

Arabic Handwriting Recognition Using Deep Learning Neural Networks

BY

Mohamed M. O. Eltay

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

Electrical Engineering

May 2020

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Mohamed Muawia Mohamed Omer Eltay** under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN ELECTRICAL ENGINEERING**.



Dr. Abdelmalek Zidouri
(Advisor)



Dr. Abdallah S. Al-Ahmari
Department Chairman



Dr. Suliman S. Al-Homidan
Dean of Graduate Studies



14-05-2020

Date



Dr. Irfan Ahmad
(Member)



Dr. Azzedine Zerguine
(Member)

© Mohamed Eltay

2020

I dedicate this work to my mother, who inspired me all the way and who motivated me to do whatever I could to finish what I started. I devote this to my beloved father, who meant and still means so much to me. Though he's no longer of this world, his memories still influence my life. |

ACKNOWLEDGMENTS

First of all, I would like to thank Dr. Abdelmalek Zidouri of the Department of Electrical Engineering at King Fahd University of Petroleum & Minerals (KFUPM). When I came to a problem spot or had a concern regarding my work or writing, the door to his office was always open. He guided me right anytime he felt I wanted it.

I would also like to acknowledge with much appreciation the crucial role of Dr. Irfan Ahmad for providing me with unfailing support and continuous encouragement through the process of researching and writing this thesis. This accomplishment would not have been possible without him. Thank you.

I would also like to thank Dr. Azzedine Zerguine, who has participated in this research project validation. The validation survey could not have been carried out effectively without his active support and feedback.

Finally, I would like to express my heartfelt gratitude to all my KFUPM colleagues for the constant support and motivation I have provided during my years of study and the process of research and writing of this thesis. Without them, this success would not have been possible. Thank you. |

TABLE OF CONTENTS

ACKNOWLEDGMENTS	V
TABLE OF CONTENTS	VI
LIST OF TABLES	IX
LIST OF FIGURES	X
LIST OF ABBREVIATIONS	XI
ABSTRACT	XIII
ملخص الرسالة	XIV
CHAPTER 1 INTRODUCTION	1
1.1 The motivation for the current work	2
1.2 Problem Statement	3
1.3 Research Objectives	4
1.4 Background and Pre-analysis of Arabic Scripts	4
1.5 Contributions of the thesis	8
1.6 Thesis Structure	8
CHAPTER 2 LITERATURE REVIEW	9
2.1 Pattern Recognition (PR)	9
2.2 Optical Character Recognition (OCR)	10
2.2.1 OCR's History	11
2.2.2 OCR Methods	13
2.2.3 OCR Applications	14
2.3 Handwriting Recognition Systems	14

2.3.1	Pre-processing	17
2.3.2	Features Extraction.....	17
2.3.3	Segmentation	19
2.3.4	Classification	19
2.4	Literature Review on Arabic Handwriting Recognition	20
2.4.1	A Review on Handwritten Arabic Databases	20
2.4.2	Related Works on Arabic Handwriting Recognition	23
CHAPTER 3 METHODOLOGY		27
3.1	Modeling Options for Arabic Text Recognition.....	27
3.1.1	Whole-Word Modeling (Holistic-Handwriting-Recognition).....	27
3.1.2	Using Characters as modeling units	28
3.1.3	Using Character shapes as modeling units	28
3.1.4	Modeling core shapes and diacritics separately	28
3.2	Deep Learning Approaches for Handwriting Recognition	30
3.2.1	Convolutional Neural Networks (CNN)	31
3.2.2	Recurrent Neural Networks (RNNs)	36
3.2.3	Long Short-Term Memory (LSTM)	39
3.2.4	Connectionist Temporal Classification (CTC) Layer	41
3.2.5	Token Passing Decoder.....	42
3.2.6	Word Beam Search (WBS) Decoder.....	43
3.2.7	Bi-Directional Long Short-Term Memory (BLSTM)	45
3.2.8	Multi-stage Deep Neural Network (MDNN)	46
CHAPTER 4 ADAPTIVE DATA AUGMENTATION ALGORITHM		48
4.1	The Problem of Imbalanced Data Classification.....	48
4.2	Adaptive Data Augmentation Algorithm.....	50

CHAPTER 5 EXPERIMENTS AND RESULTS.....	52
5.1 The IFN/ENIT Database.....	52
5.2 The AHDB Database.....	53
5.3 Preprocessing.....	53
5.4 Experiment Configurations.....	55
5.5 Performance Evaluation.....	56
5.6 Results of Holistic Handwriting Recognition.....	58
5.7 Results of Arabic Handwriting Recognition Using Characters Shapes as Models.....	59
5.8 Results of Arabic Handwriting Recognition Using Characters as Models.....	61
5.9 Results of Arabic Handwriting Recognition Using a Multi-stage Recognition System.....	62
5.10 Results of Using Adaptive Data Augmentation Technique	63
5.11 Comparison with Other State-of-the-Art Systems.	68
CONCLUSION	71
REFERENCES.....	73
VITAE.....	83

LIST OF TABLES

Table 1. Arabic Characters and their Position Dependent Shapes.....	5
Table 2. A Brief Time Frame for Optical Character Reconition Researches.	12
Table 3. The Databases Available for Arabic Character iRecognition.....	22
Table 4. Pseudo-Code for A Basic Word Beam Searching [94].....	44
Table 5. Pseudo-Code for Adaptive Data Augmentation Technique.	51
Table 6. AlexNet vs. RNN (Holistic Recognition).	58
Table 7. Accuracy Rates of the Systems using character shapes as models (LSTM)	59
Table 8. Accuracy Rates of the Systems using characters' shapes as models (LSTM)....	60
Table 9. Accuracy Rates of the Systems using characters' shapes as models (BLSTM). 60	
Table 10. Accuracy Rates of the Systems using characters as models (BLSTM).....	61
Table 11. Accuracy Rates of the Systems using characters as models (BLSTM).....	61
Table 12. Accuracy Rates of the Systems using Multi-stage recognition system	63
Table 13. IFN/ENIT database [57] statistics.....	64
Table 14. AHDB [58] statistics.....	65
Table 15. ALEXNET VS. RNN (HOLISTIC RECOGNITION)	67
Table 16. Accuracy Rates of the Systems using characters as models (BLSTM).....	68
Table 17. Accuracy Rates of the Systems using character shapes as models (BLSTM)..	68
Table 18. Comparison with Other State-of-the-Art Systems.....	69

LIST OF FIGURES

Figure 1. Utilization of dots in some Arabic letters with dots.	6
Figure 2. Different diacritics for Arabic characters.	6
Figure 3. (a) Laam-Alif, (b) Laam-Meem, and (c) Laam-Heh combinations.....	6
Figure 4. Computer printed and handwritten versions of the same Arabic text.	7
Figure 5. Challenges in handwriting Arabic recognition.....	7
Figure 6. Classification of character recognition systems.	15
Figure 7. General steps for handwritten text recognition.	16
Figure 8. Some character shapes to core shapes mapping for Arabic script.....	29
Figure 9. Words representation using core-shape.....	29
Figure 10. A rule-based algorithm for separating core-shapes and diacritics	30
Figure 11. LeNet 5 [77].	32
Figure 12. Illustration of a single convolutional layer that produces k -feature maps	33
Figure 13. Illustration of pooling layer that produces the same k -feature maps.....	34
Figure 14. AlexNet architecture [86].	35
Figure 15. A folded recurrent neural network [89].	37
Figure 16. Unfolded recurrent neural network through time [89].	38
Figure 17. RNN cell unit vs. LSTM cell unit [91,92].	40
Figure 18. CTC output layer connected to the LSTM network.	42
Figure 19. The architecture of a bi-directional recurrent neural network.	46
Figure 20. An illustration of the multi-stage word recognition isystem.	46
Figure 21. Samples from the IFN/ENIT database.	52
Figure 22. Samples from the AHDB database.....	53
Figure 23. A word taken from the IFN/ENIT database rotated at different angles	54
Figure 24. Shearing a word in the database with different shearing factor.	54
Figure 25. Separating core-shapes and diacritics.....	62
Figure 26. Adaptive vs. standard data augmentation (IFN/ENIT Database).	65
Figure 27. Adaptive vs. standard data augmentation (AHDB Database)	67

LIST OF ABBREVIATIONS

AI	:	Artificial Intelligence
AHR	:	Arabic Handwriting Recognition
ANN	:	Artificial Neural Network
BLSTM	:	Bidirectional Long Short-Term Memory
CAR	:	Character Accuracy Rate
CNN	:	Convolutional Neural Network
CTC	:	Connectionist Temporal Classification
CV	:	Computer Vision
DLNN	:	Deep Learning Neural Network
GT	:	Ground Truth
HMM	:	Hidden Markov Model
LSTM	:	Long Short-Term Memory
MDRNN	:	Multi-Dimensional Recurrent Neural Networks
OCR	:	Optical Character Recognition
PAW	:	Piece of Arabic Word
PR	:	Pattern Recognition

ReLU	:	Rectified Linear Unit
RNN	:	Recurrent Neural Networks
SVM	:	Support Vector Machine
WAR	:	Word Accuracy Rate
WBS	:	Word Beam Search

|

ABSTRACT

Full Name : Mohamed Muawia Mohamed Omer Eltay
Thesis Title : Arabic Handwriting Recognition Using Deep Learning Neural Networks
Major Field : Electrical Engineering
Date of Degree : May 2020

Recognition of cursive handwritten Arabic text is a difficult problem because of context-sensitive character shapes, the non-uniform spacing between words and within a word, diverse placements of dots, and diacritics, and very low inter-class variation among individual classes. In this paper, we review and investigate different deep learning architectures and modeling choices for Arabic handwriting recognition. Further, we address the problem that imbalanced data sets present to deep learning systems. In order to address this issue, we are presenting a novel adaptive data-augmentation algorithm to promote class diversity. This algorithm assigns a weight to each word in the database lexicon. This weight is calculated based on the average probability of each class in a word. Experimental results on the IFN/ENIT and AHDB databases have shown that our presented approach yields state-of-the-art results |

|

ملخص الرسالة

الاسم الكامل: محمد معاوية محمد عمر التاي

عنوان الرسالة: التعرف على الخطوط العربية باستخدام الشبكات العصبية العميقة

التخصص: هندسة كهربائية

تاريخ الدرجة العلمية: مايو 2020

إن التعرف على النصوص المكتوبة باليد باستخدام الحاسوب من المسائل التي نواجه فيها تحديات عظيمة ، ولا سيما التعرف على النصوص العربية المكتوبة باليد. وذلك لأن النص العربي عادةً ما يكتب من اليمين إلى اليسار كما أن الحروف العربية تكون متشابكة مما يجعل عملية التعرف على الحروف مسألة صعبة تتطلب في بعض الأحيان القيام بتجزئة النص إلى أجزاء صغيرة ثم التعرف على كل جزء منفرداً. نظراً لقلة الأبحاث التي تعنى باللغة العربية فإننا في هذا البحث سوف نقوم ببناء نظام تعرف على النصوص العربية باستخدام الشبكات العصبية العميقة. بسبب التطور المذهل في قدرات المعالجة الحاسوبية والتطور السريع لأنظمة الشبكات العصبية فقد أثبتت الشبكات العصبية خلال العقد الماضي أنها وسيلة فعالة للتعرف على النصوص المكتوبة باليد، خاصةً النصوص المتشابكة مثل اللغة العربية. إننا نهدف في هذا البحث في بناء نظام فعال وسريع قادر على التعرف على النصوص العربية المكتوبة باليد، والقيام بعدد من التجارب التي تثبت فعالية هذا النظام. لقد قمنا أيضاً في هذا البحث بتقديم خوارزمية جديدة للتعامل مع مشكلة عدم وجود عناصر كافية لتدريب الشبكة العصبية. لقد أثبتت هذه الخوارزمية بأنها قادرة على رفع كفاءة التعرف على النصوص العربية المكتوبة. لقد استخدمنا في هذا البحث قاعدة البيانات المشهورة IFN/ENIT و قاعدة البيانات AHDB وذلك لإثبات أن كفاءة النظام الجديد الذي نقترحه مع استخدام الخوارزمية أفضل من بعض الأنظمة التي نشرها حديثاً |

CHAPTER 1

INTRODUCTION

Over the last ten years, there has been a significant advance in machine learning algorithms. Indeed, the increasing power of computer data processing has supported the analytical capabilities of handwriting recognition systems. Recently, deep learning methods employed in document analysis have enjoyed tremendous success. These methods are robust to deformations, scaling, and rotation [1]. Thus, they are best suited for text recognition. Mainly, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been extensively used for text recognition.

Optical Character Recognition (OCR) is an old field of Pattern Recognition (PR). The human reading process is the inspiration behind the development of a machine capable of reading texts with the same expertise as people. The very first OCR systems were designed to help blind people to read [1]. Decades of scientific research have produced many practical systems ranging from the sorting of posts in post offices [1] to automatic recognition of bank checks [2], and advanced systems that read the text in natural scenes [3].

The main goal of the handwriting recognition system is to convert handwritten text documents from digital image format to encoded character format documents that are readable and editable using word processing application systems. Recognition of handwriting is becoming more critical as it helps to accomplish the office tasks efficiently and has solved many problems that lead to a reduction in time and effort.

1.1 The motivation for the current work

More than 400 million people around the world speak Arabic [4]. Besides, Arabic letters are used in several other languages, such as Persian, Urdu, and Jawi. Compared to other languages, Arabic has not yet received much research and study. Handwritten Arabic text recognition can be achieved using either holistic or segmentation approaches.

The holistic approach generally processes words into smaller components without segmentation. However, in the segmentation approach, we divide the words into characters or strokes and then pass the segments to the identifier [5,6,7]. The development of practical cursive script segmentation algorithms is difficult and requires extensive expertise. However, algorithms that assume the text is already segmented, generally lead to significantly higher recognition accuracy [6]. Recently, holistic approaches have been more successful in a limited vocabulary handwriting database [8,9,10].

Several studies and researches have shown that recurrent neural networks (RNNs) are superior to Hidden Markov Models (HMMs) for sequence labeling tasks such as speech and online handwriting recognition [11,12,13,14]. One possible reason for this is that RNNs are trained in a discriminative manner, whereas HMMs are generative. Discriminative methods tend to give better results in pattern recognition tasks when a large set of data is available [15,16].

In general, Hidden Markov Models (HMMs) are much simpler than Recurrent Neural Networks (RNNs). The Markov chain assumes the Markov property; it is "memoryless." The probability of the next symbol is calculated based on the k previous

symbols. In practice, k is limited to low values because the transition matrix grows exponentially. Thus, sentences generated by a Hidden Markov Model are very inconsistent. An RNN may perform better if we have a very large dataset since the extra complexity can take better advantage of the information in the given data (e.g., with LSTM units) not bound by the Markov property. Their prosperous internal state allows them to keep track of long-distant dependencies.

1.2 Problem Statement

In the last decades, many character recognition systems have been developed and used for commercial purposes. Besides, several applications, including mail sorting, bank processing, document reading, postal address recognition, and bank checking process, require a proper and accurate offline handwriting recognition system. However, recognition of handwritten Arabic text is considered a difficult task compared to the Latin script due to its cursive nature, presence of diacritics, diagonal strokes, change in shape of characters according to their location in the word, inter-word, and intra-word spaces, and difficulty in segmentation.

1.3 Research Objectives

This thesis aims to develop an efficient handwritten text recognition system based on using Deep Learning Neural Networks (DLNN). Generally, the main objectives of this research are:

- To build an efficient deep network handwriting recognition system by selecting the appropriate algorithms for processes in the preprocessing stage and other stages in the framework.
- To develop an algorithm based on using Deep Learning Neural Networks to recognize handwritten Arabic scripts.
- To outperform other classical approaches and state of the art models.
- To explore different modeling options for the Arabic script.
- To Design a novel data augmentation algorithm that can be used for any handwritten database.

1.4 Background and Pre-analysis of Arabic Scripts

Arabic is an ancient Semitic language. It is the lingua franca of the Arab world. Arabic is ranked among the top six most spoken languages in the world [17]. There are several other languages, including Persian and Urdu, which use Arabic scripts to write; however, they have a few characters other than Arabic to cover non-Arabic phones.

Most of the Arabic characters are connected while they are written. Arabic has 28 characters that are written from right to left. Each character may have two or four distinct shapes depending on its position within the word. Table.1 the Arabic alphabets and the shapes for each letter based on its Position.

Table 1. Arabic Characters and their Position Dependent Shapes.

Name	Isolated	Final	Medial	Initial
'Alif	ا	آ		
Bā'	ب	ب	ب	ب
Tā'	ت	ت	ت	ت
<u>Thā'</u>	ث	ث	ث	ث
Jīm	ج	ج	ج	ج
Ḥā'	ح	ح	ح	ح
<u>Khā'</u>	خ	خ	خ	خ
Dāl	د	د		
<u>Dhāl</u>	ذ	ذ		
Rā'	ر	ر		
Zāy	ز	ز		
Sīn	س	س	س	س
<u>Shīn</u>	ش	ش	ش	ش
Ṣād	ص	ص	ص	ص
Ḍād	ض	ض	ض	ض
Ṭā'	ط	ط	ط	ط
Ẓā'	ظ	ظ	ظ	ظ
Ayn	ع	ع	ع	ع
<u>Ghayn</u>	غ	غ	غ	غ
Fā'	ف	ف	ف	ف
Qāf	ق	ق	ق	ق
Kāf	ك	ك	ك	ك
Lām	ل	ل	ل	ل
Mīm	م	م	م	م
Nūn	ن	ن	ن	ن
Hā'	ه	ه	ه	ه
Wāw	و	و		
Yā'	ي	ي	ي	ي
Hamza	ء			

Dots play a significant role in Arabic characters. Some of the shapes of the characters are very similar, but the distinction is made between the characters by the position and the number of dots that can be either above or below the character body. For instance, the three characters (Thaa ث) (Taa ت) and (Baa ب) have the same main body shape and differ only by the dots and their positions. Fig.1 shows Arabic characters with different position and number of dots.

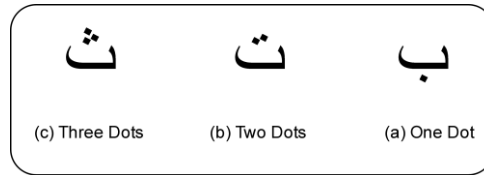


Figure 1. Utilization of dots in some Arabic letters with dots.

In addition, short marks, e.g., (Hamza ء), may be placed above or below five characters or may appear as isolated characters. Besides, Arabic utilizes diacritical imprints (vocalized content) to control the articulation of words, Fig.2 shows different types of diacritics used in Arabic scripts. However, these are seldom showing up in handwritten documents. They mostly used in formal documents and where the context is ambiguous.

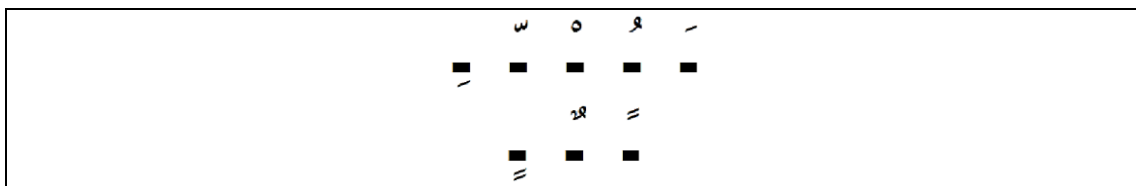


Figure 2. Different diacritics for Arabic characters.

Moreover, when two or more letters are joined as a single glyph, a ligature occurs. The use of ligature in Arabic is common. For example, (a) "Laam-Alif," (b) "Laam-Meem," and (c) "Laam-Heh,". Figure 3 shows cases of associated letters.

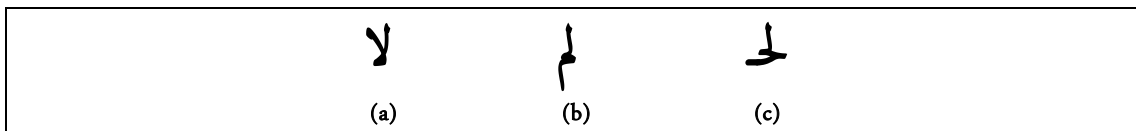


Figure 3. (a) Laam-Alif, (b) Laam-Meem, and (c) Laam-Heh combinations.

Unlike Latin content languages, handwritten and machine-printed Arabic scripts are extremely comparative. While letter shapes have minor varieties because of style in handwritten content, the manner in which letters are associated is dependably

the same. Fig.4 demonstrates a case of similar content with a machine-printed form and a handwritten variant.

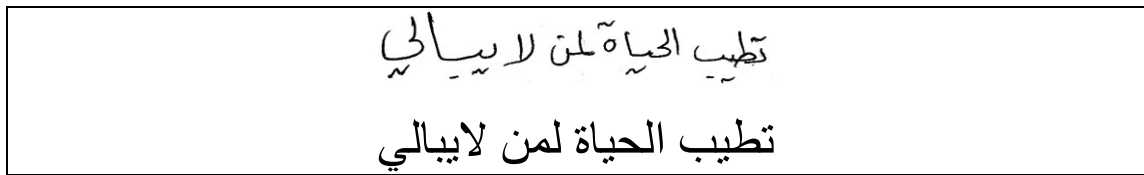


Figure 4. Computer printed and handwritten versions of the same Arabic text.

In general, some of the challenging structural characteristics of Arabic handwriting are described below:

1. Arabic word comprises at least one associated segment (sub-word), and each one contains at least one character that can be covered with different characters or diacritics. Besides, different characters can be consolidated vertically to shape a ligature. (Fig.5a)
2. Some words contain touching, interlaced, or broken characters. (Fig.5b)
3. Each writer has an individual handwriting style. (Fig.5c)
4. Some Arabic characters have diacritics (a diacritic might be set above or beneath the body of the character). These diacritics can also be overlapping. (Fig.5d)

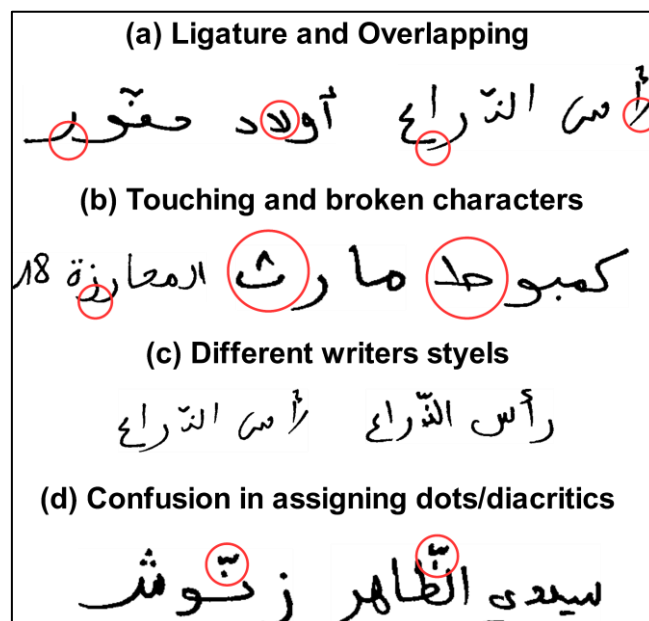


Figure 5. Challenges in handwriting Arabic recognition.

1.5 Contributions of the thesis

This thesis is related to the recognition of handwritten Arabic text by using deep neural networks (DNNs). Below are the summaries of the main thesis contributions:

1. We present and evaluate different deep learning architectures used for Arabic handwriting recognition. Each system is evaluated and compared with other state-of-the-art systems.
2. We present various modeling options for Arabic handwriting recognition, such as (character shapes, characters, and whole-word modeling) and present the experimental results using these modeling options on a publicly available dataset.
3. We present a novel data augmentation algorithm to handle the imbalanced class problem in handwritten text recognition tasks.

1.6 Thesis Structure

The thesis consists of five chapters: the present one presents Arabic handwriting features, challenges, problem statement, list of contributions, and objectives of the thesis. In chapter 2, the terminology and literature review of related work used in this thesis are presented. This chapter provides a history and the technical knowledge required for research on handwritten Arabic text recognition. Chapter 3 presents details of Arabic handwriting recognition deep learning systems and the different modeling options for Arabic text recognition. In chapter 4, we present a novel adaptive data augmentation algorithm that handles unbalanced training data in machine learning. In chapter 5, we present the experiments and the results, and a comparison with the state-of-the-art systems. The final section sets out conclusions and plans for future work. Finally, we present the conclusions of our work and some possible future works.

CHAPTER 2

LITERATURE REVIEW

Handwritten text recognition (HTR) is an active research area in Artificial Intelligence (AI), Pattern Recognition (PR), and Computer Vision (CV). In this chapter, the concepts related to and used in handwriting recognition are described in the following sections.

2.1 Pattern Recognition (PR)

The recognition of patterns is one of the human beings' most important abilities. With this ability, people gather useful information about their environment. Today, as digital computer technology has been used and developed to simulate this unique capability, it is becoming more realistic to stimulate this unique human capacity with automated machines [18].

Pattern recognition is aimed at the categorization of data (patterns) using either prior knowledge or statistical information extracted from patterns. It is usually a group of measurements or observations that classified patterns, defining points in the relevant, multidimensional space [19]. A complete system of pattern recognition includes a sensor collecting the observations to be classified or described; a feature extraction system that computes numeric or symbolic information on the observations and a scheme of classification or description using the features extracted [20] that performs the actual task of classifying or describing the observation.

Automatic (machine) identification, description, classification and grouping of patterns across a variety of scientific and engineering disciplines such as biology, psychology, medicine, marketing, computer vision, artificial intelligence, and remote sensing are all important issues. Watanabe [20] defines a pattern as the opposite of chaos; it is an entity that could have a name vaguely defined. For example, a pattern might be a fingerprint image, a handwritten cursive word, a human face or a voice signal. The pattern recognition area covers the following:

- Character Recognition: printed or handwritten.
- Image Recognition.
- Voice Data Recognition.
- Speech Analysis.
- Person Identification and Industrial Inspection.

2.2 Optical Character Recognition (OCR)

The replication of human functions by machines, like reading, is an old dream. But machine reading has grown from a dream to a reality in the last five decades [21]. OCR has become one of the best applications in pattern recognition and Artificial Intelligence (AI) in technology. For a variety of uses, there are many commercial systems for efficient OCR, even though machines still cannot compete with human readability.

OCR refers to the computer science branch, which involves the reading of paper text and translation of images into a fully editable form that can be operated by the

computer. An OCR system lets you take an electronic computer file with a hard-coded copy of your book or magazine article and then edit the file using word-processor applications.

OCR is the process by which a paper document is converted into an editable format that can be used as a typed text in word and other applications. It is already commonly used in many fields where the searches that require hours or days can be carried out in several seconds. OCR systems have a huge potential as they can edit printed records on Pcs.

2.2.1 OCR's History

OCR's research purports to replicate human functions with machines and enable the machine to carry out tasks like reading. In fact, in 1870, the origins of recognition can be found. This was the year when the Boston-Massachusetts C.R.Carey invented a retina scanner that used a mosaic from photocells as an image transmission system [22]. The Polish P. Nipkow invented the sequential scanner two decades later, which was a breakthrough for both modern TV and reading machines [23].

Several attempts were made during the first decades of the 19th century to create tools that would help blind people through OCR research. The modern version of OCR, however, did not appear until the middle of the 1940s when the digital computer was developed. From then on, the motivation for development has been potential applications in the business world.

By 1950, the high-speed technology revolution progressed, and electronic data processing became a key field. Punched cards were used in the data entry, and the

increased data quantity needed to be handled economically. At the same time, machine reading technologies matured enough to be used and in the middle of the OCR of the 1950s [24]. Reader's Digest installed in 1954, the first true OCR reading machine. This device was used to convert typed sales reports into punched cards for computer input.

Although OCR machines were already available in the 1950s on the market, only a few thousand were sold worldwide until 1986. The main reason was that the systems were expensive. The sales increased substantially, though as hardware became cheaper and OCR systems became available as software packages [25]. Today, there are several thousand systems sold every week, and over the last six years, costs for an OCR Omni-font decreased by a factor of 10 every other year. Table 2 shows a brief time frame for OCR researches.

Table 2. A Brief Time Frame for Optical Character Recognition Researches.

1870	First attempts
1940	OCR's modern version.
1950	The appearance of the First OCR machines
1960 – 1965	OCR First Generation
1965 – 1975	OCR Second Generation
1975 – 1985	OCR Third Generation
1986 – till now	OCR for the public

The first generation of OCRs could be called the commercial OCR systems in the period 1960 to 1965. The main characteristics of this generation of OCR machines were limited to read letter shapes. The symbols were designed specifically to read the machine, and the first ones were not even natural. Multi-font machines began to appear over time, and up to ten different fonts could be read. The number of fonts was limited by the template matching pattern recognition method used to compare the character picture with the prototype image library of each character of each font [26].

The second-generation reading machines came into being in the mid-1960s and early 1970s. These systems could recognize regular machine-printed characters and also could recognize a hand-printed character. The character set was limited to numerals and a few letters and symbols when hand-printed characters were considered.

Documents of low qualities and large printed and handwritten characteristics formed the challenge for the third generation of OCR systems in the early 1970s. Low cost and high performance were also an important target, helped by the significant advances in hardware technology [27].

2.2.2 OCR Methods

The main principle in automatic pattern recognition is to first teach the machine which pattern classes can occur and how they look. OCR patterns include letters, numbers, and specific symbols, such as commas, question marks, etc., while the classes correspond to the various characters. The training of the machine is done by showing the machine examples of characters of all classes. The machine creates a prototype, a description of each character class. Based on these examples. The unknown characters are then compared to the previously obtained descriptions, and the class that gives the best match is assigned.

2.2.3 OCR Applications

In recent years, commercial optical character recognition products have shown a widespread appearance that meets the needs of different users. Some of the different application areas for OCR are:

- Data Entry
- Text entry
- Process automation
- Aid for blind
- Automatic number-plate readers
- Handwriting Recognition
- Signature verification and identification
- Form readers

2.3 Handwriting Recognition Systems

A computer can receive and interpret intelligible, man-input from sources like paper documents, photographs, touch screens and other devices, known as handwriting text recognition (HWR) also as Handwritten Text Recognition (HTR). Handwritten recognition advances have helped automate many challenging tasks in our daily lives. Many applications rely on the handwritten reading of postal addresses for purposes of mail sorting, check recognition and word spotting on a handwritten text line, etc. There are various types of recognition, for example, the recognition of digit numbers, the recognition of characters, and word recognition. In general, character recognition systems can be divided into two types, as shown in Fig.6.

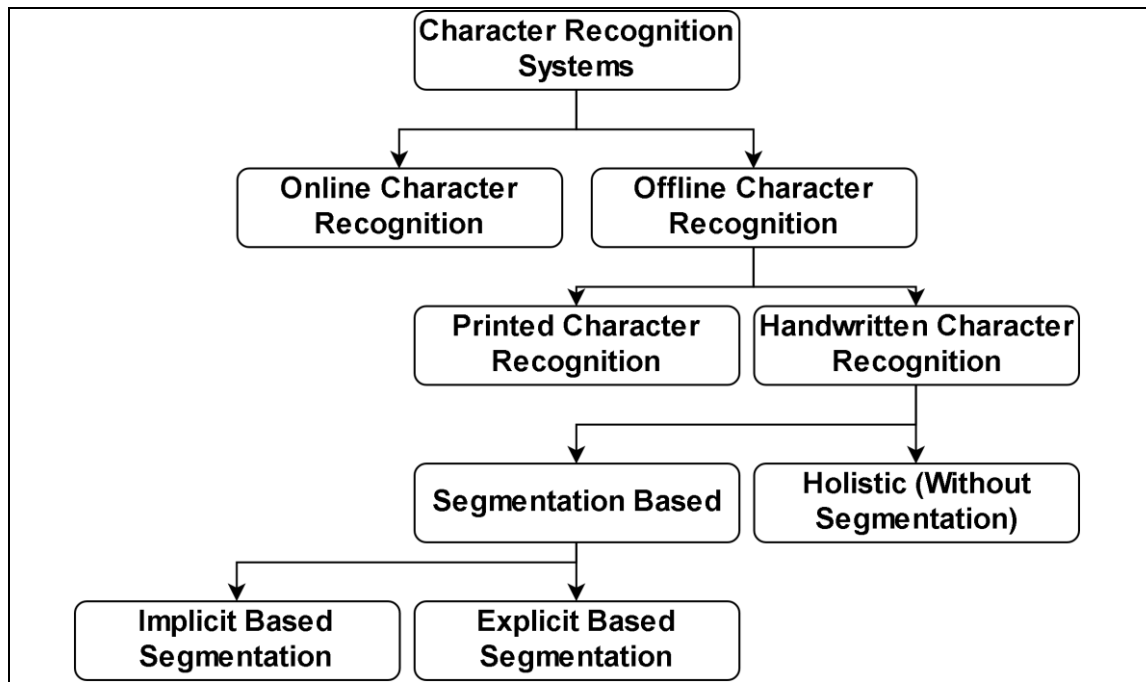


Figure 6. Classification of character recognition systems.

The handwritten text is collected and recognized in real-time in online text recognition. To generate this type, a special digitizer tablet and pen are used. A digitizer is an electromagnetic tablet that constantly transfers the pen position coordinates to the computer. A simple example of the text generation for online text recognition is the Personal Digital Assistant (PDA) and tablet. Online text recognition is however less complicated than offline text recognition since dynamic information is usually made available to recognize online text, such as the number of strokes, order of strokes, direction for each stroke, and writing speed in each stroke. This valuable information helps in document recognition and often leads to better system performance compared to offline text recognition.

The offline recognition techniques work on scanned images. The images are generally first enhanced and features extracted from the images by digital image processing techniques. However, the recognition of offline text tends to be more

challenging because of the noise caused by the processing of input devices. The offline recognition of handwritten text is more complicated than the printed text. Printed characters are uniform in shape and structure, while handwritten characters have different styles and sizes for both the same writer and different writers. Handwritten words can be recognized in two ways: whole word recognition without segmentation or segmentation-based recognition.

Generally, recognition of any handwritten script involves several steps, as shown in Fig.7. The first step is to improve the quality of the text image and to remove any unwanted details. The pre-processing step usually involves many tasks such as binarization, noise removal, baseline detection, and normalization. Next, we extract features from handwritten texts. Typically, the feature vector usually contains the characteristics of the given character or word. In the final step, we try to identify the given character or word by comparing its characteristics to one of the classes in the database, assuming that the training classes and the training model have been provided.

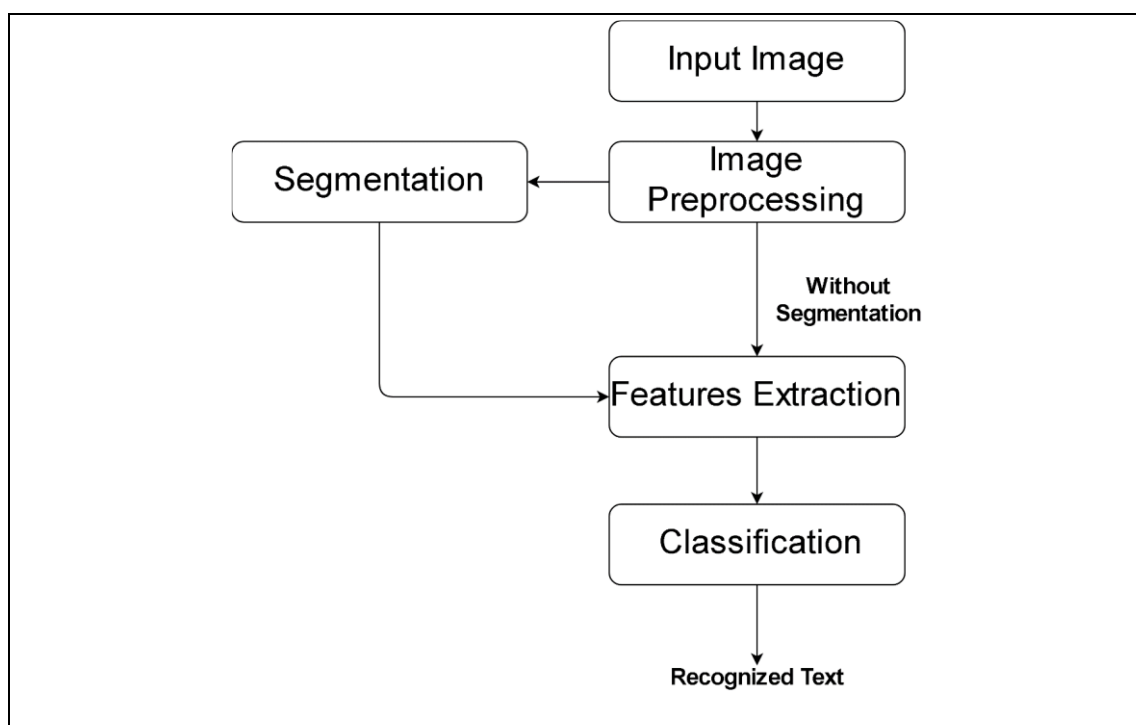


Figure 7. General steps for handwritten text recognition.

2.3.1 Pre-processing

In any recognition system, the preprocessing task is an important step [28]. It aims to improve the readability of the text image and remove those details without discriminatory power during the recognition process [29, 30]. In [31] standardized characters in size 64x64 pixel picture, Mowlaei et al. skeletonized the width of the stroke. Cheung et al. presented a simple noise reduction smoothing method in [32]. In [33], Nawaz et al. converted the image to monochrome to remove noise and skew detection and correction of drift is used. Altuwaijri et al. achieved binarization in [34] by eliminating the noise and smoothing the image. The pre-processing step usually involves several tasks: binarization, removal of noise, detection of baseline, and standardization.

2.3.2 Features Extraction

The features capture information from the text image in the printed and handwritten text. The essential characteristics of the character or word that make it different from another should be found in this information, i.e., all the characteristics must be removed and the qualities, which distinguish one character or word from another, maintained [35]. The information is forwarded to the classifier for classification. The techniques of feature extraction differ from one application to the next. Techniques that succeed in one application may not be successful for other applications [36]. The choice of the technique for the extraction of features, therefore, remains a significant step towards attaining high precision of recognition. Handwritten text characteristics can be categorized as follows:

- 1) **Structural Features**: The most common features that researchers use are structural features [37]. The geometric and topology of the text image is illustrated by describing its local and global properties. The structural characteristics will depend on the pattern category to be classified. The features for an Arabic text are, for example, dots and their location, stroke, width, and height of the stroke, directions, line segment intersection, and loop [38, 39].
- 2) **Statistical Features**: Statistical features analyze the spatial distribution of pixels by counting local features at each pixel and drawing statistics from local feature distributions [40]. The main statistical characteristics include zoning, which splits the character into overlapping or non-overlapping areas and analyzes the pixel density distribution across separate areas. In [41], Mohiuddin et al. measured the direction of the character's contour by dividing the image character into zones. Chain codes histograms are then used to calculate the direction of the contour in these areas.
- 3) **Global Transformation**: Global transformation techniques can convert pixel representation into a more compact form. Thus, a linear combination of a series of simple, well-defined functions can represent a signal. The expansion of the series provides compact encoding through the linear combination coefficients [46, 47]. Common techniques for transformation used to recognize character are:
 - i. Fourier Transform [42].
 - ii. Discrete Cosine Transform [43].
 - iii. Wavelets [44].
 - iv. Hough Transform [45,46].
 - v. Moments [47].

2.3.3 Segmentation

Although the segmentation process is the primary source of error recognition, most systems avoid this step and recognize the word as a whole especially, if the texts presented in the image are not valid words in the database or the case of open vocabulary recognition, holistic methods will fail, and we need to rely on segmentation approaches. Segmentation-based approaches are broadly divided into two categories:

1. Implicit segmentation or internal segmentation where segmentation and character recognition are achieved at the same time [48]. Implicit segmentation is also referred to as recognition-based segmentation. The implicit segmentation approach let the recognizer to decide the best segmentation hypothesis.
2. Explicit segmentation or external segmentation approaches [49], where the input word image of a sequence of characters is divided into sub-images of individual characters, which are then classified. Because of the presence of ligature as well as the cursive style of the Arabic script, several researchers proposed methods relying on the recognition of the whole term without segmentation [50, 51].

2.3.4 Classification

Classification is a task to identify an object by comparing its characteristics to one specific class. The classes and the training models are presumed to have been provided. The training model consists of several instances in which the right outputs are provided for every input. A classifier is therefore used to identify an object through its features. In the classification phase, conventional systems [52] are based on:

- i. Decision Trees
- ii. K-nearest neighbor
- iii. Hidden Markov Model (HMM)
- iv. Deep Learning Neural Networks (DLNN)

2.4 Literature Review on Arabic Handwriting Recognition

This section reviews the most popular databases used to evaluate Arabic character recognition systems.

2.4.1 A Review on Handwritten Arabic Databases

A standard database is needed for a fair comparison of the performance of Arabic handwriting systems. Many handwriting recognition systems achieve high accuracy because results were tested on a small database and not a standard database [46]. Some databases were also developed for particular domains, such as postal addresses and financial use (such as checks). In the handwritten text, there is no control over writing style or the writer's ability to produce text with touching or overlapping characters.

As mentioned before, most of the research has been carried out on the recognition of Latin handwritten characters. Many databases have, therefore, been created for handwritten English recognition [53] [54] [55] [56].

However, a limited number of Arabic handwriting databases are available that have been created for particular domains. The most popular is the IFN/ENIT database released for handwritten Arabic by Pechwitz et al. [57]. It contains 27,000 Arabic

handwritten names representing 937 Tunisian town-village names written by 411 writers and is divided into five groups for training and testing and is free for research purposes.

The Arabic Handwritten Database (AHDB) developed by Alma'adeed et al. [58, 59], written by 100 various writers, contains numbers and quantities used in bank checks in addition to the most popular Arabic words.

Al-Ohali, et al., in [60] have built an Arabic cheque database containing 29,498 sub-words, 15,175 Indian-Arabic digits and 2,499 samples each of legal and courtesy amounts extracted from 3,000 real-life cheques.

El-Sherif, et al., in [61] have developed the ADBase database for handwritten Arabic digits, containing 70,000 digits written by 700 different authors, in each of the ten digits. The database is divided into two groups (60,000 training words and 10,000 testing words). The same authors developed a modified version of the MADBase [62]. This database is in the same format as the Latin database (MNIST) so that comparisons can be made between problems with the Latin and Arabic digits. The images in the MADBase are gray, where the images of the ADBase are binary. The ADBase, as well as the MADBase, are available free of charge for research purposes. Each digital image is normalized to 28x28 pixels in size.

Awaidah et al. [63] created a database with 21,120 samples by 44 distinct authors for handwritten Arabic numbers. By using a half-transparent paper over a tabular grid, each writer wrote 48 samples. The samples were scanned at 300 dpi and turned into binary pictures. The horizontal and vertical histogram was computed to obtain the position of each digit for each scanned paper. Every digit was then saved to a file.

Lawgali et al. [64] have developed a handwritten Arabic character database. This database covered all shapes and overlaps of Arabic characters. It contains 6,600 shapes of character. Data from 50 writers aged between 14 and 50 were gathered. Each writer filled in two forms of all shapes of Arabic character e.g. (isolated, at the beginning of the word, in the middle and at the end of the word) also the shapes of overlapping characters, but without dots. The database has two versions. For training and testing, the first version is used. This version can be used to compare the efficacy of certain techniques to recognize all shapes of handwritten Arabic characters. The second version stores all shapes and can be used to train methods that recognize the word after segmentation. The HACDB is available for academic applications.

KHATT [65] (KFUPM Handwritten Arabic Text) database is an unconstrained handwritten Arabic text database written by 1,000 different writers. The database includes 2000 images of paragraphs of similar text and 2000 images of paragraphs of unique text and their extracted images of a line of text. The images are accompanied by manually verified ground-truth and Latin ground-truth representation. Arabic Character Recognition databases are listed in Table 3.

Table 3. The Databases Available for Arabic Character Recognition

Author(s)	Database name	Type of data
Pechwitz et al. [57]	IFN/ENIT	Words
Alma'adeed et al. [58, 59]	AHDB	Digit + Most popular Arabic Words
Al-Ohali, et al [60]	-	Subwords and Digits
El-Sherif et al. [61,62]	ADBase MADbase	Digits
Awaidah et al. [63]	-	Digits
Lawgali et al. [64]	HACDB	Shapes of Arabic characters and overlapping ones
Mahmoud et al. [65]	KHATT	Paragraphs

2.4.2 Related Works on Arabic Handwriting Recognition

We present in this section the literature review of offline Arabic text recognition research trying to focus on deep learning techniques. Following a deep learning trend in computer vision and Graves et al. [50] research on offline text recognition, deep learning-based recognition systems have gained a high level of interest.

Al-Hajj et al. [66] present a two-stage system for the off-line recognition of cursive Arabic handwritten words. The proposed method is analytic without segmentation and can cope with handwriting inclination and with shifted positions of diacritical marks. First, the recognition stage relies on 3 classifiers based on Hidden Markov Modeling (HMM). The second stage depends on the combination of these classifiers. The feature vectors used for recognition are related to pixel density distribution and local pixel configurations. These vectors are extracted on word binary images by using a sliding window approach with different angles. They have experimented with different combination schemes. The neural network-based combined system yields the best performance on the IFN/ENIT [58] benchmark database of handwritten names of Tunisian villages/towns.

Dreuw et al. [67] proposed a continuous HMMs system to explicitly model white-spaces for Arabic handwriting recognition within different writing variants.

Kessentini et al. [68] presented a multi-stream approach for off-line handwritten word recognition. The proposed method combines low-level feature streams, namely, density-based features extracted from 2 different sliding windows with different widths, and contour-based features extracted from upper and lower contours.

The first effective use of Deep Learning Neural Networks (DLNN) for Arabic handwriting recognition was Multi-Dimensional Recurrent Neural Networks (MDRNNs) by Graves et al [50]. Raw image pixels are given as input to a multi-dimensional RNN (MDRNN). A hierarchical architecture is designed to convert the two-dimensional image into a one-dimensional sequence which is finally labelled by the output layer.

Rothacker et al. [69] focus on feature learning by estimating and applying a statistical bag-of-features model. These models are successfully used in image categorization and retrieval. The novelty of this work was the integration with a Hidden Markov Model (HMM) that used for recognition. The method was evaluated on the IFN/ENIT [57] .

Azeem et al. [70] utilized the slanting window idea, as presented by Al-Hajj et al. [19], to develop three separate continuous HMM systems. The results of the individual classifiers were combined using sum, majority vote, and maximum rules.

Abandah et al. [71] used a system similar to the one presented by Graves [17], but the text was explicitly segmented into graphemes before extracting features from the graphemes. Their method is based on explicit text line image segmentation into graphemes. The Arabic text is segmented into PAWs and subsequently segmented into graphemes by a rules-based algorithm. Dots and diacritics were isolated from text and subsequently attached to PAWs. Several features were extracted from the segmented graphemes. For training and recognition, RNNs with bi-LSTM (BLSTMs) and CTC layers are used. The BLSTM architecture enables contextual information to be integrated from both directions through the handwritten text. Finally, a dictionary is used to correct the recognizer output based on the decoding lexicon.

Hamdani et al. [72] presented in combination with an HMM system a Bi-directional Long Short-Term Memory (BLSTM). In the first step, pixel gray values were used to extract features from repositioned sliding windows for the HMM system training. RNN-trained features were used to train HMMs in the next step. Gaussians were split into two successive HMM states. In addition, contextual HMMs (trigraphs) were used with decision-tree clusters.

Then, Elleuch et al. [73] investigated two deep architectures: Deep Belief Networks (DBN) and Convolutional Deep Belief Networks (CDBN), which are applied respectively on low-level dimension and high-level dimension in textual images for recognizing Arabic Handwritten Script (AHS). Specifically, they experimented with two data sets: the HACDB dataset for offline Arabic character recognition [64] and the ADAB dataset for online Arabic word recognition [39].

Stahlberg and Vogel [74] proposed a novel text line image normalization procedure and a new feature extraction method using the HMMs system with deep neural network training. The system was evaluated on IFN/ENIT [58] and KHATT [65] database.

Ahmad et al. [75] presented a multi-stage HMM-based system where two separate systems were trained: one for the recognition of the Arabic core-shapes after removing dots from text image and another for the recognition of the various dots and diacritics from Arabic text. During recognition, the core shapes recognition system was used to identify the core shapes of the Arabic text, and finally the Arabic word was recognized by including information from the diacritics recognition system. Ahmad et al.[76] also showed a new way of depicting Arabic characters by separating the core

shapes from diacritic ones and then by depicting them in smaller units called sub-core shapes.

Hassan et al. [77]. Proposed handwritten recognition method for offline Arabic. The proposed hand-written method for recognition Arabic word without segmentation to sub letter based on the Scale Invariant Feature Transformation Extraction (SIFT) and Support Vector Machines (SVM). The proposed method was analyzed using the database (AHDB). They previously also in [78] proposed a system depends on the segmentation of the text in word only. handwriting text recognition system based on efficient division, DCTs and HOG Features.

Alnuzali et al. [79] examined ELM and SMO classifiers with Gabor's features. The evaluation is conducted on two Arabic Cheque datasets, AHDB and CENPARMI. The results from Gabor are superior to earlier studies with the SMO classification.

Rabi et al. [80] proposed a cursive handwritten Arabic text recognition system based on Hidden Models (HMM). This system is analytical and uses embedded training to perform and improve the character models without explicit segmentation.

Almodfer et al. [81] improved AlexNet's recognition of handwritten Arabic words by adopting a drop-out regularization that prevented the proposed system from overfitting problems and reduced word error rates. Besides, they investigated the performance of ReLU and tanh activation functions in the fully connected layers. Also, they have present in [82] an offline Arabic handwriting recognition system by using the multi-column deep neural networks. Across multiple experiment settings using the IFN/ENIT benchmarking database, they have increased the accuracy rate of the system.

CHAPTER 3

METHODOLOGY

3.1 Modeling Options for Arabic Text Recognition.

In this chapter, we present the various modeling options we examined during this work. In the next three sub-sections, the four different but relevant modeling approaches that we have examined will be presented. Four broad options for Arabic handwriting modeling approaches are thus described:

1. Whole-Word Modeling
2. Using the characters as models
3. Using the character's shapes as models.
4. Modeling core shapes and diacritics separately.

3.1.1 Whole-Word Modeling (Holistic-Handwriting-Recognition)

Arabic words or PAWs have been used by some researchers as modeling units. There are many issues with this approach. One such problem is that it usually leads to a huge number of classes as every word is represented by a separate class. A lexicon may have hundreds, or thousands, or even more words. This will also mean the need for more training data as each word should have sufficient samples in the training set to adequately train the model parameters. It may be suitable only for text recognition tasks involving a very small lexicon size [16, 57].

3.1.2 Using Characters as modeling units

Researchers selected characters as modeling units, mainly during the initial days of Arabic text recognition. Since Arabic characters have many position-dependent forms, it is not best to use characters as models. In this modeling option, the leading Arabic characters (28 characters, ten digits, and seven ligatures) are used as modeling units.

3.1.3 Using Character shapes as modeling units

The most successful and widely used approach is the use of Arabic character shapes as distinct models [75]. Although this approach is generally well-functioning, there is a problem with using it. The number of modeling units increases four times from 28 (i.e., the number of Arabic characters) to approximately 100, excluding numbers and special characters. This creates a large model set with a bulky recognizer. Besides, a wide range of modeling units requires large amounts of training data to ensure that each model is adequately trained.

3.1.4 Modeling core shapes and diacritics separately

This modeling choice was presented by Ahmed et al. [75,76]. Due to the fact that many Arabic characters have the same core shapes, but are only vary based on the numbers of dots as well as other diacritics we can isolate the main shapes from the diacritics, then the model set will end up being drastically reduced. Based on these findings we have examined a multi-stage text recognition approach, which is used to

independently train a core-shape recognition system from a diacritical recognition system, and which incorporates the outcomes of both systems to create the final text recognition hypothesis. The outline of the Multi-Recognition System is presented in the next few sub-sections. Fig.8 shows some character classes that have the same core structure but that are distinct because of the diacritics presence or absence. Fig.9 shows that the words (ثابت ، كاتب ، خالق) could be built with 7 core shapes and four diacritics one-dot above, two-dots above, three-dots above and one-dot below.

Isolated character shapes	Core shapes	Beginning character shapes	Core shapes	Middle character shapes	Core shapes	Ending character shapes	Core shapes
ا ا ا ا	ا					ا ا ا ا	ا
ب ب ب ب	ب					ب ب ب ب	ب
ي ي ي ي	ي	ب ب ب ب	ب	ب ب ب ب	ب	ي ي ي ي	ي
ن ن ن ن	ن	ي ي ي ي	ن	ي ي ي ي	ن	ن ن ن ن	ن
ح ح ح ح	ح	ح ح ح ح	ح	ح ح ح ح	ح	ح ح ح ح	ح
د د د د	د					د د د د	د
ر ر ر ر	ر					ر ر ر ر	ر

Figure 8. Some character shapes to core shapes mapping for Arabic script. (Figure adapted from [74])

ثابت	كاتب	خالق
ثابت	كاتب	خالق
ب	ا	ب

Figure 9. Words representation using core-shape.

We need to separate the core shapes and the diacritics from the text images in order to model and train them separately. A text image is thus divided into two images: one with the core forms and the other with the diacritics. Ahmad et al. [75] presented the core shapes and the diacritics separation algorithm. Besides, they have modified the transcriptions for the text images of the IFN/ENIT training sets. They created two separate transcriptions one for the core shapes and the other for the diacritics [75]. Fig.10 shows the pseudo-code for separating core-shapes and diacritics with two illustration examples.

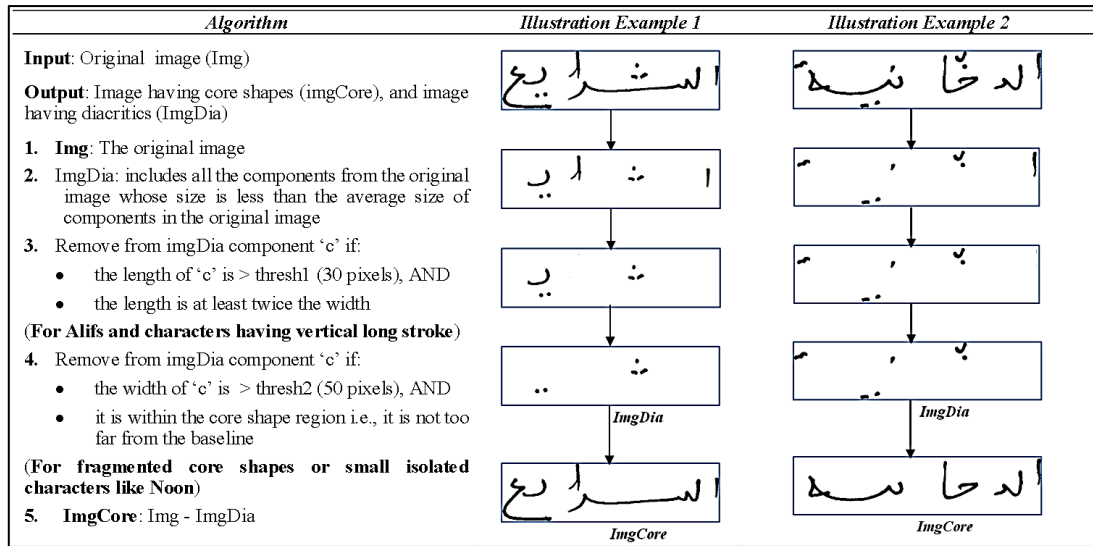


Figure 10. A rule-based algorithm for separating core-shapes and diacritics. (Figure adapted from [75])

3.2 Deep Learning Approaches for Handwriting Recognition

Image Classification and object recognition using deep learning is the state-of-the-art technology in the field of Computer Vision (CV) and robotics. In this section, we focus on building an efficient and robust Deep Learning Neural Network (DLNN) to recognize handwritten Arabic scripts. There are numerous handwritten recognition methods [5], depending on whether one considers an explicit or implicit character

segmentation, the type of features used, or the choice of classifier implemented for characters' recognition. The common point of these approaches is that they all rely on a lexicon driven decoding stage. Indeed, since character recognition is not free of error, the common word recognition strategy is to postpone the character decision process at the end of the word recognition process, where the best character hypotheses combination forming a valid word are finally selected. The various stages in DLNN along with its operation is discussed in detail.

3.2.1 Convolutional Neural Networks (CNN)

The convolutional neural network is a class of artificial deep-feed-forward neural networks, usually used for image classification. CNN consists of several layers which can learn high-level features efficiently from labeled training data. In fact, its key idea is to use convolutional and pooling layers to gradually extract more abstract patterns that can be carried out locally [83]. CNN is like other deep neural networks consist of an input layer, an output layer, and many hidden layers in between. These layers perform data-specific operations to learn features.

In 1998, Yann LeCun has presented the LeNet5 network [84] one of the initial convolution neural networks which propelled the area of deep learning. At that time the LeNet architecture was used mainly for character recognition tasks such as reading zip codes, digits, etc. There have been several new architectures proposed in recent years which are improvements over the LeNet, but they all use the main concepts from the LeNet. The Convolutional Neural Network in Figure 11 is similar in architecture to the original LeNet.

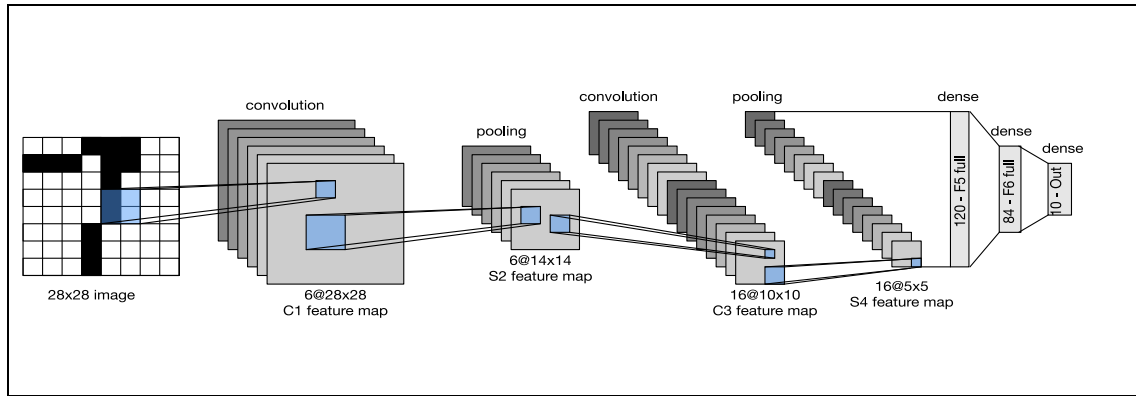


Figure 11. LeNet 5 [77].

In general, there are four main building blocks in the CNN shown in Figure 11 above:

1. Convolution
2. Non-Linearity (ReLU)
3. Pooling or Sub Sampling
4. Classification (Fully Connected Layer)

A convolutional layer puts the input images through a set of convolution filters; each filter will extract a specific feature from the image. In other words, the filter is slid across the width and height of the input and the dot products between the input and filter are computed at every spatial position. The output volume of the convolutional layer is obtained by stacking the activation maps of all filters along the depth dimension. Since the width and height of each filter are designed to be smaller than the input, each neuron in the activation map is only connected to a small local region of the input volume. In other words, the receptive field size of each neuron is small and is equal to the filter size. The local connectivity is motivated by the architecture of the animal visual cortex [85] where the receptive fields of the cells are small. The local connectivity of the convolutional layer allows the network to learn filters that maximally respond to a local region of the input, thus exploiting the spatial local correlation of the input (for an input image, a pixel is more correlated to the nearby pixels than to the distant pixels). Besides,

as the activation map is obtained by performing convolution between the filter and the input, the filter parameters are shared for all local positions. The weight sharing reduces the number of parameters for efficiency of expression, efficiency of learning, and good generalization.

Let layer l be a convolutional layer. Suppose that we have some $M \times M$ square neuron nodes which are followed by our convolutional layer. If we use an $N \times N$ filter W then convolutional layer output will be of size $(M - N + 1) \times (M - N + 1)$ which produces k -feature maps as shown in Fig.12.

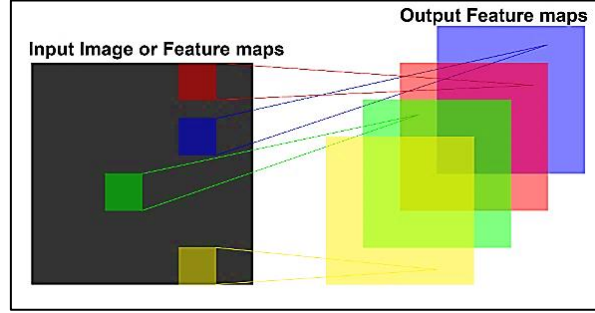


Figure 12. Illustration of a single convolutional layer that produces k -feature maps

The convolutional layer acts as a feature extractor that extracts salient features of input such as corners, edges, and endpoints. To compute the pre-nonlinearity input to some unit. Then the input of layer $l - 1$ comprises is computed as:

$$Y_i^l = B_i^{(l)} + \sum_{a=1}^M \sum_{b=1}^M W_i X_{(i+a)(j+b)}^{l-1} \quad (1)$$

Where B_i^l is a bias matrix and W_i^l is the filter of size $N \times N$. Then the convolutional layer applies its activation function as:

$$Z_i^l = \sigma(Y_i^l) \quad (2)$$

The activation function that is used in our proposed approach is Rectified Linear Units (ReLU) that applies the non-saturating $\sigma(Y_i^l) = \max(0, Y_i^l)$. We apply the ReLU non-linearity as an activation function to the output of every convolutional layer and fully connected layer. The ReLU increases the nonlinear properties of the decision function and the overall network without affecting the receptive fields of the convolution layer.

Often convolutional layers are followed by a pooling layer. Pooling layers compute the maximum or average over a region of a feature map. This results in a subsampling of the input. Pooling reduces the dimensionality, increases the scale, and also supports translation invariance in the sense that a slight translation of the input does not (or only marginally) change the output. Fig.13 shows the pooling layer takes a small rectangular block from the convolutional layer and subsamples it to produce a single output from that block. The pooling layer reduces the resolution of the image that reduces the precision of the translation (shift and distortion) effect. There are several ways to do this pooling, such as taking the average or the maximum, or a learned linear combination of the neurons in the block. Our pooling layers will always be max-pooling layers; that takes the maximum of the block that they are pooling.

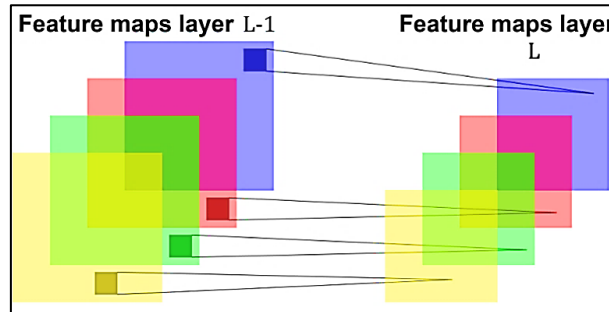


Figure 13. Illustration of pooling layer that produces the same k -feature maps

Convolutional layers and pooling layers take the spatial structure of their input into account, which is the main reason for their good performance in computer vision

and image analysis. In contrast, when standard non-convolutional neural networks are applied to raw (or only slightly preprocessed) images, the images are vectorized. The resulting vectors serve as the input to the neural network. In this process, information about the spatial relation between pixels or voxels is discarded and there is neither inherent equi-variance to translation nor translation invariance, which are important properties to achieve generalization in object recognition tasks.

Finally, after several convolutional and pooling layers, the high-level reasoning in the neural network is done via fully connected layers. A fully connected layer takes all neurons in the previous layer (be it fully connected, pooling, or convolutional) and connects it to every single neuron it has. Fully connected layers are not spatially located anymore (you can visualize them as one-dimensional), so there can be no convolutional layers after a fully connected layer.

In this work, we have adopted the popular AlexNet for Arabic handwritten words recognition. AlexNet (proposed by Krizhevsky et al. [86]) is one of the deep convolution nets designed to deal with complex scene classification tasks on the ImageNet database. The task is to classify the given input into one of the 1000 classes. The arrangement and configuration of all the layers of AlexNet are depicted in Fig. 14.

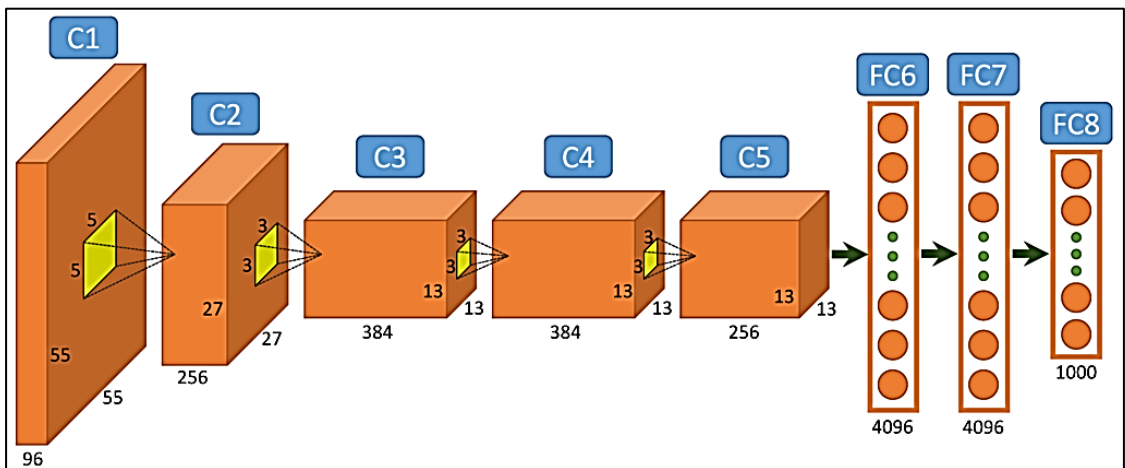


Figure 14. AlexNet architecture [86].

AlexNet includes five convolutional layers, three max-pooling layers, two Local Response Normalization (LRN) layers, three fully connected layers, and a linear layer with softmax activation in the output. The fully connected layers are simply implementing the dot product between the input and weight vectors, where each neuron in layer l is connected to all outputs of neurons in layer $l - 1$. Moreover, it uses the dropout regularization method to reduce overfitting in the fully connected layers and applies Rectified Linear Units (ReLUs) for the activation of those and the convolutional layers function to speed up the learning process. It was initially trained to recognize 1000 different objects. The intuition behind this network is that each convolutional layer learns a more detailed representation of the images (feature map) than the previous one. For example, the first layer can recognize very simple forms or colors, and the last one more complex forms such as full words for instance. The convolutional layers are representing the image in a much better way for the classification. After the convolutions, each image is represented as a vector of 4096 features (whereas they were initially vectors of $227 \times 227 \times 3 = 154,587$ features). The two fully connected and SoftMax layers are similar to a multi-layer perception and could be replaced by other kinds of classifiers such as Random Forests or SVMs. However, they are really important for the training phase of the neural net.

3.2.2 Recurrent Neural Networks (RNNs)

The main focus of the research is on the Recurrent Neural Networks (RNN), which is a specific form of the neural network defined by the presence of interconnected nodes. A Recurrent Neural Network (RNN) is a robust deep neural network, commonly used to solve problems of machine learning involving sequence inputs. This particular

ability enables RNN to learn the context of labels, as the word's characters cannot be regarded as genuinely independent components, which is a great advantage in handwriting recognition.

Although the traditional RNN gives the network a memory, its ability to preserve old contents is still quite limited due to the vanishing gradient problem [87]. Usually, it is difficult for an RNN to bridge gaps between the presentation of the relevant input and the relative target events over more than 10-time steps. Because of this, the Long Short-Term Memory (LSTM) [88] model is used to overcome this problem. RNNs architecture involving (LSTM) enables it to capture longer contexts which may be important for offline text recognition tasks.

RNNs, are capable of modeling sequential data for sequence recognition and prediction. Unlike traditional neural networks, input and output data are assumed to be independent of each other. In many applications, using this assumption is not a good idea.

A simple RNN has three layers, as shown in Fig.15, which are input, recurrent, and output layers [89]. There are N input units in the input layer. This layer's inputs are a sequence of vectors over time t such as $\{\dots, \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}, \dots\}$ where $\mathbf{x}_t = \{x_1, x_2, x_3, \dots, x_N\}$.

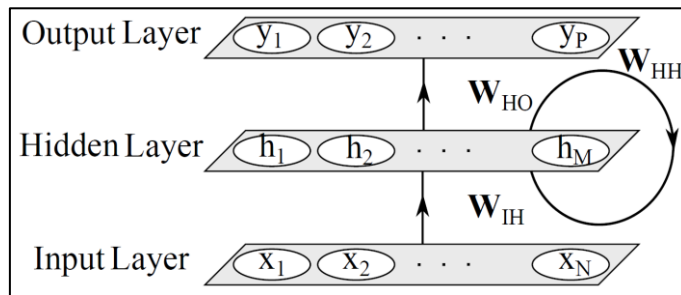


Figure 15. A folded recurrent neural network [89].

The input units in a fully connected RNN are connected to the hidden units in the hidden layer where a weight matrix $\mathbf{W}_{IH}$ is used to define the connections. The hidden layer has $\mathbf{M}$ hidden units $\mathbf{h}_t = (h_1, h_2, \dots, h_M)$ connected over time with recurrent connections [89] see Fig.16.

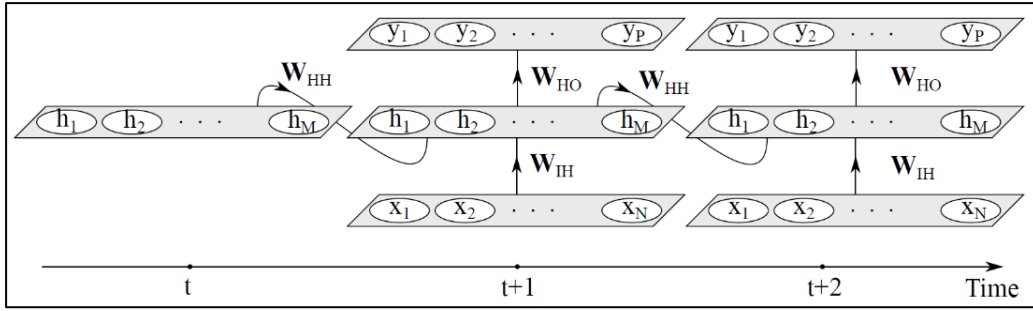


Figure 16. Unfolded recurrent neural network through time [89].

The initialization of hidden units with small non-zero elements may improve overall network performance and stability [15]. The hidden layer defines the system's state space or "memory" as

$$\mathbf{h}_t = fH(\mathbf{o}_t), \quad (3)$$

where

$$\mathbf{o}_t = \mathbf{W}_{IH}\mathbf{x}_t + \mathbf{W}_{HH}\mathbf{h}_{t-1} + \mathbf{b}_h, \quad (4)$$

$fH(\cdot)$ is the activation function of the hidden layer, and $\mathbf{b}_h$ is the hidden units' bias vector. The hidden units are connected with weighted $\mathbf{W}_{HO}$ connections to the output layer. The output layer has P units $\mathbf{y}_t = (y_1, y_2, \dots, y_P)$ that is computed as

$$\mathbf{y}_t = f_o(\mathbf{W}_{HO}\mathbf{h}_t + \mathbf{b}_o) \quad (5)$$

where

$f_o(\cdot)$ is the activation functions and $\mathbf{b}_o$ is the bias vector in the output layer.

As the input-target pairs are sequential over time, the above steps are repeated over time $t = (1, \dots, T)$.

The equations (3) and (5) show that RNN consists of absolute non-linear equations of state that can be iterated over time. The hidden states provide a prediction in each time step based on the input vector at the output layer.

The hidden state of an RNN is a set of values that, apart from the effect of any external factors, over several time steps, summarizes all the unique information needed about the network's past states. This integrated information can define the network's future behavior and predict accurately at the output layer [90]. In each unit, an RNN uses a simple nonlinear activation function. However, such a simple structure is capable of modeling rich dynamics if it is well trained in time steps.

3.2.3 Long Short-Term Memory (LSTM)

Despite having the right framework for modeling temporal data, RNNs suffer from some inadequacies. Neural networks learn how to modify network weights by back-propagating the error. If the network is too large or the number of time-steps is too long, the distance over which the error has to be transmitted will become unfeasible and the network will stop learning. In literature, this problem is referred to as the problem of the vanishing gradient. The LSTM [91] is a modified RNN designed to overcome vanishing gradients. In addition to the hidden state, the LSTM carries a cell state that preserves long term information. The LSTM solves the problem by having a shortcut path to transmit the gradients back. This unique architecture involves three gates. The forget gate decides the contents of the cell state that are relevant to the

problem and discards everything else. The input gate's function is to ensure that relevant information in the input is stored in the cell state. Finally, the output gate adds the relevant input to the cell state and passes it on to the next time-step. This architecture ensures that the gradients are propagated back into the earlier time-steps in the network.

In 1997 Hochreiter et al. [92]. introduced the first LSTM cell as shown in Fig.17. This cell includes an input t , input modulation g_t , forget f_t and output o_t gates in addition to the memory cell c_t and the hidden unit h_t . as:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (6)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (7)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (8)$$

$$g_t = \sigma(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (9)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (10)$$

$$h_t = o_t \odot \phi(c_t) \quad (11)$$

Where W are the weights matrices between input or hidden state to the gates and b are the biases terms.

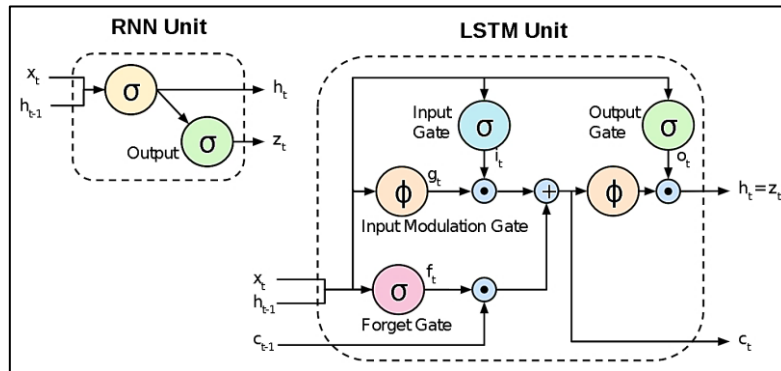


Figure 17. RNN cell unit vs. LSTM cell unit [91,92].

3.2.4 Connectionist Temporal Classification (CTC) Layer

In the context of RNN, the use of Connectionist Temporal Classification (CTC) enables recognition without prior segmentation [50]. This has enabled the use of neural network classifiers for offline recognition of handwritten texts, which has become more popular in recent years [11,50,71,72].

The segmentation process for Arabic text, printed or handwritten, is a highly error-prone task. The Connectionist Temporal Classification (CTC) layer can transcribe the data without prior segmentation. Graves et al. [11] presented the CTC layer which is dedicated to transforming RNN outputs into class posterior probabilities. It was initially designed to recognize speech [14], and then the idea was extended to recognize handwriting [50]. The CTC layer predicts the transcription for the test images in the recognition phase.

By interpreting the network output as a distributed probability over all possible label sequences on the given input sequence, the CTC uses an objective function integrating a variable to determine the best path through a lattice of possibilities. These variables enable the calculation of error between the network output and the ground truth at every time step for every label. The CTC layer has $N+1$ (N outputs basic class labels and one for blank). These outputs specify the probability that each label is observed or not observed at a given time. With the use of single-character probabilities, a set of probabilities can be obtained that possible match outputs given to an input one. This logic can be described by equation (12).

$$p(\pi|x) = \prod_{t=1}^T y_{\pi_t}^t, \forall \pi \quad (12)$$

where:

- x : The input sequence.
- π : A possible output sequence.
- $y_{\pi_t}^t$: The probability of a given label from π at time t

A special β operator is defined for the correspondence between previous steps π and the final output sequence l through the removal of blank labels. As more than one sequence of π can be a single final sequence, the probability of a specific final sequence of l will be calculated as follows :

$$p(l|x) = \sum_{\beta(\pi)=l} p(\pi|x) \quad (13)$$

The maximum probability $p(l|x)$ is considered to be the final output for each given input x and sequence l . For handwriting recognition, a word image is considered a time sequence, in which time is modeled as one unit of time along the width of the image with a one-pixel slice. Following pre-processing, pixel values have been used as network input features. The general recognizer structure is displayed in Fig.14.

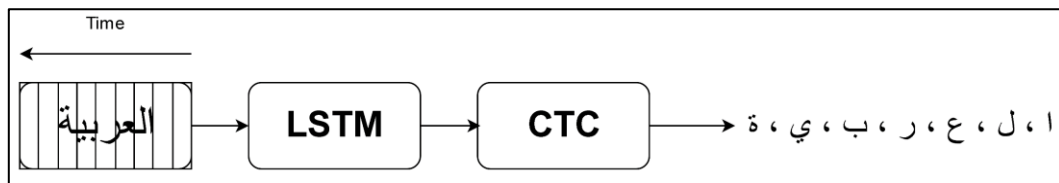


Figure 18. CTC output layer connected to the LSTM network.

3.2.5 Token Passing Decoder

Graves et al. [93] introduced the token passing algorithm which restricts its output to a dictionary. The token passing decoder solves the problem of word decoding, i.e. the most likely word in the input sequence, where the word output is limited by

vocabulary. The CTC token passing algorithm for single words expects a sequence of letter probabilities of length t as provided by the neural network, along with a word w as a sequence of ASCII characters, and returns a corresponding score, i.e. the probability that the input to the neural network was indeed the given word [14]. This means that, the probability $p(l|x) = \sum_{\beta(\pi)=l} p(\pi|x)$ in (13) could be approximated by:

$$p(w|x) = \max_{\pi, B(\pi)=w} p(\pi) \quad (14)$$

3.2.6 Word Beam Search (WBS) Decoder

The output of the CTC layer is a matrix containing character-probabilities for each time-step (horizontal position). This matrix must be decoded to get the final text. One algorithm to achieve this is beam search decoding which can easily integrate a character-level language model. Word Beam Search (WBS) or beam search decoding is an algorithm used in the CTC layer. The CTC-WBS decoder is placed just following the RNN layers for output decoding. The main advantages of WBS decoder over token passing decoder are :

- ❖ It is faster than token passing.
- ❖ It allows an arbitrary number of non-word characters between words (numbers, punctuation marks).
- ❖ Words constrained by dictionary.

Scheidl et al. [94], presented the Vanilla Beam Search (VBS) algorithm. The WBS decoder which is a modification of VBS decoding has the following characteristics :

- ❖ Words are limited to words in the dictionary.
- ❖ Any non-word character number between words is permitted.
- ❖ Optionally, we can integrate an LM word-level bigram.
- ❖ A better runtime (in terms of time-complexity and real-time on a computer) than token passing.

The iterative beam search decoding creates and scores the text (beams). Table III displays the pseudo-code for a basic-version : the beam list is initialized with an empty beam (line.1) and the corresponding scoring score (line.2). The algorithm then iterates the neural network output matrix in all-time stages (line.3 to 15). At each time-step, only the best scoring beams from the previous time-step are kept (4). The beam width (BW) specifies the number of beams to keep. For each of these beams, the score at the current time-step is calculated (8). Further, each beam is extended by all possible characters from the alphabet (10) and again, a score is calculated (11). After the last time-step, the best beam is returned as a result (16).

Table 4. Pseudo-Code for A Basic Word Beam Searching [94].

Input	: NN output matrix mat, BW
Results	: decode text
1	beams= { $\emptyset$ }
2	score { $\emptyset$,0} = 1
3	for $t = 1 \dots T$ do
4	$bestBeams = bestBeams(beams, BW);$
5	$beams = \{ \};$
6	for $b \in bestBeams$ do
7	$beams = beams \cup b;$
8	$scores(b, t) = calcScore(mat, b, t);$
9	for $c \in alphabet$ do
10	$b' = b + c;$
11	$scores(b', t) = calcScore(mat, b', t);$
12	$beams = beams \cup b';$
13	end
14	end
15	end
16	return $bestBeams(beams, 1);$

Beam search allows arbitrary character strings, which are required to decode numbers and punctuation marks while token-passing limits its output to words in the dictionary to avoid spelling mistakes. Of course, all words to be recognized in the dictionary must be included.

3.2.7 Bi-Directional Long Short-Term Memory (BLSTM)

The Bi-directional Long Short-Term Memory (BLSTM) architectures were introduced by Graves [50, 93] to overcome limitations of LSTM. One of the main drawbacks of RNN and LSTM is that they are processing information just based on the previous context.

In many applications, including speech recognition, we prefer another structure called Bi-directional LSTM (BLSTM), which can be trained using all available input information in the past and future of a specific time frame. BLSTMs contain two separate hidden layers, one processes the input sequence forwards (positive time direction), while the other processes it backward (negative time direction).

Both hidden layers are connected to the same output layer, providing it with access to the past and future context of every point in the sequence. BLSTM outperforms unidirectional LSTMs and standard RNNs, and it is also much faster and more accurate. Fig.13 shows the architecture of a BLSTM network.

The Bi-directional RNN can principally be trained with the same algorithms as a regular unidirectional RNN because there are no interactions between the two types of state neurons and, therefore, can be unfolded into a general feed-forward network.

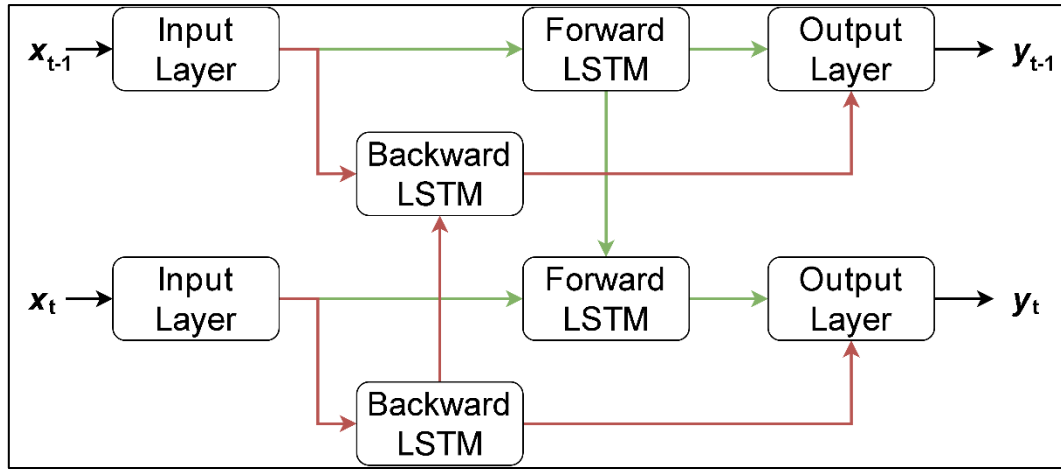


Figure 19. The architecture of a bi-directional recurrent neural network.

3.2.8 Multi-stage Deep Neural Network

In section 3.1.4, we have explained how to model core shapes individually from the diacritics, and later mix the knowledge from each source to form the ultimate text recognition hypothesis. The idea of a multi-stage classifier was originally proposed by Ahmed et al. [75]. In this approach, two deep learning systems are trained individually and severally from one another. After splitting the image into core and diacritics component. We use those components to train two separate (RNN) networks. Finally, we use a third classifier to obtain the final result based on the output of the first two networks. Fig.20 illustrates with the help of an example of the multi-stage text recognition framework.

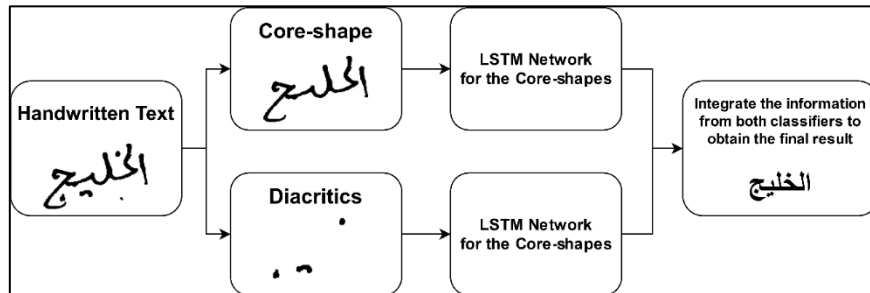


Figure 20. An illustration of the multi-stage word recognition system.

The LSTM core-shape system is the primary detection engine. For character recognition, the core-shape image and a diacritics image are separated from the input image. This system uses the characteristics from the image of the core-shape to assume a sequence of core shapes. Then, it uses the character to core shape dictionary for decoding. A character recognition network is constructed based on the hypothesized core-shape sequence. The diacritics system uses the characteristics of the diacritics image together with the designed character network to create the final hypothesis of identification. It uses the character to diacritics dictionary for decoding.

For word recognition, the core-shape system uses the features of the image of the core-shape to produce an N-best word list. For decoding, it uses a word-to-core dictionary. A network for word recognition is built to only contain the N-best words. The diacritic system uses the diacritic image features together with the network of words built in to record the N-best words. It uses the dictionary word to diacritics for scoring. System scores are added for each word and the word with the best score is produced as the final hypothesized word.

CHAPTER 4

ADAPTIVE DATA AUGMENTATION ALGORITHM

4.1 The Problem of Imbalanced Data Classification

Imbalanced classifications pose challenges as they lead to systems with poor predictive performance, especially for the minority class. A neural network that is trained on a database with imbalanced classes will bias its decision towards the prevalent classes in the set [95]. In the presence of limited training samples, data augmentation is commonly used in many deep learning approaches. Augmented data using standard augmentation techniques e.g. (rotation, position shifting, zoom, shear, etc.) does not affect the distribution of labels in the original dataset. It means that if we have imbalanced data, the data will continue to be imbalanced after data augmentation.

A neural network that is trained on a database with unbalanced classes will bias its decision towards the most prevalent class in the set [95]. If there is a class imbalance in the training data, classifiers usually overclassify the majority because of its increased prior probability. Consequently, minority classes are more frequently misclassified than those of the majority. To learn from imbalanced data, it is important to consider the representation of minority and majority classes. Krawczyk [96] suggested that irrespective of class disproportion, good results could be obtained if both groups are well represented and come from non-overlapping distributions. Japkowicz [97] examined the effects of class imbalance by creating artificial data sets that combine complexity, training set size, and degrees of imbalance. The results show that sensitivity to an imbalance increases with increased problem complexity and that non-

complex, linearly separable problems do not have any impact on all levels of imbalance. There is a genuine lack of data in some areas because of the low frequency with which events happen, such as handwriting recognition. Learning from extreme class imbalanced data, in which the minority class accounts for 0.1 percent or lower of training data [96, 98], is extremely important since usually, we are interested in these rare occurrences.

The issue of class imbalance based on the difference in character frequencies is not specific to any language or dataset. It is a common observation for texts in natural languages. For example, characters ‘e’, ‘t’, and ‘a’ are the most frequent characters whereas characters ‘z’, ‘q’, and ‘x’ are the least frequent characters in the English language [99]. Similarly, for Arabic, based on text analysis of an Arabic corpus containing more than five million characters [100], it was found that not only the characters are unevenly distributed but also the distribution is highly skewed. The two most common characters, i.e., alif (ا) and lām (ل), constituted almost a quarter of all the texts. On the other hand, the five least occurring characters (out of the total of 28 characters) constitute only around 2% of the entire text. The frequency of character zā’ (ظ) was almost 70 times lower than the frequency of character alif (ا).

Class imbalance handling methods in machine learning can be categorized into three categories: data-level techniques, algorithm-level methods and hybrid approaches. Techniques at the data level attempt to reduce imbalance levels by different methods of data sampling. Algorithmic methods of handling imbalance in classes, commonly implemented with weight or cost schemes, include the modification of the underlying learner or its output to reduce bias in favor of the majority group. Finally, hybrid systems combine both sampling and algorithmic methods strategically.

4.2 Adaptive Data Augmentation Algorithm

Now, if we consider the character shapes distribution rather than characters, the distribution will be even more skewed. There will be very few occurrences in some character shapes from the low-frequency characters and thus there is the issue of adequately training them. In this section, we present a novel data augmentation algorithm that can artificially balance the distribution of any handwriting database.

Our algorithm is simple and straightforward. First, we need to know the size of the database lexicon (L) and the number of modeling units or classes (n). For example, if we use Arabic characters as modeling units, in this case, we will have 45 classes (28 basics letters, 10 digits, and 7 ligatures) ($n=45$).

Second, we assign a weight (W_i) to each word in the lexicon. This weight is calculated for each word based on the sum of the inverse average probability of the characters that make up that word. The inverse probability were used because we are interested in words composed of infrequent characters. Therefore, words with a high weight are composed of characters that infrequently appear in training data, while words with a low weight value are composed of characters frequently seen in training data.

In general,

$$W_i = \frac{\sum_1^N \frac{1}{p_n}}{N} \quad (15)$$

where $i = \{1, \dots, L\}$, N is the total number of characters in the word i and p_n is the probability of each modeling unit (class) in the training data sets.

$$p_n = \frac{\text{Total Number of class (n) occurrence}}{\text{Total Number of all classes occurrence}} \times 100 \quad (16)$$

After finding all the weights, we normalize them by dividing each of the weights by the sum of all the weights. Therefore, normalized weights are like probabilities:

$$W_{i,normalized} = \frac{w_i}{\sum w_i}, \text{ where } i = \{1, \dots, L\}; \quad (17)$$

Now we can use these to augment new samples to the original database e.g. if we want to augment M new samples to the original database then we can calculate the number of augmented samples needed for each word image by:

$$[W_{i,normalized} \times M] = \text{Total No. of samples augmented for class } i. \quad (18)$$

By applying this to all the words in the lexicon, we will eventually balance the database, because we augment new data based on the inverse average probability of occurrence of each class. Words composed of infrequent characters will be augmented by a greater number compared to words composed from high frequent characters. Table 5 shows the pseudo-code of our data-augmentation technique.

Table 5. Pseudo-Code for Adaptive Data Augmentation Technique.

Step 1: Determine the lexicon size of the given database (L) and the number of modeling units (n).
step 2: For each word (i) in the lexicon, calculate the inverse probability of each model ($1/p_n$).
step 3: For each word (i) in the lexicon, $W_i = \frac{\sum_1^N \frac{1}{p_n}}{N} \quad \text{where } i = \{1, \dots, L\};$
step 4: For each W_i , we normalize it by: $W_{i,normalized} = \frac{W_i}{\sum W_i} \quad \text{where } i = \{1, \dots, L\};$
step 5: Determine the number of new augmented data (M).
step 6: For each word (i) in the lexicon, $\text{No. of samples augmented} = M \times W_{i,normalized}$

CHAPTER 5

EXPERIMENTS AND RESULTS

In the next subsections, we present the IFN/ENIT database, AHDB database, data preprocessing, experiment configurations, results of the evaluation and the comparison with the state-of-the-art systems.

5.1 The IFN/ENIT Database

The IFN/ENIT database is the most widely used and popular database for handwritten Arabic text recognition research published by Pechwitz et al [57]. It consists of 1000 forms, written by 411 distinct authors and it is divided into five training and testing groups and it is freely available for research purposes. It includes approximately 27,000 handwritten Arabic names representing Tunisian city names. The size of the lexicon is 937 place names. A Ground Truth (GT) file has been compiled for every word in the database. This file contains information on the word, such as the baseline position and information about the individual characters used. Fig. 21 shows some sample images from the IFN/ENIT database.

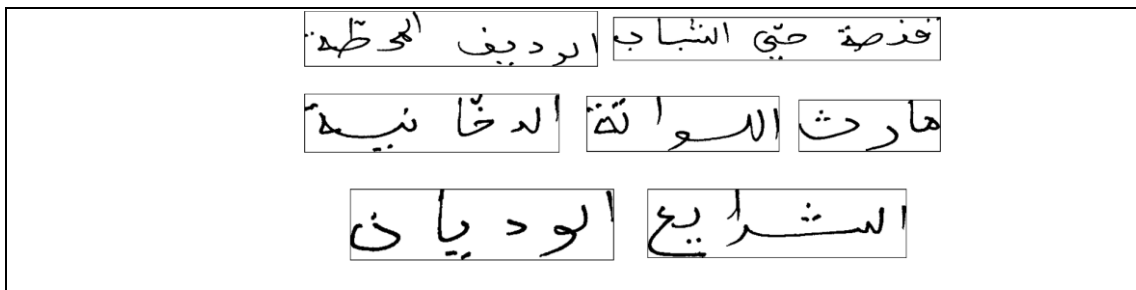


Figure 21. Samples from the IFN/ENIT database.

5.2 The AHDB Database

The Arabic handwritten database (AHDB) developed by Alma'adeed et al. [58,59] includes approximately 10,081 handwritten Arabic words representing numbers and quantities used in cheques, as well as the most popular words in Arabic writing. The lexicon size is 96 words, 67 of which were handwritten words corresponding to numbers that can be used in handwritten cheque writing. The other 29 words were the most popular Arabic words. Fig.22 shows some sample images from the AHDB database.

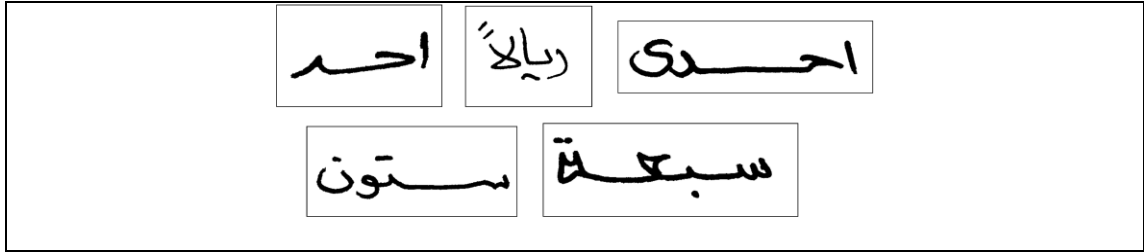


Figure 22. Samples from the AHDB database.

5.3 Preprocessing

In any recognition system, the pre-processing of the text images is an important step. The purpose of this step in handwritten text recognition is to improve text image readability and remove those details that do not have any discriminative power in the recognition process. The images were already binarized by the authors [57,58], and the only steps we took in the pre-processing were:

For the LSTM-CTC network, we resized all images to a fixed height of 96 pixels, and the width was chosen such that each image keeps its original aspect ratio. This step is essential in our system to ensure constant vector dimensions because LSTM expects a fixed-size vector.

The depth of the proposed deep learning networks is higher than traditional classifiers because of the various hidden layers. In this case, we face this serious problem because we have limited data. To overcome this issue, we used standard data augmentation techniques. In the first part of the experiments, the standard augmentation techniques were selected such as:

(i) For each image in the database, we have generated 6 additional augmented images by rotating images by a small angle from -3 to 3 degrees and adding those images to our training sets. Fig.23 shows a word image from the IFN/ENIT database rotated at different angles (-3 to 3 degrees)

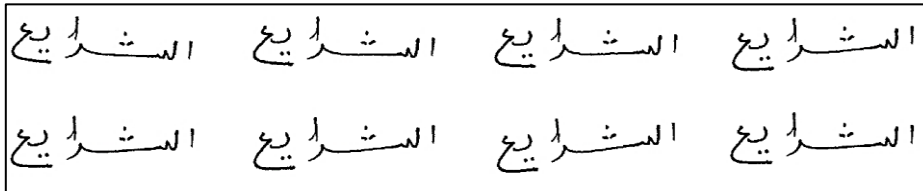


Figure 23. A word taken from the IFN/ENIT database rotated at different angles (-3 to 3 degrees).

(ii) For each of the original images in the database, we generated five additional augmented images using shear transformation. Fig.23. shows a word image from the IFN/ENIT sheared with a different shearing coefficient.

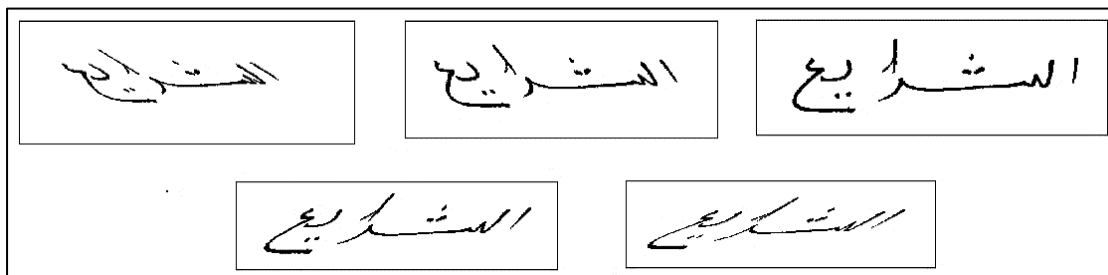


Figure 24. Shearing a word in the database with different shearing factor.

5.4 Experiment Configurations

We have compared the performance of the RNN-LSTM network to CNN (AlexNet [86]) holistically. The AlexNet network consisted of five convolutional layers, a max-pooling layer, a dropout layer, and three fully connected layers. After each convolutional and fully connected layer, ReLU is applied. Before the first and second fully connected layer dropout is applied. The receptive field of each convolution layer is 3x3. The fifth convolution layer is followed by three fully connected layers, with the first two having 512 channels each. The AlexNet was originally designed to identify 1000 classes. We have replaced the last fully connected layer by a new fully-connected layer with as many outputs as the number of classes (937 for the IFN/ENIT database and 96 for the AHDB database). Due to the high training time associated with such a large network, we have used transfer learning method to retrain the pretrained AlexNet[101].

The architecture of the LSTM network for classification was chosen such that it starts with a sequence input layer followed by an LSTM layer. The LSTM layer is followed by a fully connected layer to predict class labels, a SoftMax layer, and a classification output layer. The SoftMax layer is connected to the CTC-token passing decoder. Later, the CTC layer is modified to adopt the word beam search algorithm [94]. The LSTM was composed 200 LSTM blocks, separated by a subsampling layer of 100 hidden neurons without bias, and with an hyperbolic tangent activation function.

The BLSTM recognizer was identical to the one used in (Graves [50]). Thus, we have used the Recurrent Neural Network Library (RNNLIB) for sequence learning problems which is forked from Alex Graves's work [102]. This library provides 1D and multidimensional LSTM network with ground truth alignment using the CTC decoder.

It also provides a heuristic decoding process for mapping the output of the network to a sequence of symbols. The BLSTM hidden layers were fully connected to themselves, and the input and output layers. Each memory block contained a single LSTM cell. The sizes of the input and output layers were determined by the numbers of inputs and labels in each task. Our purposed network has 200 extended LSTM memory blocks in each of the forward and backward layers. Each memory block has one memory cell, an input gate, an output gate, a forget gate, and three peephole connections. The cell input and output activation functions are *tanh* while the gate activation function is logistic sigmoid. A decaying learning rate has been used in all the CTC systems. The initial learning rate was set to 0.01 and was reduced gradually to 0.0001.

The weights were randomly initialized from a Gaussian distribution with zero mean and a standard deviation of 0.1. For all the experiments reported in this paper, the IFN/ENIT database and AHDB were used to train and test all the systems. MATLAB [103] was used to pre-process the images, and experimental models were implemented using TensorFlow [104] and OpenCV [105].

5.5 Performance Evaluation

In this section, we present the experiments we conducted, the results we obtained, and the resulting discussions of all systems presented previously. The performance of all handwriting recognition systems was evaluated using two metrics.

1. **Word Accuracy Rate (WAR):** Word percentage correctly recognized with all characters.

$$WAR(\%) = \left[1 - \frac{S+I+D}{N} \right] \times 100 \quad (19)$$

where:

S , is the total number of substituted words.

I , is the total number of inserted words.

D , is the total number of deleted words.

N , is the total number of words in the evaluation set.

After aligning the recognized word string against the transcription, the number of words replaced, inserted, and deleted is counted. Alignment is known as the problem of maximum substring matching [74,75].

2. **Character Accuracy Rate (CAR):** The percentage of characters that have been correctly recognized in all words. This rate is usually measured using the edit distance method. It is defined as the minimum number of insertions, deletions, or substitutions of a single terminal needed to transform other of the strings into the other one.

$$CAR(\%) = \left[1 - \frac{S+I+D}{N} \right] \times 100 \quad (20)$$

where:

S , is the total number of substituted characters.

I , is the total number of inserted characters.

D , is the total number of deleted characters.

N , is the total number of characters in the evaluation set.

After aligning the recognized character string against the transcription, the number of characters substituted, inserted, and deleted is counted [74,75].

5.6 Results of Holistic Handwriting Recognition.

Table.6 shows the results of AlexNet and RNN's holistic recognition on the IFN/ENIT and AHDB databases. Both of the networks were trained using the same configuration described in Section 5.4 to evaluate their performances similarly. Both of the systems were trained and tested using similar train-test configurations and both of the networks were trained for 1,000 epochs. The overall difference in performance between the two networks was negligible. The AlexNet network was able to achieve 90.29% compared to 89.13% WAR on the IFN/ENIT database without using any augmented data. On the AHDB database was able to achieve 92.97% compared to 91.30% WAR without using any augmented data. Conversely, standard neural augmentation performs remarkably better than no augmentation. In AlexNet, the neural augmentation performed 95.27% to 90.29% compared to no augmentation on the IFN/ENIT database while it performed 96.80% to 92.97% on the AHDB database. the RNN with neural augmentation performed the second-best, where the word accuracy rate increased from 89.13% to 95.77%. on the IFN/ENIT database while it performed 96.69% to 92.97% on the AHDB database.

Table 6. AlexNet vs. RNN (Holistic Recognition).

Database	Train-test configurations	ALEXNET	RNN (LSTM)	ALEXNET	RNN (LSTM)
	Augmented Data	×	×	√	√
	(Standard)				
		WAR %	WAR %	WAR %	WAR %
IFN/ENIT	abc-d	90.29%	89.13%	95.27%	95.77%
	bcd-a	86.34%	85.66%	92.17%	93.29%
	abcd-e	83.12%	82.16%	89.39%	90.16%
	abcde-f	71.43%	73.67%	83.92%	81.66%
AHDB	70% training 20% testing 10% validation	94.97%	93.30%	96.80%	96.69%

5.7 Results of Arabic Handwriting Recognition Using Characters Shapes as Models.

In this section, we shall present the results obtained for word recognition tasks using character shapes as modeling units. The IFN/ ENIT database contains the transcriptions for the word image at character-shape level and we have created similar transcriptions for the AHDB database. We ended up with a total of 157 models in our recognition system. Table.7 shows the accuracy rate of each system evaluated on the IFN/ENIT and AHDB database without augmenting any new data.

**Table 7. Accuracy Rates of the Systems using character shapes as models (LSTM)
(No Augmented data)**

Database	Train-test configuration	LSTM-CTC (Token Passing).		LSTM-CTC (WBS-Decoder).	
		CAR %	WAR %	CAR %	WAR %
IFN/ENIT	abc-d	95.23%	94.12%	96.17%	94.83%
	bcd-a	90.14%	89.97%	93.42%	92.72%
	abcd-e	88.29%	86.13%	89.94%	88.91%
	abcde-f	77.41%	75.17%	79.12%	81.86%
AHDB	70% training 20% testing 10% validation	91.27%	90.21%	92.89%	93.49%

Table.8 shows the accuracy rate when using augmented data. The results show that by using standard data augmentation techniques, the accuracy rate is improved. The maximum CAR obtained on the IFN/ENIT database was 97.17% while the maximum WAR on the IFN/ENIT database was 96.22%. The maximum CAR obtained on the AHDB database was 96.77 while the maximum WAR was 95.99 %. In this experiment we have used the LSTM-CTC architecture with the WBS decoder explained in Sec.5.5, Clearly, by combining the CTC layer with the WBS decoder, the network gave much better results.

**Table 8. Accuracy Rates of the Systems using characters' shapes as models (LSTM)
(Using Standard Data Augmentation)**

Database	Train-test configuration	LSTM-CTC (Token Passing).		LSTM-CTC (WBS-Decoder).	
		CAR %	WAR %	CAR %	WAR %
IFN/ENIT	abc-d	96.03%	95.39%	97.17%	96.22%
	bcd-a	93.16%	92.29%	94.18%	93.17%
	abcd-e	90.43%	88.19%	91.24%	90.14%
	abcde-f	87.33%	85.29%	89.14%	87.94%
AHDB	70% training 20% testing 10% validation	94.57%	94.11%	96.77%	95.99%

A further improvement of the current system can be achieved if we replace the LSTM layer by the Bi-directional LSTM (BLSTM) layer. The use of BLSTM can further improve the network. In the BLSTM layer, the forward layer learns the context of the input by processing the sequence from input to output, while the backward layer performs the opposite operation by processing the sequence from output to input. Table.9 shows the results of using the BLSTM network. We can see that the accuracy rates have improved significantly compared to LSTM. In sporadic cases when using BLSTM the improvement of the results can be very limited, this is can be explained in Lee et al. [106] by the prevalence of short words within the dataset but in general, replacing the LSTM layer with Bi-directional Long Short-Term Memory (BLSTM) enables higher recognition rates to be achieved.

**Table 9. Accuracy Rates of the Systems using characters' shapes as models (BLSTM)
(Using Standard Data Augmentation)**

Database	Train-test configuration	BLSTM-CTC (Token Passing).		BLSTM-CTC-WBS (Word Beam Search).	
		CAR %	WAR %	CAR %	WAR %
IFN/ENIT	abc-d	97.23%	96.82%	98.16%	97.70%
	bcd-a	95.16%	94.19%	94.98%	93.20%
	abcd-e	92.13%	91.95%	94.14%	93.75%
	abcde-f	91.23%	90.20%	92.33%	91.92%
AHDB	70% training 20% testing 10% validation	96.72%	95.41%	97.47%	97.09%

5.8 Results of Arabic Handwriting Recognition Using Characters as Models.

In this section, we shall present the results of Arabic handwriting recognition using Arabic characters as modeling units. The transcriptions for the word image were modified to character level for both databases. We ended up with a total of 46 models in our recognition system. Table 10 shows the handwriting recognition accuracy rates for standard train-test configurations without using any augmented data. Table 11 shows the handwriting recognition accuracy rates when using augmented data generated by standard techniques. In all experiments, the accuracy of handwriting recognition was evaluated by using the BLSTM network combined with the CTC layer.

**Table 10. Accuracy Rates of the Systems using characters as models (BLSTM)
(No Augmented data)**

Database	Train-test configuration	BLSTM-CTC (Token Passing).		BLSTM-CTC (Word Beam Search).	
		CAR %	WAR %	CAR %	WAR %
IFN/ENIT	abc-d	85.33%	81.15%	86.70%	83.90%
	bcd-a	84.91%	79.77%	82.78%	81.19%
	abcd-e	80.13%	77.11%	81.84%	80.72%
	abcde-f	74.31%	72.61%	78.10%	77.20%
AHDB	70% training 20% testing 10% validation	80.92%%	79.71%	82.74%	81.24%

**Table 11. Accuracy Rates of the Systems using characters as models (BLSTM)
(Using Standard Data Augmentation)**

Database	Train-test configuration	BLSTM-CTC (Token Passing).		BLSTM-CTC-WBS (Word Beam Search).	
		CAR %	WAR %	CAR %	WAR %
IFN/ENIT	abc-d	91.77%	89.25%	93.10%	91.19%
	bcd-a	89.85%	86.17%	91.41%	89.47%
	abcd-e	85.93%	83.91%	88.16%	86.15%
	abcde-f	81.74%	79.74%	84.20%	82.55%
AHDB	70% training 20% testing 10% validation	85.66%	84.42%	87.74%	86.52%

As we mentioned earlier, it is not the best choice to use characters as models because Arabic characters have different position-dependent shapes. The accuracy rates of using characters as models are worse than the one obtained using characters shapes as models. This because many characters have a similar shape, e.g., س, ش, ص, ض, which

makes the task of the system in determining the real character more difficult. Besides, the shapes of the characters change according to their position in the word, which causes more confusion for the network for determining the correct class.

5.9 Results of Arabic Handwriting Recognition Using a Multi-stage Recognition System (Modeling Core-shapes and Diacritics Separately).

We have constructed two LSTM recognizers, one for the core-shapes and the other for diacritics. Each network is configured as in Section 5.4. First, for each image on the IFN/ENIT database we used the algorithm explained in Fig.10 to separate the input image into two components. Fig.25 shows the results of separating core-shape and diacritics.

Core-shape	Diacritics

Figure 25. Separating core-shapes and diacritics.

We used standard train-test configurations for the first set of experiments. We have used *abc-d* as the system development train-test set configuration. The core shape system has 71 outputs to model all the characters and their shape variation. The diacritics system has 17 output to model dots and other diacritics e.g. (*Shadda* and *Hamza*). The core-shape system was trained such that it can generate an N-best list of

words based on core-shape recognition utilizing a dictionary that has entry pairs for word and its corresponding core-shape definition. The diacritic system was trained on diacritics. Finally, the word hypothesis was made using Viterbi decoding. Table.12 shows, in comparison to other modelling choices, that the multi stage system is quite effective.

Table 12. Accuracy Rates of the Systems using Multi-stage recognition system (modeling core-shapes and diacritics separately). (Using Standard Data Augmentation)

Database	Train-test configuration	MULTI-STAGE SYSTEM
		WAR %
IFN/ENIT	<i>abc-d</i>	93.75%
	<i>bcd-a</i>	90.13%
	<i>abcd-e</i>	86.82%
	<i>abcde-f</i>	82.22%

5.10 Results of Using Adaptive Data Augmentation Technique

Let us look at the statistics of the IFN/ENIT database. Table 13 shows the number of occurrences of each character in the database along with its p_n . We can see that the distribution is highly skewed. By looking at the table, we can notice that there is a huge difference between the appearance of some characters in the database. For example, the letter Alif (ا) is repeated more than 36 thousand times, compared to the letter Za (ظ), which we can barely see in the database. By knowing the characters sequence of each word in the database, we can calculate W_i , e.g., the word (الشرايع) has seven characters with each character has a certain p_n so:

$$W_{\text{الشرايع}} = \frac{\frac{1}{p_{\text{Alif}}} + \frac{1}{p_{\text{Laam}}} + \frac{1}{p_{\text{Shin}}} + \frac{1}{p_{\text{Raa}}} + \frac{1}{p_{\text{Alif}}} + \frac{1}{p_{\text{Yaa}}} + \frac{1}{p_{\text{Ain}}}}{7}$$

$$W_{\text{الشرايع}} = \frac{\frac{1}{0.17} + \frac{1}{0.092} + \frac{1}{0.0156} + \frac{1}{0.074} + \frac{1}{0.17} + \frac{1}{0.083} + \frac{1}{0.023}}{7} = 22.253$$

Suppose we decide to augment new 100,000 samples to the original database. We need to decide how many samples should we augment the word “الشرايع” out of the 100,000 so:

$$\text{Total No. of augmented samples} = [W_{i, \text{normalized}} \times 100,000].$$

$$= \frac{W_{\text{الشرايع}}}{\sum W_i \text{ (sum of all classes weights in the database)}} \times 100,000$$

$$= \frac{22.253}{\sum W_i = 15900} \times 100,000 = 140 \text{ samples.}$$

where, $\sum W_i = 15900$ (for IFN/ENIT)

Table 13. IFN/ENIT database [57] statistics.

Character	No. of Occurrence	p_n	Character	No. of Occurrence	p_n
0	342	0.162	د	8657	4.106
1	279	0.132	ذ	1056	0.500
2	384	0.182	ر	15621	7.410
6	311	0.147	ز	4482	2.126
7	354	0.167	س	7012	3.326
8	284	0.134	ش	3270	1.551
9	341	0.161	ص	2764	1.311
ء	520	0.246	ض	1752	0.831
ا	36394	17.265	ط	2310	1.095
لا	2357	1.118	ظ	1029	0.488
ب	9718	4.610	ع	4902	2.325
ة	7259	3.443	غ	926	0.439
ت	1402	0.665	ف	3180	1.508
ث	1018	0.482	ق	9660	4.582
ج	1564	0.741	ك	2536	1.203
لج	539	0.255	ل	19475	9.238
ح	4896	2.322	م	9938	4.714
لح	365	0.173	لم	458	0.217
لمح	64	0.030	ن	10021	4.753
نح	100	0.047	ه	3318	1.574
خ	1640	0.778	و	10040	4.762
لخ	310	0.147	ي	17591	8.345

So, out of the new 100,000 augmented samples, 140 augmented samples will correspond to the word “الشرايع”. For all the words in the lexicon, this is repeated to find the number of increased samples for each class. This will finally balance the database. Note that it is up to the reader to choose any technique to generate the new augmented

data, e.g. we used a combination of standard data increase techniques (rotation and shear) to generate new 140 samples. Fig.26 shows the frequency distribution of the IFN/ENIT characters after using our adaptive data augmentation algorithm compared to using the standard data augmentation technique

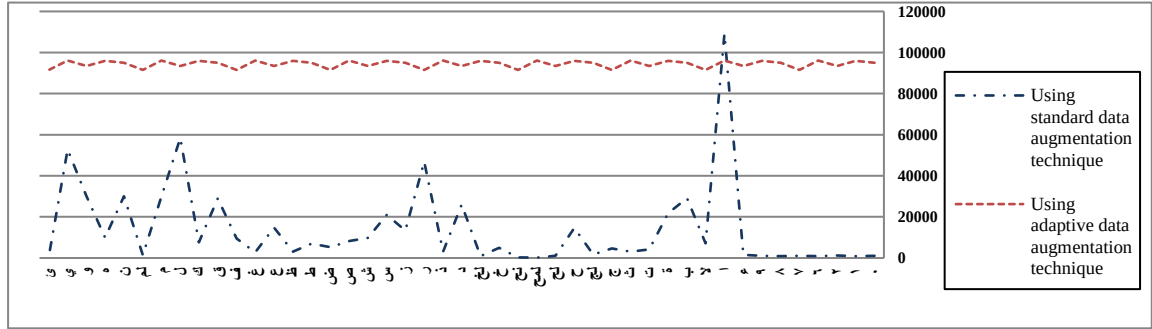


Figure 26. Adaptive vs. standard data augmentation (IFN/ENIT Database).

If we look at the statistics of the AHDB database. It contains approximately 10,000 Arabic handwritten words and approximately 31,200 characters corresponding to numbers that can be used in handwritten cheque writing and the most popular Arabic words. If the recognition system uses characters as modeling units then Table 14 shows the number of occurrences of each character in the database along with its p_n .

Table 14. AHDB [58] statistics.

Character	No. of Occurrences	p_n	Character	No. of Occurrence	p_n
ا	4400	0.141	ف	500	0.016
ب	1200	0.039	ق	400	0.013
ك	1300	0.042	ك	400	0.013
ث	1300	0.042	ل	1300	0.042
ح	300	0.01	م	2800	0.090
خ	700	0.022	ن	3100	0.1
د	300	0.01	ه	400	0.013
ذ	300	0.01	و	1300	0.042
ر	1400	0.045	ي	1500	0.048
س	2500	0.080	لا	600	0.019
ش	400	0.013	ة	2200	0.071
ظ	200	0.006	ى	300	0.01
ع	1300	0.042			

Unlike the previous database, this database lacks many Arabic characters because it was specially prepared to identify the famous words in the Arabic language and the words used in writing checks. By knowing the characters sequence of each word in the database, we can calculate W_i , e.g., the word (ريال) has four characters with each character has a certain P_n so:

$$W_{ريال} = \frac{\frac{1}{p_{Raa}} + \frac{1}{p_{Yaa}} + \frac{1}{p_{Alif}} + \frac{1}{p_{Laam}}}{4} = \frac{\frac{1}{0.045} + \frac{1}{0.048} + \frac{1}{0.141} + \frac{1}{0.042}}{4} = 18.48$$

If we want to augment e.g. 100,000 samples to the original database. We need to decide how many samples should we augment for each word in the lexicon e.g. for the word “ريال”:

$$[W_{i,normalized} \times 100,000] = \text{total No. of augmented samples.}$$

$$\begin{aligned} &= \frac{W_{ريال}}{\sum W_i \text{ (sum of all classes wieghts in the database)}} \times 100,000 \\ &= \frac{18.48}{1923} \times 100,000 = 960 \text{ samples. where } \sum W_i = 1923 \text{ (for AHDB)} \end{aligned}$$

So, out of the new 100,000 augmented samples, 960 augmented samples will correspond to the word “ريال”. We repeat this for all the words in the lexicon to find the number of augmented samples for each class. This eventually will balance the given database. Note that it is up to the reader to choose any technique to generate new augmented data, e.g. we used a combination of classical data increase techniques (rotation and shear) to generate new 960 samples. Fig.27 shows the frequency distribution of the characters after using our intelligent data augmentation algorithm compared to the conventional data augmentation technique on the AHDB database.

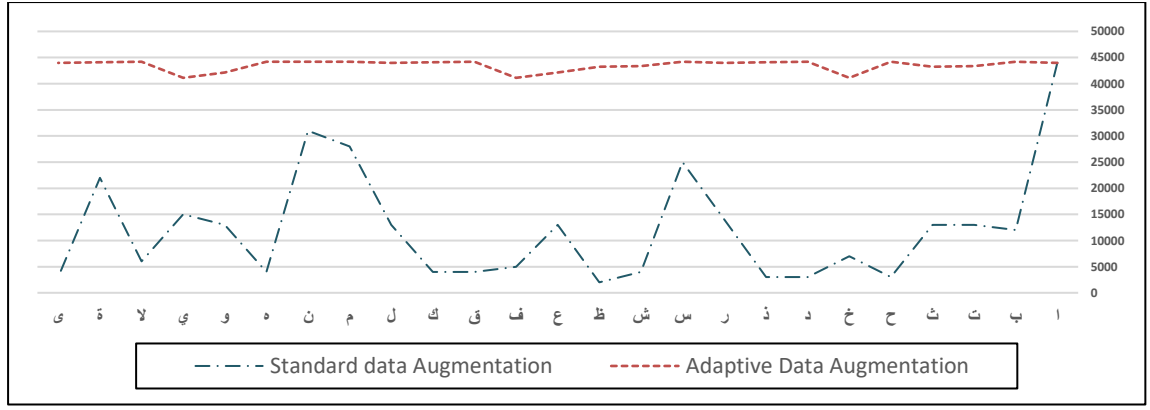


Figure 27. Adaptive vs. standard data augmentation (AHDB database).

Table 15 shows the results of AlexNet and RNN's holistic recognition on the IFN/ENIT and AHDB database. the system was trained using the same configuration described in Section 5.4 to evaluate their performances similarly. Both of the systems were trained this time by using our adaptive data augmentation algorithm described in Chapter.5. Table 15 shows that recognition rates are significantly improved. The system performs better in all training test configurations with balanced data obtained by our algorithm.

Table 15. ALEXNET VS. RNN (HOLISTIC RECOGNITION)
(Using Adaptive Data Augmentation).

Database	Train-test configurations	ALEXNET	RNN (LSTM)
		WAR %	WAR %
IFN/ENIT	abc-d	97.16%	96.91%
	bcd-a	95.85%	95.05%
	abcd-e	92.47%	91.87%
	abcde-f	91.77%	88.46%
AHDB	70% training 20% testing 10% validation	95.07%	93.87%

Now, let us consider using the balanced data resampled using our algorithm in Section 4.2 as training data for the RNN-BLSTM system using characters as models and by applying the same preprocessing and configurations steps used in Sections 5.3 and 5.4. Table 16 shows the results of handwriting recognition accuracy rates on the IFN/ENIT and AHDB databases augmented with our data augmentation algorithm. The

results show that our data augmentation technique improves CAR% and the WAR% in all train-test configurations.

**Table 16. Accuracy Rates of the Systems using characters as models (BLSTM)
(Using Adaptive Data Augmentation)**

	Train-test configuration	BLSTM-CTC-WBS.	
		CAR %	WAR %
AHDB	abc-d	95.19%	96.19%
	bcd-a	93.33%	93.11%
	abcd-e	91.21%	91.96%
	abcde-f	90.90%	89.15%
	70% training	89.67%	88.41%
	20% testing 10% validation		

The same experiment was repeated, but this time we used the character shapes as modeling units. Table 17 shows the accuracy rates of the system on all possible train-test configurations. We can see from the table that there is a small but significant improvement in the recognition rates. The system performs better with the balanced data resampled by our algorithm in all train-test configurations.

**Table 17. Accuracy Rates of the Systems using character shapes as models (BLSTM)
(using Adaptive data augmentation)**

database	Train-test configuration	BLSTM-CTC-WBS.	
		CAR %	WAR %
AHDB	abc-d	99.73%	98.99%
	bcd-a	97.66%	96.91%
	abcd-e	96.23%	95.05%
	abcde-f	94.13%	93.57%
	70% training	98.49%	98.10%
	20% testing 10% validation		

5.11 Comparison with Other State-of-the-Art Systems.

In this section, we compare the results of recent state-of-the-art systems evaluated on the IFN/ENIT and AHDB databases with the results obtained by our systems. Table 18 presents a comparison of the results from the state-of-the-art systems evaluated on the IFN/ENIT and AHDB databases.

**Table 18. Comparison with Other State-of-the-Art Systems
Evaluated on The IFN/ENIT and AHDB Databases**

Author	Word Accuracy Rate (WAR)		
	The IFN/ENIT Database		
	Train-Test Configuration		
	<i>abc-d</i>	<i>abcd-e</i>	<i>abcde-f</i>
Al-Hajj, et al. [66]	90.96%	-	-
Dreuw et al. [67]	94.18%	88.78%	-
Graves et al. [50]	-	-	93.37%
Kessentini et al. [68]	-	79.69%	-
Pechwitz et al. [57]	91.80%	-	-
Rothacker et al. [69]	-	92.09%	-
Azeem et al. [70]	97.70%	93.44%	93.10%
Abandah et al. [71]	98.96%	93.46%	92.46%
Hamdani et al. [72]	-	-	92.20%
Elleuch et al. [73]	83.70%	-	-
Stahlberg et al. [74]	97.60%	93.09%	-
Ahmad et al. [75]	96.07%	94.93%	92.30%
Rabi et al. [80]	87.93%	-	-
Almodfer et al. [81]	92.55%	-	-
Ahmad et al. [76]	97.71%	94.76%	93.32%
Almodfer et al. [82]	-	-	91.50%
Present work BLSTM-CTC-WBS. (using adaptive data augmentation)	98.99%	95.05%	93.57%

Author	Word Accuracy Rate (WAR)
	The AHDB Database
S. Al-Ma'adeed [58]	45.00%
S. Al-Ma'adeed et al. [59]	63.00%
Al-Nuzaili et al. [79]	89.29%
Hassan et al. [77]	99.08%
Hassan et al. [78]	96.32%
Present work BLSTM-CTC-WBS. (using adaptive data augmentation)	98.10%

Many works are carried out on handwritten Arabic text recognition using the IFN/ENIT and AHDB database. The best-reported systems in the literature on the evaluation sets (e and f) of the IFN/ENIT database were Ahmed et al. [75] and Graves

et al. [50]. On the evaluation of set (d) Abandah et al. [71] were the best reported system. The overall results of our systems are among the best in comparison with the other state-of-the-art system. It's also clear that our system integrated with adaptive data augmentation technique has the highest accuracy rate on the IFN/ENIT database. In fact, our systems outperforms the best reported systems in the literature on the evaluation set e and set f of the database.

For the AHDB database, the highest accuracy obtained were Hassan et al [77]. The authors proposed a handwritten Arabic recognition system that use a bag of features with Support Vector Machine (SVM). Their system was evaluated using AHDB database with 2072 images for training set and 868 for testing set while our system was implemented by using 7057 images for training and 2017 images for testing which explains why they obtained a higher accuracy rate than our system. Also, the other reason for obtaining higher accuracy from us is that SVM typically perform better on small datasets [95].

CONCLUSION

The automatic Arabic handwriting recognition is still a challenging research area in the field of pattern recognition. Deep Neural Networks (DLNN) has gained a reputation for taking care of numerous computer vision issues. Its application to the field of handwriting recognition has been shown to give fundamentally excellent results over conventional techniques. In this work, we experimented with four different modeling approaches. Three of them being the standard modeling approaches as reported in the literature and the remaining one was proposed by Ahmed et al. [75]. Based on the experiments conducted using Arabic character shapes as modeling unit is the best choice among the four options although it increases the number of output classes to approximately 157 models.

Besides, we performed extensive experiments using different deep learning architectures to recognize handwritten Arabic text. Specifically, we used CNN, RNN, LSTM, Bi-LSTM, and CTC. Each system was evaluated by using the IFN/ENIT and AHDB databases. We can see from Table.18 that the overall results from our presented systems are among the best compared to the other state-of-the-art systems.

Besides, we have proposed an adaptive data-augmentation algorithm that also attempts to deal with the issue of imbalanced classes. It has been shown that this method can solve rather effectively the data imbalance problem. Our purposed method is performed in such a way that the minority classes will be expanded by calculating the average probability of each class. The results show that the presented algorithm, when applied to the IFN/ENIT and AHDB databases can balance the distribution of the minority classes. The presented system in this work is among the best state-of-the-art systems.

While all systems have produced excellent results, the accuracy rates were outperformed by using the CTC layer integrated with Word Beam Search (WBS) decoder. During the tests, we have noticed that AlexNet and LSTM tend to over-fit and do not generalize well to the test data. Besides, the BLSTM system is robust to overfit and has higher accuracy, especially to previously unseen data. During the training stage, experiments showed that augmenting new samples increases accuracy rates and prevent over-fitting, but it increases the training time of the network. Besides, using image transformations, e.g., vertical alignment of characters and removal of unnecessary white-space at the top and bottom of word images, can be used as some extra preprocessing steps.

As future works, we plan to use more advanced methods to enhance the current work, e.g., recursive recurrent nets with attention modeling [106], multi-dimensional LSTM network [107] and using other modeling options such as using sub-character and sub-core-shape as modeling units [76] which contributes to significant reductions in the number of models to be trained for the task of text recognition. Another observation on the IFN/ENIT we can see the decrease in performance on set f. One of the reasons being that it was collected in a different environment and was added later to the dataset and is regarded as a relatively difficult set.

References

- [1] Berchmans, D. and Kumar, S.S., 2014, July. Optical character recognition: an overview and an insight. In 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT) (pp. 1361-1365). IEEE.
- [2] Nashif, M.H.H., Miah, M.B.A., Habib, A., Moulik, A.C., Islam, M.S., Zakareya, M., Ullah, A., Rahman, M.A. and Al Hasan, M., 2018. Handwritten Numeric and Alphabetic Character Recognition and Signature Verification Using Neural Network. *Journal of Information Security*, 9(3), pp.209-224.
- [3] Shi, B., Yang, M., Wang, X., Lyu, P., Yao, C. and Bai, X., 2018. Aster: An attentional scene text recognizer with flexible rectification. *IEEE transactions on pattern analysis and machine intelligence*, 41(9), pp.2035-2048.
- [4] *Ethnologies: Languages of the World*, 22nd Edition: SIL International. Released on February 21st, 2019.
- [5] Boufenar, C., Kerboua, A. and Batouche, M., 2018. Investigation on deep learning for off-line handwritten Arabic character recognition. *Cognitive Systems Research*, 50, pp.180-195.
- [6] Ali, A.A.A. and Suresha, M., 2019, March. An efficient character segmentation algorithm for recognition of Arabic handwritten script. In 2019 International Conference on Data Science and Communication (IconDSC) (pp. 1-6). IEEE.
- [7] Suresha, M. and Ali, A.A.A., 2018. Segmentation of handwritten text lines with touching of line. *International Journal of Computer Engineering and Applications*, 12(6), pp.1-12.
- [8] Al-Ayyoub, M., Nuseir, A., Alsmearat, K., Jararweh, Y. and Gupta, B., 2018. Deep learning for Arabic NLP: A survey. *Journal of computational science*, 26, pp.522-531.
- [9] Vásquez, J.L., Ravelo-García, A.G., Alonso, J.B., Dutta, M.K. and Travieso, C.M., 2018. Writer identification approach by holistic graphometric features using off-line handwritten words. *Neural Computing and Applications*, pp.1-14.
- [10] Nashwan, F., Rashwan, M.A., Al-Barhamtoshy, H.M., Abdou, S.M. and Moussa, A.M., 2018. A holistic technique for an Arabic OCR system. *Journal of Imaging*, 2018,4, 6; doi:10.3390/jimaging4010006 pp.1-11.
- [11] Graves, Alex, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks." In *Proceedings of the 23rd international conference on Machine learning*, pp. 369-376. ACM, 2006.
- [12] Maalej, R. and Kherallah, M., 2018, November. Convolutional neural network and BLSTM for offline Arabic handwriting recognition. In 2018 International Arab conference on information technology (ACIT) (pp. 1-6). IEEE.

- [13] Jemni, S.K., Kessentini, Y., Kanoun, S. and Ogier, J.M., 2018, April. Offline Arabic handwriting recognition using BLSTMs combination. In 2018 13th IAPR International Workshop on Document Analysis Systems (DAS) (pp. 31-36). IEEE.
- [14] Graves, Alex, Marcus Liwicki, Horst Bunke, Jürgen Schmidhuber, and Santiago Fernández. "Unconstrained on-line handwriting recognition with recurrent neural networks." In *Advances in neural information processing systems*, pp. 577-584. 2008.
- [15] Yogatama, D., Dyer, C., Ling, W. and Blunsom, P., 2017. Generative and discriminative text classification with recurrent neural networks. arXiv preprint arXiv:1703.01898.
- [16] Ahmad, Irfan, and Sabri A. Mahmoud. "Arabic bank check processing: state of the art." *Journal of Computer Science and Technology* 28, no. 2 (2013): 285-299.
- [17] Al-Jallad, Ahmad. "Stefan Weninger in collaboration with Geoffrey Khan, Michael P. Streck and Janet CE Watson (eds), *The Semitic Languages: An International Handbook*." (2013): 395-398
- [18] Fu, King-Sun. *Applications of pattern recognition*. CRC Press, 2019.
- [19] Shamir, Lior, John D. Delaney, Nikita Orlov, D. Mark Eckley, and Ilya G. Goldberg. "Pattern recognition software and techniques for biological-image analysis." *PLoS computational biology* 6, no. 11 (2010).
- [20] Watanabe, Satoshi, ed. *Methodologies of pattern recognition*. Academic Press, 2014.
- [21] Chaudhuri, Arindam, Krupa Mandaviya, Pratixa Badelia, and Soumya K. Ghosh. "Optical character recognition systems." In *Optical Character Recognition Systems for Different Languages with Soft Computing*, pp. 9-41. Springer, Cham, 2017.
- [22] Eikvil, Line. "OCR-optical character recognition." (1993).
- [23] N. Arica; F.T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," *Systems, Man, and Cybernetics, Part C: Applications and Reviews*, IEEE Transactions on, vol.31, no.2, pp.216-233, May 2001.
- [24] G. Nagy, "At the frontiers of OCR," *Proceedings of the IEEE*, vol.80, no.7, pp.1093-1100, Jul 1992.
- [25] Jisheng Liang; I.T Phillips.; V. Chalana; R. Haralick, "A methodology for special symbol recognition," *Pattern Recognition*, 2000. *Proceedings. 15th International Conference on*, vol.4, no., pp.11-14 vol.4, 2000.
- [26] S. Mori; C.Y. Suen; K. Yamamoto, "Historical review of OCR research and development," *Proceedings of the IEEE*, vol.80, no.7, pp.1029-1058, Jul 1992.

- [27] Smith R., "An Overview of the Tesseract OCR Engine," Document Analysis and recognition, 2007. ICDAR 2007. Ninth International Conference on, vol.2, no., pp.629-633, 23-26 Sept. 2007.
- [28] El Abed, Haikal & Märgner, Volker. (2007). Comparison of Different Preprocessing and Feature Extraction Methods for Offline Recognition of Handwritten Arabic words. 2. 974-978. 10.1109/ICDAR.2007.4377060.
- [29] H. Al-Rashaideh, "Preprocessing Phase for Arabic Word Handwritten Recognition", Russian Academy of Sciences, Russian Federation, vol. 6, no. 1, (2006), pp. 11-19.
- [30] F. Farooq, V. Govindaraju and M. Perrone, "Pre-processing Methods for Handwritten Arabic Documents", In Eighth International Conference on Document Analysis and Recognition, vol. 1, (2005), pp. 267–271.
- [31] A. Mowlaei, K. Faez, and A. T. Haghighat, Feature Extraction with Wavelet Transforms for Recognition of Isolated Handwritten Farsi/Arabic Characters and Numerals", In 14th International Conference on Digital Signal Processing, vol. 2, (2002), pp. 923–926.
- [32] A. Cheung, M. Bennamoun and N. W. Bergmann, "A recognition-based Arabic Optical Character Recognition System", In IEEE International Conference on Systems, Man, and Cybernetics, vol. 5, (1998), pp. 4189 –4194.
- [33] S. N. Nawaz, M. Sarfraz, A. Zidouri and W. G. Al-Khatib, "An Approach to Offline Arabic Character Recognition Using Neural Networks", In 10th IEEE International Conference on Electronics, Circuits and Systems, vol. 3, (2003), pp. 1328– 1331.
- [34] M. M. Altuwaijri and M. A. Bayoumi, "Arabic Text Recognition Using Neural Networks, In IEEE international Symposium on Circuits and Systems, vol. 6, (1994), pp. 415–418.
- [35] M. Khorsheed, "Off-line Arabic Character Recognition - A Review, Pattern Analysis Applications, vol. 5, (2002), pp. 31–45.
- [36] S. Al-Ma'adeed, "Recognition of Off-line Handwritten Arabic Words", PhD Thesis, University of Nottingham, (2004).
- [37] V. K. Govindan and A. P. Shivaprasad, "Character Recognition A Review", Pattern Recognition, vol. 23, no.7, (1990), pp. 671–683.
- [38] A. Amin and J. Mari, "Machine Recognition and Correction of Printed Arabic Text", IEEE Transactions on Systems, Man and Cybernetics, vol. 19, no. 5, (1989), pp. 1300 –1306.

- [39] M. S. Khorsheed, "W. F. Structural Features of Cursive Arabic Script", In Proceedings of The Tenth British Machine Vision Conference, (1999), pp. 442 - 431.
- [40] G. N. Srinivasan and G. Shobha, "Statistical Texture Analysis", In Proceedings of World Academy of Science, Engineering and Technology, vol. 36, (2008), pp. 1264–1269.
- [41] K. M. Mohiuddin and J. Mao, "A Comparative Study of Different Classifiers for Handprinted Character Recognition, In Pattern Recognition in Practice IV, (1994), pp. 437–448.
- [42] M. S. Khorsheed and W. F. Clocksin, "Multi-font Arabic Word Recognition Using Spectral Features", In 15th International Conference on Pattern Recognition, vol. 4, (2000), pp. 543–546.
- [43] J. H. AlKhateeb, R. Jinchang, J. Jianmin, S. S. Ipson and H. El-Abed, "Word-based Handwritten Arabic Scripts Recognition using DCT Features and Neural network Classifier", In 5th International Multi-Conference on Systems, Signals, and Devices, (2008), pp. 1–5.
- [44] I. A. Jannoud, "Automatic Arabic Handwritten Text Recognition System", American Journal of Applied Sciences, vol. 4, (2007), pp. 857–864.
- [45] N. Ben Amor and N. E. Ben Amara, "Multifont Arabic Characters Recognition Using Hough Transform and HMM/ANN Classification", Journal of Multimedia, vol. 1, no. 2, (2006), pp. 50–54.
- [46] Lawgali, Ahmed. "Recognition of Handwritten Digits using Histogram of Oriented Gradients." (2016).
- [47] Nawaz SN, Sarfraz M, Zidouri A, Al-Khatib WG. An approach to offline Arabic character recognition using neural networks. In 10th IEEE International Conference on Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003 2003 Dec 14 (Vol. 3, pp. 1328-1331). IEEE.
- [48] A. Rehman, D. Mohamad, and G. Sulong "Implicit Vs Explicit based script segmentation and recognition: A performance comparison on benchmark database," International Journal Open Problems Compt. Math., vol. 2, pp. 352-364, 2009.
- [49] A. Choudhary, "A review of various character segmentation techniques for cursive handwritten words recognition," International Journal of Information and Computation Technology, vol. 4, pp. 559-564, 2014.
- [50] Graves, Alex, and Jürgen Schmidhuber. "Offline handwriting recognition with multidimensional recurrent neural networks." In Advances in neural information processing systems, pp. 545-552. 2009.

- [51] Elleuch, Mohamed, Najiba Tagougui, and Monji Kherallah. "Deep learning for feature extraction of Arabic handwritten script." In International Conference on Computer Analysis of Images and Patterns, pp. 371-382. Springer, Cham, 2015.
- [52] Ghadhiban, H.Q., Othman, M., Samsudin, N.A., Ismail, M.N.B. and Hammoodi, M.R., 2020, January. Survey of Offline Arabic Handwriting Word Recognition. In International Conference on Soft Computing and Data Mining (pp. 358-372). Springer, Cham.
- [53] J. J. Hull, "A Database for Handwritten Text Recognition Research", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 16, (1994), pp. 550–554.
- [54] P. J. Grother, "Handprinted Forms and Character Database", Technical report, Technical Report and CDROM, National Institute of Standards and Technology, (1995).
- [55] L. Deng, "The Mnist Database of Handwritten Digit Images for Machine Learning Research", Signal Processing Magazine, vol. 29, no. 6, (2012), pp. 141–142.
- [56] Marti, Urs-Viktor & Bunke, H.. (2002). The IAM-database: An English sentence database for offline handwriting recognition. International Journal on Document Analysis and Recognition. 5. 39-46. 10.1007/s100320200071.
- [57] M. Pechwitz, S. S. Maddouri, V. Mrgner, N. El- louze and H. Amiri, "IFN/ENIT - Database of Handwritten Arabic words", In Colloque Inter, Fran-cophone Sur l'Ecrit et le Document, CIFED2002, (2002), pp. 129–136.
- [58] S. Al-Ma'adeed, D. Elliman and C. A. Higgins, "A Database for Arabic Handwritten Text Recognition Research, In Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition, (2002), pp. 485 – 489.
- [59] S. Al-Ma'adeed, D. Elliman and C. A. Higgins, "A Database for Arabic Handwritten Text Recognition Research", The International Arab Journal of Information Technology, vol. 1, no. 1, (2004), pp. 117–121.
- [60] Y. Al-Ohali, M. Cheriet and C. Suen, "Databases for Recognition of Handwritten Arabic Cheques", Pattern Recognition, vol. 36, no. 1, (2003), pp. 111–121.
- [61] E. EL-Sherif and S. Abdleazeem, "A Two-Stage System for Arabic Handwritten Digit Recognition Tested on A New Large Database", In International Conference on Artificial Intelligence and Pattern Recognition, (2007), pp. 237–242.

- [62] S. Abdleazeem and E. El-Sherif, "Arabic handwritten digit recognition," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 11, no. 3, pp.127-141, 2008
- [63] S. M. Awaidah and S. M., Mahmoud, "A Multiple Feature/Resolution Scheme to Arabic (Indian) Numerals Recognition Using Hidden Markov Models", *Signal Processing*, vol. 89, no. 6, (2009), pp. 1176-1184.
- [64] A. Lawgali, M. Angelova and A. Bouridane, "HACDB: Handwritten Arabic Characters Database for Automatic Character Recognition", In *4th European Workshop on Visual Information Processing*, (2013), pp. 255–259.
- [65] Mahmoud, Sabri & Ahmad, Irfan & Al-Khatib, Wasfi & Alshayeb, Mohammad & Parvez, Mohammad & Märgner, Volker & Fink, Gernot. (2014). KHATT: An open Arabic offline handwritten text database. *Pattern Recognition*. 47. 1096–1112. 10.1109/ICFHR.2012.224.
- [66] Al-Hajj, Ramy, Chafic Mokbel, and Laurence Likforman-Sulem. "Combination of HMM-based classifiers for the recognition of Arabic handwritten words." In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, pp. 959-963. IEEE, 2007.
- [67] Dreuw, Philippe, Stephan Jonas, and Hermann Ney. "White-space models for offline Arabic handwriting recognition." In *2008 19th International Conference on Pattern Recognition*, pp. 1-4. IEEE, 2008.
- [68] Kessentini, Y., Paquet, T. and Hamadou, A.B., 2010. Off-line handwritten word recognition using multi-stream hidden Markov models. *Pattern Recognition Letters*, 31(1), pp.60-70.
- [69] Rothacker, L., Vajda, S., and Fink, G.A., 2012, September. Bag-of-features representations for offline handwriting recognition applied to Arabic script. In *2012 International Conference on Frontiers in Handwriting Recognition* (pp. 149-154). IEEE.
- [70] Azeem, Sherif Abdel, and Hany Ahmed. "Effective technique for the recognition of offline Arabic handwritten words using hidden Markov models." *International Journal on Document Analysis and Recognition (IJDAR)* 16, no. 4 (2013): 399-412.
- [71] Abandah, Gheith A., Fuad T. Jamour, and Esam A. Qaralleh. "Recognizing handwritten Arabic words using grapheme segmentation and recurrent neural networks." *International Journal on Document Analysis and Recognition (IJDAR)* 17, no. 3 (2014): 275-291.
- [72] Hamdani, Mahdi, Patrick Doetsch, Michal Kozielski, Amr El-Desoky Mousa, and Hermann Ney. "The RWTH large vocabulary arabic handwriting

- recognition system." In 2014 11th IAPR International Workshop on Document Analysis Systems, pp. 111-115. IEEE, 2014.
- [73] Elleuch, Mohamed, Najiba Tagougui, and Monji Kherallah. "Deep learning for feature extraction of Arabic handwritten script." In International Conference on Computer Analysis of Images and Patterns, pp. 371-382. Springer, Cham, 2015.
 - [74] Stahlberg, Felix, and Stephan Vogel. "The QCRI recognition system for handwritten Arabic." In International Conference on Image Analysis and Processing, pp. 276-286. Springer, Cham, 2015.
 - [75] Ahmad, Irfan, and Gernot A. Fink. 2015. Multi-Stage HMM Based Arabic Text Recognition with Rescoring. In Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR 2015), 751–55. IEEE. doi:10.1109/ICDAR.2015.7333862.
 - [76] Ahmad, Irfan, and Gernot A. Fink. "Handwritten Arabic text recognition using multi-stage sub-core-shape HMMs." International Journal on Document Analysis and Recognition (IJAR) 22, no. 3 (2019): 329-349.
 - [77] Hassan, A.K.A., Mahdi, B.S. and Mohammed, A.A., 2019. Arabic handwriting word recognition based on scale invariant feature transform and support vector machine. Iraqi Journal of Science, pp.381-387.
 - [78] Hassan, A.K.A. and Kadhm, M.S., 2016. Arabic handwriting text recognition based on efficient segmentation, DCT and HOG features. International Journal of Multimedia and Ubiquitous Engineering, 11(10).
 - [79] Al-Nuzaili, Q., Al-Maadeed, S., Hassen, H. and Hamdi, A., 2018, March. Arabic Bank Cheque Words Recognition Using Gabor Features. In 2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR) (pp. 84-89). IEEE.
 - [80] M. Rabi, M. Amrouch, Z. Mahani and D. Mammass, "Recognition of cursive Arabic handwritten text using embedded training based on HMMs," 2016 International Conference on Engineering & MIS (ICEMIS), Agadir, 2016, pp. 1-5.
 - [81] R. Almodfer, S. Xiong, M. Mudhsh and P. Duan, "Enhancing AlexNet for Arabic Handwritten words Recognition Using Incremental Dropout," 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI), Boston, MA, 2017, pp. 663-669.
 - [82] Almodfer, Rolla, Shengwu Xiong, Mohammed Mudhsh, and Pengfei Duan. "Multi-column Deep Neural Network for Offline Arabic Handwriting Recognition." In International Conference on Artificial Neural Networks, pp. 260-267. Springer, Cham, 2017.

- [83] Wen, Y., Shao, Y. and Zheng, D., 2019, August. A Novel Deep Convolutional Neural Network Structure for Off-line Handwritten Digit Recognition. In Proceedings of the 2nd International Conference on Big Data Technologies (pp. 216-220).
- [84] LeCun, Y.; Bottou, L.; Bengio, Y. Gradient-based learning applied to document recognition. IEEE Proc. 1998, 86, 2278–2324.
- [85] El-Sawy, A., Loey, M. and Hazem, E.B., 2017. Arabic handwritten characters recognition using convolutional neural network. WSEAS Transactions on Computer Research, 5, pp.11-19.
- [86] Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).
- [87] Hochreiter, Sepp, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies." (2001).
- [88] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9, no. 8 (1997): 1735-1780.
- [89] Salehinejad, Hojjat, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. "Recent advances in recurrent neural networks." arXiv preprint arXiv: 1801.01078 (2017).
- [90] Sutskever, Ilya, James Martens, and Geoffrey E. Hinton. "Generating text with recurrent neural networks." In Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 1017-1024. 2011.
- [91] Yousefi, M.R., Soheili, M.R., Breuel, T.M. and Stricker, D., 2015, February. A comparison of 1D and 2D LSTM architectures for the recognition of handwritten Arabic. In Document Recognition and Retrieval XXII (Vol. 9402, p. 94020H). International Society for Optics and Photonics.
- [92] Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.
- [93] Graves, Alex, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. "A novel connectionist system for unconstrained handwriting recognition." IEEE transactions on pattern analysis and machine intelligence 31, no. 5 (2008): 855-868.
- [94] Scheidl, H., Fiel, S. and Sablatnig, R., 2018, August. Word beam search: A connectionist temporal classification decoding algorithm. In 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR) (pp. 253-258). IEEE.

- [95] Marceau, L., Qiu, L., Vandewiele, N. and Charton, E., 2019. A comparison of Deep Learning performances with others machine learning algorithms on credit scoring unbalanced data. arXiv preprint arXiv:1907.12363.
- [96] Krawczyk B. Learning from imbalanced data: open challenges and future directions. *Prog Artif Intell.* 2016;5(4):221–32. <https://doi.org/10.1007/s13748-016-0094-0>.
- [97] Japkowicz N. The class imbalance problem: Significance and strategies. In: In proceedings of the 2000 international conference on artificial intelligence (ICAI). 2000;111–7.
- [98] Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A. Mining data with rare events: a case study. In: Proceedings of the 19th IEEE international conference on tools with artificial intelligence—Vol. 02. ICTAI '07, IEEE Computer Society, Washington, DC. 2007. p. 132–9. <https://doi.org/10.1109/ICTAI.2007.130>.
- [99] Letter Frequency. 2016. Accessed November 13. https://en.wikipedia.org/wiki/Letter_frequency.
- [100] Intellaren. 2016. A Study of Arabic Letter Frequency Analysis. Accessed March 9. <http://www.intellaren.com/articles/en/a-study-of-arabic-letter-frequency-analysis>.
- [101] Shin, Hoo-Chang, Holger R. Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M. Summers. "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics, and transfer learning." *IEEE transactions on medical imaging* 35, no. 5 (2016): 1285-1298.
- [102] Graves, A. "RNNLIB: A Recurrent Neural Network Library for Sequence Learning Problems. 2016." URL <http://sourceforge.net/projects/rnnl>.
- [103] MATLAB version 2019a. Natick, Massachusetts: The MathWorks Inc., 2019.
- [104] Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin et al. "Tensorflow: A system for large-scale machine learning." In 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), pp. 265-283. 2016.
- [105] Bradski, Gary, and Adrian Kaehler. "OpenCV." *Dr. Dobb's journal of software tools* 3 (2000).
- [106] Lee, Chen-Yu, and Simon Osindero. "Recursive recurrent nets with attention modeling for ocr in the wild." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2231-2239. 2016.

- [107] Bluche, Théodore, Jérôme Louradour, and Ronaldo Messina. "Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention." In the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 1050-1055. IEEE, 2017.

|

Vitae

Name :Mohamed Muawia Eltay |

Nationality :Sudanese |

Date of Birth :1/1/1989|

Email :moh_us_2000@hotmail.com|

Address :31412, Bkr Ibn Hezam St, Dammam, Saudi Arabia|

Education:

- 2020:** Master of Science (Electrical Engineering), King Fahd University of Petroleum and Minerals, Saudi Arabia. CGPA: 3.56/4.00.
- 2013:** Bachelor of Electronics Engineering (with Honors), Multimedia University, Malaysia. CGPA: 3.67/4.00.
- 2005:** High school certificate in science, King Abdullah Secondary School, Dammam, Saudi Arabia. CGPA: 97.5/100.

Areas of research interest:

- Digital Signal Processing.
- Pattern Recognition

Publications:

- M. Eltay, A. Zidouri and I. Ahmad, "Exploring Deep Learning Approaches to Recognize Handwritten Arabic Texts," in *IEEE Access*, vol. 8, pp. 89882-89898, 2020, doi: 10.1109/ACCESS.2020.2994248.