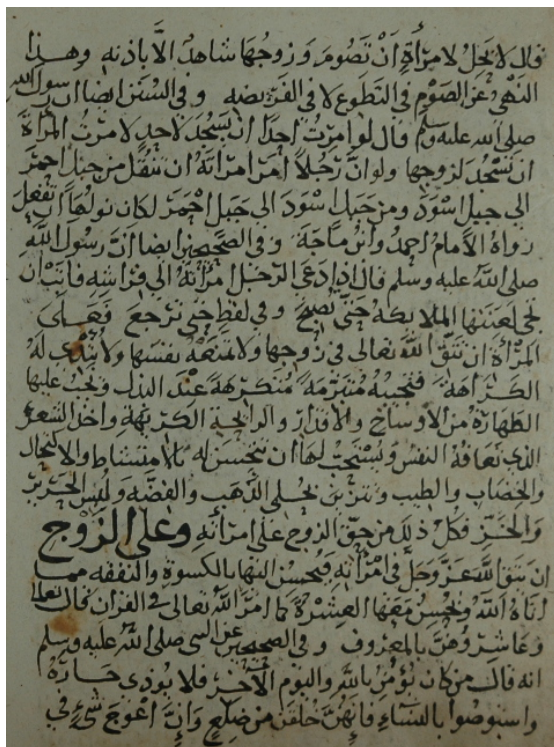
<p>من اكبر الكباير التي يكرها الله ورسوله والمؤمنون وهو اصل النفاق و اساس الباطل وراس مال الشيطان فان الشيطان اول من شرع الحيله والخداع كما اخبر الله تعالى في القران عنه في قصته مع ابينا آدم وحوي عليهما السلم فيستحيل ان يكون التحيل مشروعا على اسقاط شئ من الواجبات او تحليل شئ من المحرمات ومن كان من عذاب ربه مشققا وعلى دينه خائفا وبحسابه على مثاقيل الذر من اعماله في الاخره موقنا فانه يكتفي بالتلويح دون التصريح والذي صربت علي قلبه الغفله واحاط به سرادقها واتبع هواه وكان امره فرطا حتى صار قلبه مرباءد اسود كالكون محجبا لا يعرف معروفه ولا ينكر منكرا الا ما اشرب من هواه لا يقل يصح ناصح ولا يلو علي عدل عادل ولو جاءه يكل ايه فنسأل الله تعالى ان يهدينا الي صراطه المستقيم صراط الذين انعم عليهم من النبيين والصديقين والشهداء والصالحين وان يمسكنا بدينه القويم ويعصمنا بكتابه الحق المبين ويجعلنا مستمسكين به بفضلته وبرحمته امين والحمد لله رب العالمين وصلواته على عبده ورسوله واله اجمعين</p>	<p>من اكبر الكباير التي يكرها الله ورسوله والمؤمنون وهو اصل النفاق و اساس الباطل وراس مال الشيطان فان الشيطان اول من شرع الحيله والخداع كما اخبر الله تعالى في القران عنه في قصته مع ابينا آدم وحوي عليهما السلم فيستحيل ان يكون التحيل مشروعا على اسقاط شئ من الواجبات او تحليل شئ من المحرمات ومن كان من عذاب ربه مشققا وعلى دينه خائفا وبحسابه على مثاقيل الذر من اعماله في الاخره موقنا فانه يكتفي بالتلويح دون التصريح والذي صربت علي قلبه الغفله واحاط به سرادقها واتبع هواه وكان امره فرطا حتى صار قلبه مرباءد اسود كالكون محجبا لا يعرف معروفه ولا ينكر منكرا الا ما اشرب من هواه لا يقل يصح ناصح ولا يلو علي عدل عادل ولو جاءه يكل ايه فنسأل الله تعالى ان يهدينا الي صراطه المستقيم صراط الذين انعم عليهم من النبيين والصديقين والشهداء والصالحين وان يمسكنا بدينه القويم ويعصمنا بكتابه الحق المبين ويجعلنا مستمسكين به بفضلته وبرحمته امين والحمد لله رب العالمين وصلواته على عبده ورسوله واله اجمعين</p>
---	---

FIG. A.3. An example of a line-by-line synthesizing of Arabic handwriting from block of Unicode text.

Segmentation Results

This appendix shows sample results of the solutions proposed for the problems of text line segmentation, overlap of sub-words, and word segmentation.

B.1 Text line segmentation



خُصُوصًا وَعُمُومًا فَهَذَا بَابٌ وَاسِعٌ يُجَنَّبُ الْيَوْمَ مِنْهُ بِحَسَبِ
 حَاضِرِي أَنْ أَمْرًا الْيَوْمَ مِنْ عَمْرٍ مِنْ الْخَطَابِ رَضِيَ بِهِ عَنْهُ وَزِدْ عَلَيْهِ حَقًّا
 بِنَفْسِهِ كَانَ عَلَيْهِ فَمَنْ أَحَدُهَا سَعِيَّةٌ دَرَاهِمًا وَالْآخِرُ بَانِي عَشْرٍ
 دَرَاهِمًا فَخَلَعَ الْفَمِصْرَ الَّذِي سِنُّهُ عَشْرٌ دَرَاهِمًا وَلَيْسَ الْفَمِصْرُ الَّذِي بَانِي
 عَشْرٌ حَسْبُهُ أَنْ يَرَاهُ عَشْرٌ فَمَنْ مَوَدَّةٌ فَلَمَّا دَخَلَ عَلَيْهِ وَرَأَاهُ وَعَلَيْهِ الْفَمِصْرُ
 الَّذِي بَانِي عَشْرٌ مَسَّنَهُ عَمْرٌ رَضِيَ بِهِ عَنْهُ بَدَأَ فَقَالَ لَهُ الْفَمِصْرُ
 لَيْتَنِي بَكَرْتُ أَخَذْتَهُ وَقَالَ يَا أَمْرًا الْيَوْمَ مِنْ أَخَذْتَهُ بَانِي عَشْرٌ دَرَاهِمًا
 قَالَ عَمْرٌ فَهَلَّا جَعَلْتَهُ سِنُّهُ دَرَاهِمًا وَجَعَلْتَهُ السِّنُّ الْآخِرِيَّ مَا
 تَعْلَمُ يَعْنِي لَدَى لَدَى لَدَى وَفِي الصَّحِيحِ عَنْ ابْنِ مَسْعُودٍ رَضِيَ اللَّهُ
 عَنْهُ قَالَ قَالَ رَسُولُ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ بَاهُزْنَا يَا أَصْدِقَ قَدِّ فَيَنْطَلِقُ حَتَّى يَأْتِيَ
 إِلَى السُّوقِ فَيَجْمَلُ فَيُصَلِّبُ الْمُدَّ مَعْنَاهُ أَنْ رَسُولُ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ
 وَسَلَّمَ كَانَ يَأْتِي أَهْلًا بِالْأَصْدَقِ فِيهِ وَفِيهِمْ مَنْ لَا قَدْرَةَ لَهُ فَيَنْطَلِقُ فَيَجْمَلُ عَلَى
 ظَهْرِهِ بِالْآخِرِ ثُمَّ يَنْصَلِقُ وَفِي الصَّحِيحِ مِنْ بَعْضِ عُرَاةٍ مِنْ بَنِي رَضِيَ اللَّهُ
 عَنْهُ عَنِ النَّبِيِّ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ قَالَ عَلِيٌّ كُلُّ مَسَاءٍ صِدْقَةٌ وَالْوَأْتِ بِرَسُولِ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ
 يُجَدُّ قَالَ يَجْمَلُ بِهَا فَيَقْبَلُ النَّاسُ وَيَصْدُقُ قَالَ الْوَأْتِ بِالْمَرْجَلِ وَالْمَرْجَلُ
 ذَا الْحَاجَةِ الْمَنْهُوفِ قَالَ الْوَأْتِ بِالْمَرْجَلِ وَالْمَرْجَلُ وَالْمَرْجَلُ
 عَنِ النَّبِيِّ فَأَتَاهَا صَدَقَةٌ وَقَالَ اللَّهُ تَعَالَى يَا أَيُّهَا الَّذِينَ آمَنُوا
 انْفَعُوا مِمَّا زَكَّيْتُمْ مِنْ قَبْلِ أَنْ يَأْتِيَنَا يَوْمَ لَا يَسْعَى فِيهِ وَلَا خَلَّةٌ وَلَا سَفَلَةٌ
 وَقَالَ تَعَالَى سَارِعُوا إِلَى مَعْفَةٍ مِنْ رَبِّكُمْ وَحِذِّعُوا عُرُوشَ السُّؤَالِ وَالْأَرْضِ
 أَعْدَابُ الْعَنْقَبَاتِ الَّذِينَ يَنْفَعُونَ فِي السَّرَّاءِ وَالضَّرَّاءِ وَالْكَاطِبِينَ الْغِيظِ وَالْعَا
 عِلَّ النَّاسِ وَاللَّعْنَةُ عَلَى الْجَحِيمِينَ قَوْلُهُ سَيِّئَةٌ أَنْفَعُوا مِمَّا زَكَّيْتُمْ عَامَّةً

خُصُوصًا وَعُمُومًا فَهَذَا بَابٌ وَاسِعٌ يُجَنَّبُ الْيَوْمَ مِنْهُ بِحَسَبِ
 حَاضِرِي أَنْ أَمْرًا الْيَوْمَ مِنْ عَمْرٍ مِنْ الْخَطَابِ رَضِيَ بِهِ عَنْهُ وَزِدْ عَلَيْهِ حَقًّا
 بِنَفْسِهِ كَانَ عَلَيْهِ فَمَنْ أَحَدُهَا سَعِيَّةٌ دَرَاهِمًا وَالْآخِرُ بَانِي عَشْرٍ
 دَرَاهِمًا فَخَلَعَ الْفَمِصْرَ الَّذِي سِنُّهُ عَشْرٌ دَرَاهِمًا وَلَيْسَ الْفَمِصْرُ الَّذِي بَانِي
 عَشْرٌ حَسْبُهُ أَنْ يَرَاهُ عَشْرٌ فَمَنْ مَوَدَّةٌ فَلَمَّا دَخَلَ عَلَيْهِ وَرَأَاهُ وَعَلَيْهِ الْفَمِصْرُ
 الَّذِي بَانِي عَشْرٌ مَسَّنَهُ عَمْرٌ رَضِيَ بِهِ عَنْهُ بَدَأَ فَقَالَ لَهُ الْفَمِصْرُ
 لَيْتَنِي بَكَرْتُ أَخَذْتَهُ وَقَالَ يَا أَمْرًا الْيَوْمَ مِنْ أَخَذْتَهُ بَانِي عَشْرٌ دَرَاهِمًا
 قَالَ عَمْرٌ فَهَلَّا جَعَلْتَهُ سِنُّهُ دَرَاهِمًا وَجَعَلْتَهُ السِّنُّ الْآخِرِيَّ مَا
 تَعْلَمُ يَعْنِي لَدَى لَدَى لَدَى وَفِي الصَّحِيحِ عَنْ ابْنِ مَسْعُودٍ رَضِيَ اللَّهُ
 عَنْهُ قَالَ قَالَ رَسُولُ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ بَاهُزْنَا يَا أَصْدِقَ قَدِّ فَيَنْطَلِقُ حَتَّى يَأْتِيَ
 إِلَى السُّوقِ فَيَجْمَلُ فَيُصَلِّبُ الْمُدَّ مَعْنَاهُ أَنْ رَسُولُ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ
 وَسَلَّمَ كَانَ يَأْتِي أَهْلًا بِالْأَصْدَقِ فِيهِ وَفِيهِمْ مَنْ لَا قَدْرَةَ لَهُ فَيَنْطَلِقُ فَيَجْمَلُ عَلَى
 ظَهْرِهِ بِالْآخِرِ ثُمَّ يَنْصَلِقُ وَفِي الصَّحِيحِ مِنْ بَعْضِ عُرَاةٍ مِنْ بَنِي رَضِيَ اللَّهُ
 عَنْهُ عَنِ النَّبِيِّ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ قَالَ عَلِيٌّ كُلُّ مَسَاءٍ صِدْقَةٌ وَالْوَأْتِ بِرَسُولِ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ
 يُجَدُّ قَالَ يَجْمَلُ بِهَا فَيَقْبَلُ النَّاسُ وَيَصْدُقُ قَالَ الْوَأْتِ بِالْمَرْجَلِ وَالْمَرْجَلُ
 ذَا الْحَاجَةِ الْمَنْهُوفِ قَالَ الْوَأْتِ بِالْمَرْجَلِ وَالْمَرْجَلُ وَالْمَرْجَلُ
 عَنِ النَّبِيِّ فَأَتَاهَا صَدَقَةٌ وَقَالَ اللَّهُ تَعَالَى يَا أَيُّهَا الَّذِينَ آمَنُوا
 انْفَعُوا مِمَّا زَكَّيْتُمْ مِنْ قَبْلِ أَنْ يَأْتِيَنَا يَوْمَ لَا يَسْعَى فِيهِ وَلَا خَلَّةٌ وَلَا سَفَلَةٌ
 وَقَالَ تَعَالَى سَارِعُوا إِلَى مَعْفَةٍ مِنْ رَبِّكُمْ وَحِذِّعُوا عُرُوشَ السُّؤَالِ وَالْأَرْضِ
 أَعْدَابُ الْعَنْقَبَاتِ الَّذِينَ يَنْفَعُونَ فِي السَّرَّاءِ وَالضَّرَّاءِ وَالْكَاطِبِينَ الْغِيظِ وَالْعَا
 عِلَّ النَّاسِ وَاللَّعْنَةُ عَلَى الْجَحِيمِينَ قَوْلُهُ سَيِّئَةٌ أَنْفَعُوا مِمَّا زَكَّيْتُمْ عَامَّةً

فَمَادَامَ الْعَيْدُ مَسْتَقْبَلًا أَمْرًا قَوْمًا بِالْحَقِّ عَلَى نَفْسِهِ وَعَنْ نَفْسِهِ فَإِنَّ
 اللَّهَ سَيِّئَةٌ تَكْلَافَةٌ وَتُجْفِظَةٌ وَبُرْذُومَةٌ مِنْ قَضَائِهِ وَتَعْصَمُهُ مِنَ السَّرِّ
 وَأَهْلِهِ وَتَجْعَلُ جَمِيعَ مَا تَجِدُكَ وَتُجَدُّ لَكَ مِنْ مَكْرُوهَاتِ نَفْسِهِ خَيْرًا
 وَزِيَادَةً فِي الْخَيْرِ مَعَ طَاعَةِ اللَّهِ وَرَسُولِهِ قَالَ مَنْ سَوَّاهُ لِحُجْلِهِ لَمْ يَخْرُجْ
 إِلَى قَوْلِهِ فَجَعَلَ اللَّهُ لِكُلِّ شَيْءٍ قَدْرًا

فَمَادَامَ الْعَيْدُ مَسْتَقْبَلًا أَمْرًا قَوْمًا بِالْحَقِّ عَلَى نَفْسِهِ وَعَنْ نَفْسِهِ فَإِنَّ
 اللَّهَ سَيِّئَةٌ تَكْلَافَةٌ وَتُجْفِظَةٌ وَبُرْذُومَةٌ مِنْ قَضَائِهِ وَتَعْصَمُهُ مِنَ السَّرِّ
 وَأَهْلِهِ وَتَجْعَلُ جَمِيعَ مَا تَجِدُكَ وَتُجَدُّ لَكَ مِنْ مَكْرُوهَاتِ نَفْسِهِ خَيْرًا
 وَزِيَادَةً فِي الْخَيْرِ مَعَ طَاعَةِ اللَّهِ وَرَسُولِهِ قَالَ مَنْ سَوَّاهُ لِحُجْلِهِ لَمْ يَخْرُجْ
 إِلَى قَوْلِهِ فَجَعَلَ اللَّهُ لِكُلِّ شَيْءٍ قَدْرًا

فَصَلِّ فِي الْحَجِّ مِنْ مَنِّ فِتْنَةِ النِّسَاءِ قَالَ اللَّهُ تَعَالَى
 رَزَقْنَا نِسَاءَ الْجَنَّةِ شَهْوَاتٍ مِنَ النِّسَاءِ وَالْبَنِينَ وَالْقَنَاطِيرَ الْمُقَنْطَرَةَ مِنَ
 الذَّهَبِ وَالْفِضَّةِ وَالْخَيْلَ الْمُسَوَّمَةَ مَهْرًا لِأَنْفُسِهِمْ وَأَلْحَبْتُمْ فِي ذَلِكَ
 مَعْدَاءَ نِسَاءٍ لَأَنْهَزْنَا كَثِيرَ الشَّهْوَاتِ الَّتِي تَهْوِيهَا النَّفْسُ فِي الْفُجْهِينِ
 عَنْ رَسُولِ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ أَنَّهُ قَالَ مَا تَرَكْتُ فِتْنَةً أَضْرَعُ عَلَى الرَّجَالِ
 مِنَ النِّسَاءِ وَقَالَ اللَّهُ تَعَالَى فِيهَا أَحْبَبُ مِنْ أَمْرَةِ الْعَجْرِ نِجْصِ الْجَدِّ
 يُوسُفُ عَنْ نَفْسِهِ فَلَمَّا رَأَى فَمِصْرَهُ قَدْ مَرَّ بِرُفَّالِ تَهُ مِنْ كَيْدِهَا
 أَنْ كَبِدَتْ عَظِيمٌ وَقَالَ رَسُولُ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ لَعْنَةُ
 وَجْفِصَهُ جِبْنَ رَاجِعَةً فِي مَرِّهِ أَابَا بَكْرٍ رَضِيَ اللَّهُ عَنْهُ لَمَّا صَدَّقَ النَّاسَ
 عِنْدَ مَرِّهِ مَرَّ أَابَا بَكْرٍ فَلْيَصِلِ النَّاسُ فَاتَّخَذَ صَوْلَجًا يُوسُفُ
 هَذَا فِي الْعَجْرِ فَإِذَا كَانَ جِبْنَ النِّسَاءِ يَكْدُنُ حَبَابَ الرَّجَالِ
 وَأَكْبَرَهُمْ عَلَيْهَا وَجَبَّ الْحَدَّ مِنْ كَيْدِهَا قَالَ اللَّهُ تَعَالَى مَنْ

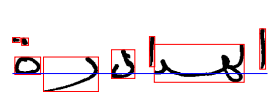
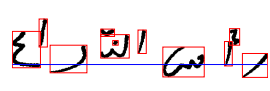
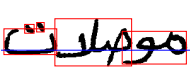
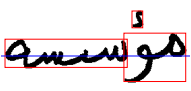
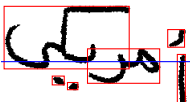
فَصَلِّ فِي الْحَجِّ مِنْ مَنِّ فِتْنَةِ النِّسَاءِ قَالَ اللَّهُ تَعَالَى
 رَزَقْنَا نِسَاءَ الْجَنَّةِ شَهْوَاتٍ مِنَ النِّسَاءِ وَالْبَنِينَ وَالْقَنَاطِيرَ الْمُقَنْطَرَةَ مِنَ
 الذَّهَبِ وَالْفِضَّةِ وَالْخَيْلَ الْمُسَوَّمَةَ مَهْرًا لِأَنْفُسِهِمْ وَأَلْحَبْتُمْ فِي ذَلِكَ
 مَعْدَاءَ نِسَاءٍ لَأَنْهَزْنَا كَثِيرَ الشَّهْوَاتِ الَّتِي تَهْوِيهَا النَّفْسُ فِي الْفُجْهِينِ
 عَنْ رَسُولِ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ أَنَّهُ قَالَ مَا تَرَكْتُ فِتْنَةً أَضْرَعُ عَلَى الرَّجَالِ
 مِنَ النِّسَاءِ وَقَالَ اللَّهُ تَعَالَى فِيهَا أَحْبَبُ مِنْ أَمْرَةِ الْعَجْرِ نِجْصِ الْجَدِّ
 يُوسُفُ عَنْ نَفْسِهِ فَلَمَّا رَأَى فَمِصْرَهُ قَدْ مَرَّ بِرُفَّالِ تَهُ مِنْ كَيْدِهَا
 أَنْ كَبِدَتْ عَظِيمٌ وَقَالَ رَسُولُ اللَّهِ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ لَعْنَةُ
 وَجْفِصَهُ جِبْنَ رَاجِعَةً فِي مَرِّهِ أَابَا بَكْرٍ رَضِيَ اللَّهُ عَنْهُ لَمَّا صَدَّقَ النَّاسَ
 عِنْدَ مَرِّهِ مَرَّ أَابَا بَكْرٍ فَلْيَصِلِ النَّاسَ فَاتَّخَذَ صَوْلَجًا يُوسُفُ
 هَذَا فِي الْعَجْرِ فَإِذَا كَانَ جِبْنَ النِّسَاءِ يَكْدُنُ حَبَابَ الرَّجَالِ
 وَأَكْبَرَهُمْ عَلَيْهَا وَجَبَّ الْحَدَّ مِنْ كَيْدِهَا قَالَ اللَّهُ تَعَالَى مَنْ

FIG. B.1. Results of text line segmentation approach.

Results prove effectiveness of the proposed approach even on manuscript documents with complex layouts. Text lines in the manuscript pages are skewed, touching, overlapping, contain gaps and different sized text.

B.2 Resolving sub-words overlapping

Table B.1. More results for the proposed sub-words overlaps resolving approach, samples are taken from IESK-arDB and IFN-ENIT databases.

			
			
			
			
			
			
			
			
			
			
IESK-arDB samples	Resolved versions	IFN-ENIT samples	Resolved versions

B.3 Words Segmentation

Two possible types of errors might occur when segmenting a handwritten word, namely under- and/or over- segmentation. the current section presents more segmentation results on samples drawn from IESK-arDB and IFN-ENIT databases. FIG. B.2 presents more zoomed-in correctly segmented results. FIG.B.3 and FIG. B.4 show results samples with over and under segmentation errors, respectively.



FIG. B.2. Correctly segmented words.



FIG. B.3. Samples results with under segmentation errors. Under segmentation errors happen when either a consecutive letter in a word is (fully) vertically overlapping the previous one, or when it extremely elongated to the right thereby vertically overlapping one or more previous letters.



FIG. B.4. Samples results with over segmentation errors. These types of errors are typical when CFPs occur inside the letter's main body.

Recognition Results

This appendix is dedicated to present additional detailed recognition results. FIG.C.1, FIG.C.2, and FIG.C.3 show recognition results obtained using CRFs and HCRFs classifiers on taxonomy Tax.1, Tax.3, and Tax.4 (results obtained using Tax.2 discussed in Chapter). TableC.1 and TableC.1 give the recognition results of each letter in each form, using HMMs, CRFs, and HCRFs.

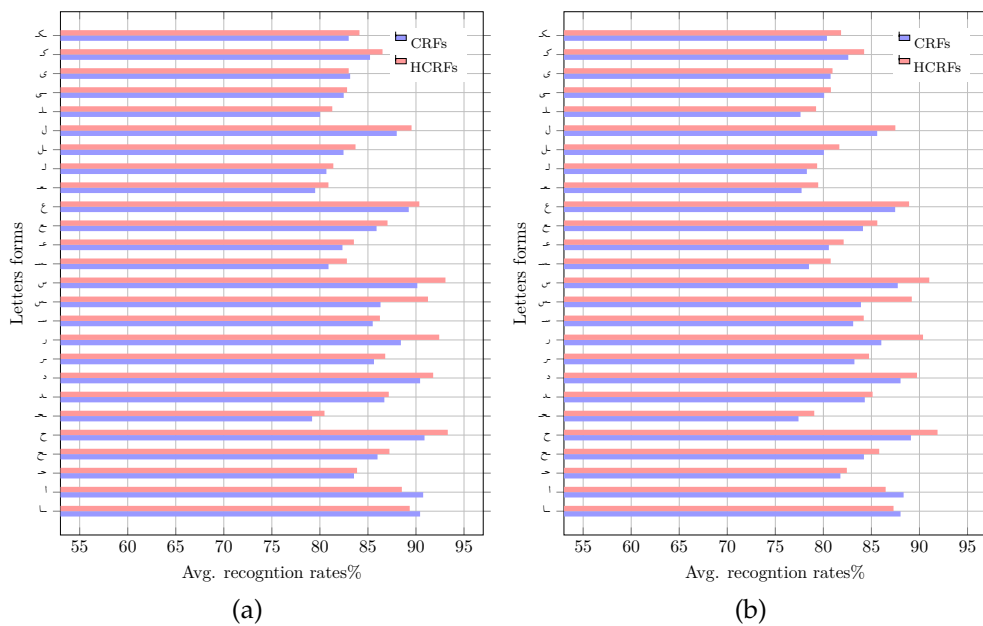


FIG. C.1. CRFs and HCRFs performance on taxonomy Tax.1: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database.

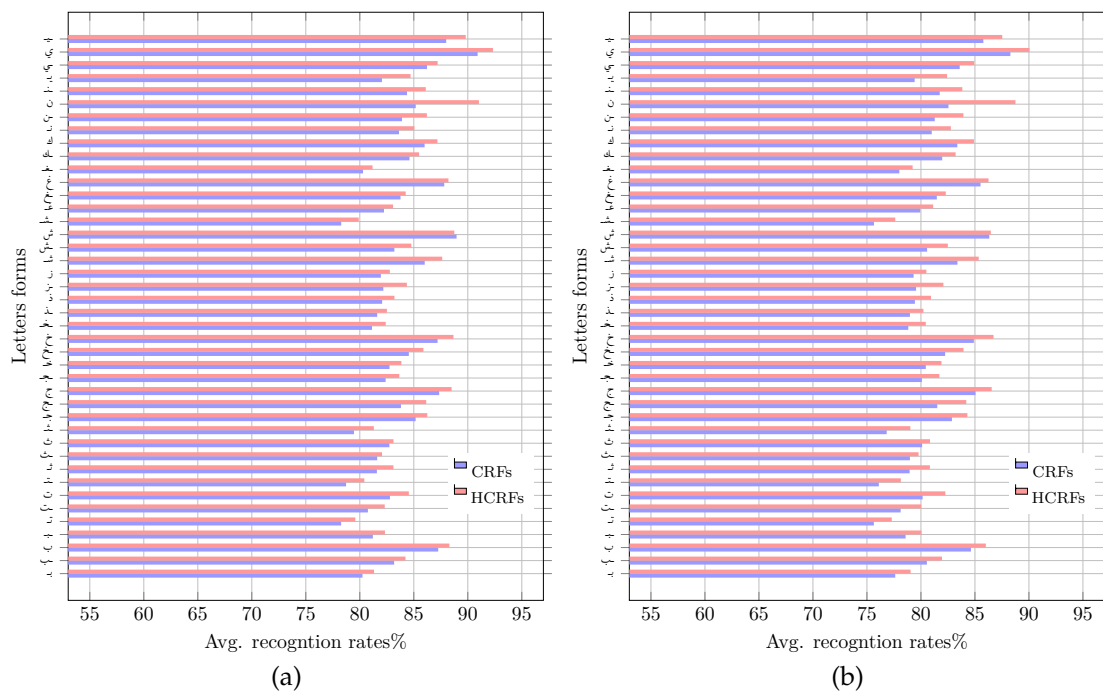


FIG. C.2. CRFs and HCRFs performance on taxonomy Tax.3: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database.

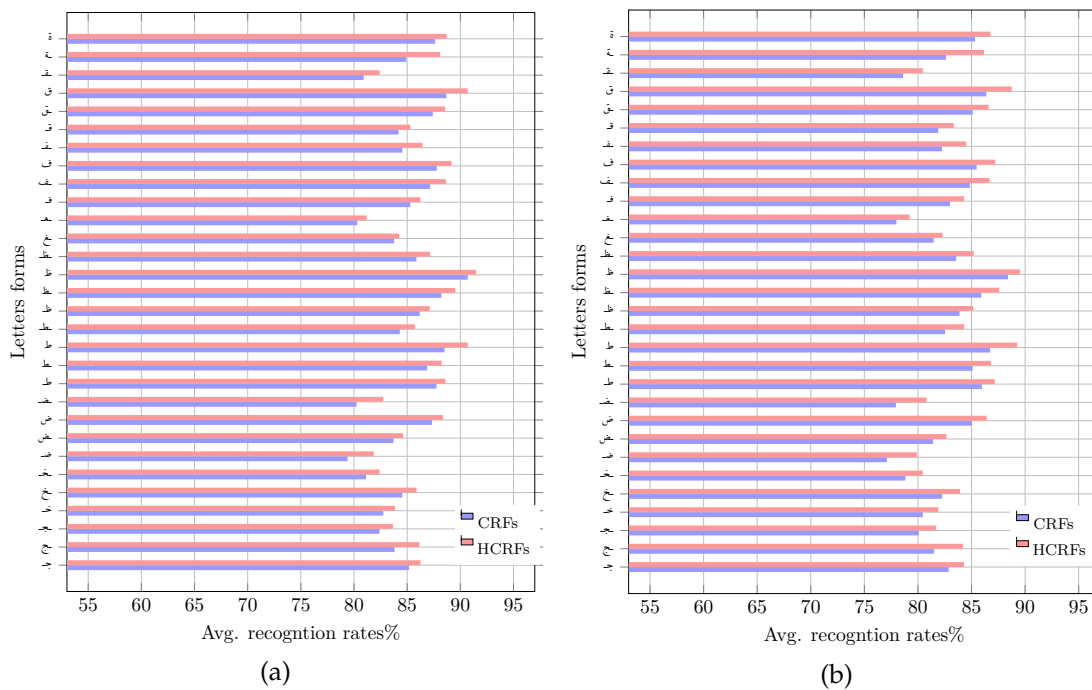


FIG. C.3. CRFs and HCRFs performance on taxonomy Tax.4: (a) performance on samples drawn from IESK-arDB database, and (b) performance on samples drawn from IFN-ENIT database.

Table C.1. Letter based recognition results on IESK-arDB database.

Letter	≈ HMMs Rec.%				≈ CRFs Rec.%				≈ HCRFs Rec.%				Avg.		
	B	E	I	M	Avg.	B	E	I	M	Avg.	B	E		I	M
ا	X	87.32	88.62	X	87.97	X	90.43	90.75	X	90.59	X	89.35	88.53	X	88.94
ب	73.80	77.10	82.91	78.05	77.97	80.25	83.18	87.26	81.21	82.98	81.32	84.23	88.29	82.33	84.04
ت	73.48	80.27	81.11	75.87	77.68	78.27	80.75	82.79	78.73	80.14	79.58	82.31	84.54	80.42	81.71
ث	78.09	79.13	80.21	77.00	78.61	81.58	81.61	82.74	79.46	81.35	83.11	82.06	83.12	81.3	82.40
ج	83.91	84.31	87	81.06	84.07	85.17	83.81	87.35	82.39	84.68	86.25	86.15	88.5	83.66	86.14
ح	78.55	80.52	84.84	83.02	81.73	83.55	86	90.89	79.19	84.91	83.87	87.24	93.31	80.48	86.23
خ	81.65	85.79	88.64	81.00	84.27	82.75	84.54	87.2	81.13	83.91	83.85	85.89	88.67	82.4	85.20
د	X	85.25	89.14	X	87.20	X	86.71	90.43	X	88.57	X	87.17	91.79	X	89.48
ذ	X	77.95	80.41	X	79.18	X	81.61	82.07	X	81.84	X	82.51	83.21	X	89.48
ر	X	84.9	88.12	X	86.51	X	85.63	88.43	X	87.03	X	86.8	92.42	X	89.61
ز	X	81.33	82.15	X	81.74	X	82.18	81.95	X	82.07	X	84.36	82.78	X	83.57
س	77.19	85.07	86.23	75.31	80.95	85.5	86.31	90.14	80.9	85.71	86.25	91.25	93.07	82.81	88.35
ش	81.4	79.74	88.51	81.36	82.75	86.01	83.2	88.96	78.28	84.11	87.63	84.77	88.75	79.9	85.26
ص	85.68	87.01	86.61	81.27	85.14	81.86	82.15	86.21	80.03	82.56	85.5	87.31	87.19	81.58	85.40
ض	81.75	84.33	86	82.06	83.54	79.4	83.71	87.34	80.24	82.67	81.85	84.61	88.37	82.75	84.40
ط	81.71	80.95	87.83	81.07	82.89	87.75	86.88	88.52	84.31	86.87	88.6	88.24	90.69	85.74	88.32
ظ	86.59	85.45	86.51	80.96	84.88	86.18	88.21	90.7	85.87	87.74	87.12	89.53	91.49	87.15	88.82
ع	74.88	88.6	88.87	73.37	81.43	82.35	85.9	89.25	79.51	84.25	83.54	87.04	90.35	80.89	85.46
غ	72.47	84.5	87.13	70.91	78.75	82.24	83.77	87.82	80.30	83.53	83.08	84.25	88.21	81.18	84.18
ف	82.25	77.56	83.17	82.06	81.26	85.29	87.15	87.78	84.54	86.19	86.24	88.63	89.16	86.44	87.62
ق	76.81	83.52	85.20	73.67	79.80	84.19	87.41	88.68	80.91	85.30	85.30	88.56	90.70	82.41	86.74
ك	87.32	81.08	85.84	87.11	85.34	85.22	84.60	86.00	83.00	84.71	86.52	85.49	87.19	84.12	85.83
ل	80.60	82.10	87.72	78.51	82.23	80.68	82.46	88.00	80.02	82.79	81.40	83.71	89.54	81.29	83.99
م	84.03	83.26	82.21	82.77	83.07	84.41	82.57	87.00	84.71	84.67	85.48	83.83	88.24	86.50	86.01
ن	80.14	81.40	83.31	72.02	79.22	83.63	83.91	85.18	84.37	84.27	85.05	86.21	91.03	86.11	87.10
ه	83.27	87.37	89.56	81.74	85.49	85.87	91.23	92.13	84.79	88.51	89.17	92.66	93.58	86.36	90.44
و	X	87.49	87.33	X	87.41	X	87.86	89.40	X	88.63	X	88.61	90.30	X	90.30
ي	73.83	83.66	85.71	76.25	79.86	82.05	86.22	90.91	88.41	86.90	84.70	87.21	92.33	89.81	88.51
ى	X	78.08	78.36	X	78.22	X	82.48	83.15	X	82.82	X	82.83	83	X	82.92
ة	X	84.20	85.00	X	84.60	X	84.92	87.63	X	86.28	X	88.11	88.71	X	88.41
Avg.%	79.97	82.97	85.48	78.93	82.46	83.37	84.91	87.56	81.92	84.89	84.79	86.29	88.90	83.44	86.27

Table C.2. Letter based recognition results on IFN-ENIT database.

Letter	≈ HMMs Rec.%						≈ CRFs Rec.%						≈ HCRFs Rec.%					
	B	E	I	M	Avg.		B	E	I	M	Avg.		B	E	I	M	Avg.	
ا	X	76.98	78.28	X	77.63		X	88.03	88.35	X	88.19		X	87.3	86.48	X	86.89	
ب	63.23	66.53	72.34	67.48	67.40		77.62	80.55	84.63	78.58	80.35		79.04	81.95	86.01	80.05	81.76	
ت	62.91	69.70	70.54	65.30	67.11		75.64	78.12	80.16	76.10	77.51		77.30	80.03	82.26	78.14	79.43	
ث	67.52	68.56	69.64	66.43	68.04		78.95	78.98	80.11	76.83	78.72		80.83	79.78	80.84	79.02	80.12	
ج	73.67	74.07	76.76	70.82	73.83		82.87	81.51	85.05	80.09	82.38		84.30	84.20	86.55	81.71	84.19	
ح	68.83	70.80	75.12	73.30	72.01		81.77	84.22	89.11	77.41	83.13		82.44	85.81	91.88	79.05	84.80	
خ	71.41	75.55	78.40	70.76	74.03		80.45	82.24	84.90	78.83	81.61		81.90	83.94	86.72	80.45	83.25	
د	X	74.91	78.80	X	76.86		X	84.31	88.03	X	86.17		X	85.12	89.74	X	87.43	
ذ	X	67.38	69.84	X	68.61		X	78.98	79.44	X	79.21		X	80.23	80.93	X	80.58	
ر	X	74.56	77.78	X	76.17		X	83.23	86.03	X	84.63		X	84.75	90.37	X	87.56	
ز	X	70.76	71.58	X	71.17		X	79.55	79.32	X	79.435		X	82.08	80.50	X	81.29	
س	66.85	74.73	75.89	64.97	70.61		83.10	83.91	87.74	78.50	83.31		84.20	89.20	91.02	80.76	86.30	
ش	70.83	69.17	77.94	70.79	72.19		83.38	80.57	86.33	75.65	81.48		85.35	82.49	86.47	77.62	82.98	
ص	75.96	77.29	76.89	71.55	75.42		80.08	80.37	84.43	78.25	80.78		84.07	85.88	85.76	80.15	83.97	
ض	71.51	74.09	75.76	71.82	73.30		77.10	81.41	85.04	77.94	80.37		79.90	82.66	86.42	80.80	82.45	
ط	72.00	71.23	78.11	71.35	73.17		85.97	85.10	86.74	82.53	85.10		87.17	86.81	89.26	84.31	86.89	
ظ	76.35	75.21	76.27	70.72	74.64		83.88	85.91	88.40	83.57	85.44		85.17	87.58	89.54	85.20	86.87	
ع	65.16	78.88	79.15	63.65	71.71		80.57	84.12	87.47	77.73	82.47		82.11	85.61	88.92	79.46	84.03	
غ	62.23	74.26	76.89	60.67	68.51		79.94	81.47	85.52	78	81.23		81.13	82.30	86.26	79.23	82.23	
ف	72.01	67.32	72.93	71.82	71.02		83.00	84.85	85.48	82.24	83.89		84.29	86.68	87.21	84.49	85.67	
ق	66.57	73.28	74.96	63.43	69.56		81.89	85.11	86.38	78.61	83.00		83.35	86.61	88.75	80.46	84.79	
ك	76.75	70.51	75.27	76.54	74.77		82.59	81.97	83.37	80.37	82.08		84.24	83.21	84.91	81.84	83.55	
ل	70.26	71.76	77.38	68.17	71.89		78.28	80.06	85.60	77.62	80.39		79.35	81.66	87.49	79.24	81.95	
م	74.31	73.54	72.49	73.05	73.35		82.63	80.79	85.22	82.93	82.89		84.05	82.40	86.81	85.07	84.58	
ن	69.57	70.83	72.74	61.45	68.65		81.00	81.28	82.55	81.74	81.64		82.77	83.93	88.75	83.83	84.82	
ه	73.55	77.65	79.84	72.02	75.77		84.09	89.45	90.35	83.01	86.73		87.74	91.23	92.15	84.93	89.01	
و	X	77.77	77.61	X	77.69		X	86.08	87.62	X	86.85		X	88.61	88.87	X	88.87	
ي	63.26	73.09	75.14	65.68	69.29		79.42	83.59	88.28	85.78	84.27		82.42	84.93	90.05	87.53	86.23	
ى	X	67.74	68.02	X	67.88		X	80.08	80.75	X	80.415		X	80.78	80.95	X	80.865	
ة	X	73.96	74.76	X	74.36		X	82.62	85.33	X	83.975		X	86.16	86.76	X	86.46	
Avg.%	69.76	72.74	75.24	68.72	72.22		81.10	82.62	85.26	79.65	82.59		82.87	84.32	86.95	81.52	84.33	

Concise Curriculum Vitae

Name:	Moftah Elzobi
Citizenship:	Libyan
Marital Status:	Married
Born in:	Cairo
Mail:	P.O. Box 4120, 39016 Magdeburg, Germany
E-mail:	moftah.elzobi@ovgu.de

Education

2009 – Present	Pursuing a PhD degree, IKT, Otto-von-Guericke University, Magdeburg, Germany.
2002 – 2005	Master of Computer Science, Otto-von-Guericke University Magdeburg, Germany.
1990 – 1995	Bachelor in Computer Science, Sirte University, Sirte, Libya.

Professional Experience

2005 – 2008	Research associate, Sirte University, Sirte, Libya.
1999 – 2001	Computer programming trainer, AFAQ Tech, Sirte, Libya.
1996 – 1999	Database programmer and Network Administrator, Tibisti Hotels Company, Sirte, Libya.

Magdeburg, 21.04.2017

Moftah Elzobi

Related Publications

Most of the material contained in this doctoral dissertation is partly based on the following collection of refereed papers published in a variety of peer-reviewed journals and/or proceedings of well reputed international conferences/symposia.

- 1) M. Elzobi, A. Al-Hamadi, Z. Aghbari, and L. Dinges, "IESK-ArDB: A Database for Handwritten Arabic and An Optimized Topological Segmentation Approach", *International Journal on Document Analysis and Recognition (IJDAR)*, pp. 295–308, 2012.
- 2) M. Elzobi, A. Al-Hamadi, L. Dinges, B. Michaelis, "A Structural Feature based Segmentation for Off-line Handwritten Arabic Text," in *5th International Symposium on Image/Video Communication over Fixed and Mobile Networks, ISIVC 2010*, Rabat, Morocco, October 2010, pp. 519–522.
- 3) M. Elzobi, A. Al-Hamadi, Z. Aghbari, and L. Dinges, "Off-line Handwritten Arabic Words Segmentation based on Structural Features and Connected Components Analysis," in *International Conference on Computer Graphics, Visualization and Computer vision, WSCG 2011*, Plzen, Czech Republic, February 2011, pp. 135–142.

- 4) M. Elzobi, A. Al-Hamadi, A. Saeed, and L. Dinges, "Arabic Handwriting Recognition Using Gabor Wavelet Transform and SVM," in *11th IEEE International Conference on Signal Processing (ICSP 2012)*, Beijing, China, October 2012, vol.3, pp. 2164–2158.
- 5) M. Elzobi, A. Al-Hamadi, and L. Dinges, "A Hidden Markov Model-based Approach with an Adaptive Threshold Model for Off-line Arabic Handwriting Recognition," in *International Conference on Document Analysis and Recognition (ICDAR 2013)*, Washington, DC, USA, August 2013, pp. 945–949.
- 6) M. Elzobi, A. Al-Hamadi, Z. Alaghbari, L. Dinges, and A. Saeed, "Gabor Wavelet Recognition Approach for Off-Line Handwritten Arabic Using Explicit Segmentation," in *5th International Conference on Image Processing and Communications (IPC 2013)*, Bydgoszcz, Poland, September 2013, *Advances in Intelligent Systems and Computing*, 2014, pp. 245–254, Springer-Verlag Berlin/Heidelberg.
- 7) M. Elzobi, A. Al-Hamadi, L. Dinges, and S. El-etriby "CRFs and HCRFs Based Recognition for Off-Line Arabic Handwriting," in *Advances in Visual Computing: The 11th International Symposium (ISVC 2015)*, Las Vegas, NV, USA, December 2015, *Proceedings Part II*, pp. 337–346.
- 8) L. Dinges, A. Al-Hamadi, M. Elzobi, Z. Aghbari, and H. Mustafa, "Offline Automatic Segmentation based Recognition of Handwritten Arabic Words", *International Journal of Signal Processing, Image Processing and Pattern Recognition (IJSIP)*, pp. 131–144, 2011.
- 9) L. Dinges, A. Al-Hamadi, M. Elzobi, S. El-etriby, and A. Ghoneim, "ASM based Synthesis of Handwritten Arabic Text Pages", *The Scientific World Journal, Hindawi Publishing Corp.*, vol.2015, Article ID 323575, 2015.
- 10) L. Dinges, A. Al-Hamadi, M. Elzobi, and S. El-etriby, "Synthesis of Common

- Arabic Handwritings to Aid Optical Character Recognition Research”, *Sensors*, vol.16, pp. 131–144, 2016.
- 11) L. Dinges, A. Al-Hamadi, M. Elzobi, and Z. Alaghbari, “Synthizing Handwritten Arabic Text Using Active Shape Models,” in *Image Processing and Communications Challenges 3 (IPC 2011)*, Bydgoszcz, Poland, September 2011, Springer-Verlag pp. 401-408.
 - 12) L. Dinges, A. Al-Hamadi, and M. Elzobi, “An Active Shape Model based Approach for Arabic Handwritten Character Recognition,” in *11th IEEE International Conference on Signal Processing (ICSP 2012)*, Beijing, China, October 2012, vol.3, pp. 1194–1198.
 - 13) L. Dinges, A. Al-Hamadi, and , M. Elzobi “A Locale Group Based Line Segmentation Approach for Non Uniform Skewed and Curved Arabic Handwritings,” in *International Conference on Document Analysis and Recognition (ICDAR 2013)*, Washington, DC, USA, August 2013, pp. 803–806.
 - 14) L. Dinges, A. Al-Hamadi, and , M. Elzobi “An Approach for Arabic Handwriting Synthesis based on Active Shape Models,” in *International Conference on Document Analysis and Recognition (ICDAR 2013)*, Washington, DC, USA, August 2013, pp. 1292–1296.
 - 15) L. Dinges, A. Al-Hamadi, and , M. Elzobi “Synthetic based Validation of Segmentation of Handwritten Arabic Words,” in *Conference of Natural Sciences and Technology in Manuscript Analysis*, Hamburg, Germany, December 2013, Manuscript cultures 2014, pp. 10–18.
 - 16) A. Saeed, A. Al-Hamadi, R. Niese, and M. Elzobi, “Frame-Based Facial Expression Recognition Using Geometrical Features”, *Adv. in Hum.-Comp. Int., Hindawi Publishing Corp.*, vol.2014, pp. 4-17, 2014.

-
- 17) A. Saeed, A. Al-Hamadi, R. Niese and M. Elzobi, "Effective geometric features for human emotion recognition," in *11th IEEE International Conference on Signal Processing (ICSP 2012)*, Beijing, China, October 2012, vol.3, pp. 623–627.