QATAR UNIVERSITY

COLLEGE OF ENGINEERING

HANDWRITING RECOGNITION IN ARABIC HISTORICAL MANUSCRIPTS

BY

HANADI HASSEN MOHAMMED

A Dissertation Submitted to

the College of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Doctorate of Philosophy in Computer Science

June [2022]

COMMITTEE PAGE

The members of the Committee approve the Thesis of
Hanadi Hassen Mohammed defended on 16/05/2022.

Prof. Somaya Al-Madeed
Dissertation /Dissertation Supervisor

Prof. Abdesselam Bouzerdoum
Committee Member

Prof. Abdelaziz Bouras
Committee Member

Prof. Mustafa Serkan Kiranyaz
Committee Member

Approved:

Khalid Kamal Naji, Dean, College of Engineering

ABSTRACT

MOHAMMED, HANADI., Doctorate : June : 2022, Doctorate of Philosophy in Computer Science

Title: Handwriting Recognition in Arabic Historical Manuscripts

Supervisor of Dissertation: Prof. Somaya Al-Madeed.

Document Analysis and Recognition significantly impact humanitarian studies by revealing information hidden in historical document collections worldwide. This research area merges the sciences of computer vision and machine learning. This PhD dissertation aims at recognizing text in Arabic historical handwritten documents by learning and extracting visual representations inside these manuscripts. The proposed approaches presented in this dissertation have the primary purpose of creating effective systems to deal with challenges linked to Arabic handwriting recognition, particularly in ancient manuscripts with old handwriting. The use of Convolutional Neural Networks (CNNs) to tackle the Arabic handwriting recognition challenges is an integral part of this dissertation. Several architectures for developing high-performing features are suggested. A model based on CNN and Gated Recurrent Units (GRUs) is used to recognize a wide range of handwritten Arabic subwords extracted from historical documents. Because recent research has shown that typical CNNs' learning performance is limited as they are homogeneous networks with a simple (linear) neuron model, a further improvement in the handwriting recognition models using non-linear neuron models is implemented. Operational Neural Networks (ONNs) are recently proposed as heterogeneous networks with a non-linear neuron model. Even with compact architectures, they can learn highly complex and multi-modal functions. This PhD dissertation investigates the use of Self-Organized Operational Neural Networks (SelfONNs) for handwriting recognition and the generalization capabilities of non-

linear neuron models, i.e., if deep discriminative features can be created. An investigation of an adequate level of non-linearity of the Self-ONN layers to provide extensive information on the Self-ONN performance under various topologies is presented. With such a novel approach, superior performance is achieved on a historical Arabic dataset and state-of-the-art performance is gained with a significant performance gap overall recent methods on an English dataset.

Furthermore, a novel method for disambiguating undotted Arabic characters is presented. While the method is useful for handwriting recognition systems dealing with Arabic manuscripts with ancient undotted letters, it also improved the visual recognition performance on current Arabic handwritten documents with dotted and diacritized characters.

DEDICATION

I want to dedicate my dissertation to my beloved parents, my family, and the people who supported me through my education.

ACKNOWLEDGMENTS

I thank Allah the Almighty for granting me the motivation and strength to complete this dissertation.

I extend my sincere gratitude to my parents, husband, kids, brothers, and sisters for their unconditional love and continuous supplication to Allah.

I would like to sincerely thank my supervisor, Prof. Somaya Al-Maadeed, for encouraging, assisting, and guiding me to finalize this work. I am grateful for her excellent guidance and positive approach throughout my research.

I would also like to thank my friends Tooba, Kalthoum, Ratiba, and Reem for supporting and encouraging me.

I would like to thank the College of Engineering at Qatar University for giving me the Graduate Teaching Assistant award during my PhD studies and providing me with the required financial support and all the needs to achieve the requirements of this study.

Finally, this work was made possible by NPRP Grant# NPRP NPRP7-442-1-082 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 Motivation

Libraries around the world are digitizing huge volumes of handwritten historical manuscripts in order to preserve cultural heritage. However, after digitization, libraries are faced with a difficult and apparently unsolvable problem: converting document images into textual content so that they can be indexed by the search engines and eventually accessible to researchers and interested readers. This type of access might be achieved in digital libraries using search engines similar to those found on the Internet. To accomplish this, the historical scripts must be transcribed first into a text format that computers understand. Given the huge collection of document images, the process of manual transcribing is tedious and impractical. Instead, the process could be automated using computational tools.

This dissertation aims to investigate how existing handwriting recognition algorithms perform when dealing with historical documents, more specifically, historical Arabic documents. According to statistics in [1], between the 7th and 14th centuries, over 90 million manuscripts were written in Arabic. Unfortunately, historical Arabic document processing has received little attention in the literature [2].

Due to its degradation over time, analyzing a historical document is a difficult task. An ancient document could have reached a degraded state in a variety of possible ways:

- Changes in temperature, humidity, light, and air pollution cause chemical deterioration. These chemical reactions might cause the occurrence of yellowish color in the paper or ink and pigment discoloration.
- Degradation of biological systems induced by animals such as rats.

• Human-caused degradations, such as annotations, scratches, etc.

• Degradation as a result of the digitizing process such as poor digitization resolution, compression standard, etc.

In addition to this, some manuscripts were written before introducing dots to the Arabic letters. The early Arabic writing had no dots and no diacritics. Arabic letters were developed by adding dots first to existing letters. As a result, some letters have the same base form and are only distinguished by single, double, or triple dots above or below a letter e.g. letters 'ب','ت','ث','ن','ي'. **Table 1** shows samples of words without dotting and some corresponding dotted words. Each un-dotted word may accept multiple meanings. To figure out the correct meaning, Arabs used their intuition, context, or memory. Later, other minor signs (diacritics) are added to the Arabic writing to denote short vowels. If they are omitted from a text, the consequence is a large number of homographs.

Table 1. Arabic Words Before and After Dotting.

Before Dotting	After Dotting
سب	بيت بنت نبت
	ثبت يبث يثب
باب	ثاب تاب بات
	باب ناب ناب
بحر	يجر يحر بحر
	يخر نحر نخز

Table 2 shows an example of one word without diacritics and some possible variations of the words after adding diacritics. A word without diacritics may also accept multiple meanings. Even though diacritics are critical for understanding, up to 95% of Arabic digital texts do not include them [3]. Young Arabic learners, non-native speakers, and computerized programs find it challenging to determine the meaning of Arabic words without diacritics. Diacritics are essential in applications such as text-to-speech synthesis, machine translation, sentiment analysis, and part-of-speech tagging. As a result, several of automatic approaches for restoring diacritics on Arabic electronic texts have been developed, each with differing degrees of accuracy.

Table 2. A single Arabic word with and without diacritics.

Without Diacritics	With Diacritics
سمر	سَمَر
سمر	سَمْر
سمر	سَمَّر
سمر	سَمُر

On the other side, although the textual content of the Arabic language that exists today contains dotted words, the existing digitized Arabic manuscripts contain scripts that have lost dots due to physical degradation or because the manuscripts themselves were written before adding dots to the Arabic script. A manuscript like the one in **Figure 1** is difficult to read, even for native speakers. The available historical benchmarks prepared for OCR tasks either assigned the undotted image to some dotted ground truth like the IBN SINA [4] dataset or provide undotted ground truth for them

like the HADARA80P [5] dataset. Thus, we propose deep learning models that tackle challenging datasets to address the document image-to-text conversion problem. The first model is a CNN-GRU-based model that successfully transcribes a wide range of Arabic subwords. The second model improved a recent CNN-only model by using Self-organized ONNs to transcribe full line images taken from Arabic and English datasets. The last model is the first to provide a solution for disambiguating un-dotted Arabic script with the help of Self-ONNs.

Figure 1. An early Arabic manuscript from the University of Birmingham Library.

1.2 Problem Statement

This dissertation studies pattern recognition approaches for handwriting recognition in historical Arabic documents. Three subproblems are addressed, namely subword recognition, line recognition, and undotted text disambiguation.

In the following, Section 1.2.1 discusses the scope of this dissertation, Section 1.2.2 summarizes the methods used for handwriting recognition, and Section 1.2.3 addresses the contributions of the dissertation.

1.2.1 Scope

Although there exists a lot of digitized historical Arabic documents, very few benchmarks intended for computer vision evaluation are available. Three fully annotated historical Arabic datasets are found: IBN SINA[4], HADARA80P [5], and VML-HD [6]. Experiments in this dissertation are evaluated with respect to these historical datasets. In addition to Arabic datasets, we evaluated our models using IAM [7] English dataset. Text extraction is not addressed in this dissertation in order to focus on handwriting recognition. Experimental assessments are carried out on a segmented and manually annotated handwritten text. Errors in text extraction can lead to inaccurate results in real-world handwriting recognition applications, thus our reported results should be viewed as upper boundaries.

1.2.2 Methods

To model and recognize handwritten text, we use state-of-the-art strategies. Convolutional Neural Networks and Gated Recurrent Unit (GRU) are used for subword recognition[8]. Based on a Self-ONNs with generative neurons [9], an improved CNN-only model is used in handwritten line recognition and dots restoration.

1.2.3 Contributions

This dissertation addresses an incrementally challenging problem ranging from subword recognition to line recognition in handwritten Arabic scripts in which letter

dotting has been entirely or partially lost. The contributions of this dissertation are as follows:

1. An accurate and efficient method using CNN and GRU for handwriting recognition on a wide range of subwords extracted from Arabic historical documents. Results over IBN SINA and VML-HD datasets showed high accuracy. The model is generalized for recognizing historical handwritten text in multiple writing styles.

2. A line-level HTR model that improves on a state-of-the-art CNN-only approach using Self-ONNs is presented. State-of-the-art results are achieved using the proposed model on the IAM English dataset and exceptional results are also gained on the HADARA80P Arabic dataset.

3. A novel method within our knowledge to restore dots from undotted Arabic script is implemented. While the method provides a solution for handwriting recognition systems that deal with Arabic manuscripts with old undotted scripts, it shows its success in recent Arabic handwritten documents with dotted and diacritized scripts.

4. Analysis of the effects of dots and diacritical marks on the performance of the OCR model is presented.

5. In addition to dots restoration, we improve on a recent diacritics restoration model by introducing Self-ONNs with generative neurons to overcome the limitations of CNNs.

1.3 Outline

In the following, a brief description of the succeeding chapters, as well as the corresponding contributions is presented:

- CHAPTER 2: Literature review: highlights current state-of-the-art approaches in document analysis and handwriting recognition.
- CHAPTER 3: Neural Networks background: presents the theoretical foundations of neural networks, their recent advances, and methods used in our experiments.
- CHAPTER 4: Subword Recognition using CNN-GRU: describes the subwords which are disconnected parts of a word, it is easier to extract them as compared to complete words, especially in the case of cursive scripts like Arabic. A hybrid CNN-GRU architecture is described in this chapter. The proposed model with shallow convolutional layers is extracting robust features from subwords while the GRU layers are used to map the feature sequences to subword class labels.
- CHAPTER 5: Handwriting recognition using Self-ONNs with generative neurons: In order to improve the state-of-the-art performance level in HTR, we propose 2D Self-ONNs in the core of a novel network model. Self-ONNs are self-organized variations of ONNs with the generative neuron model that can generate any non-linear function using the Taylor approximation. Moreover, we utilize deformable convolutions which have recently been demonstrated to tackle the variations in the writing styles for HTR in a better way.
- CHAPTER 6: Undotted text disambiguation using Self-ONNs : Old Arabic writing without dots can be found in ancient documents while in modern Arabic writing, short vowels are usually omitted. Applications to restore the dots are absent in the literature despite their need in Handwritten Text Recognition (HTR) systems. We propose to use Self-

ONNs for the problems of diacritics restoration and dots restoration in Arabic handwritten texts.

- CHAPTER 7, Conclusion and future work: This chapter concludes the outcome of the described research in this dissertation. In addition, we emphasize the most essential research directions that have yet to be thoroughly investigated and can support or complement existing work, based on our understanding of the existing literature.

CHAPTER 2: LITERATURE REVIEW

For more than four decades, document analysis and recognition have been a hot area of research. Digitizing and understanding documents are of great interest to the humanities and computer vision communities. Several approaches for analyzing document images with the objective of obtaining the underlying text have been developed. The aim is to automatically transcribe the text inside document images into an electronic format. Because of the vast amount of untranscribed manuscripts, the output of such a task would make it easier to access information recorded on document images, particularly historical documents. By systematically mining information from various sources and analyzing and recognizing numerous historical texts on a broad scale, substantial insight into historical events could be gained, a goal of significant value to cultural heritage.

In order to construct a successful text recognition system, many additional subtasks (e.g., noise removal, document segmentation into lines, words, or characters) might be established. Document processing uses both computer vision and machine learning algorithms. In the following sections, we summarize essential tasks in the field of document analysis and recognition. Following that, we go over several significant research in handwritten text recognition in both Arabic and other languages.

2.1 Layout Analysis

Paragraphs, lines, words, and other structural elements make up the majority of documents. Correctly extracting these pieces will significantly help the text recognition stage, as the problem complexity will decrease considerably.

Binarizing images of documents before processing them further was a typical pre-processing step in the preceding decade of a document analysis pipeline. The purpose of document binarization is to make a binary image by assigning pixels corresponding to characters as foreground (1) and non-text pixels as background (0). This stage could result in significant information loss; thus, it has been removed in recent approaches.

Document segmentation has been shown to be a crucial step in achieving high-quality recognition results. Character segmentation is practically impossible in handwritten manuscripts because successive characters are commonly merged into bigrams, trigrams, and so on, while word segmentation is still prone to errors. The easiest entities to segment effectively were lines, and many ways for accurately segmenting lines evolved over time (e.g., Hough transform [10]). The segmentation process should generate reliable results to avoid error propagation to the text recognition stage.

2.2 Keyword Spotting (KWS)

Keyword spotting (KWS) is an alternative to HTR. It is defined as the task of retrieving specific words of interest in a document collection without having to transcribe every single word in the collection using a specific keyword as a query. A keyword spotting algorithm generates a prioritized list of word pictures based on their similarity to the query. The keyword word that needs to be detected can be a text string (query by string (QbS)) or an example image (query by example (QbE)). Based on the search space, keyword spotting methods are divided into Segmentation-Free and Segmentation-Based. Segmentation-free approaches try to find query instances over the entire document without involving a segmentation step. Segmentation-based

approaches rely on a preceding segmentation step. Techniques based on segmentation can be further classified based on the level of segmentation. Word-based methods compare word images and assume that words have already been segmented [11]–[13]. Line-based methods, on the other hand, assume that a line segmentation has already been applied and the word spotting will be executed at the line level [14], [15].

2.3 Text Recognition

Text recognition is considered the most important task in the document processing domain. A text recognition step is included in the majority of document-related systems. The entire transcription of a document image is the ultimate goal of document analysis. The fact that there are so many distinct writing styles across people or even between eras and languages adds to the problem's complexity.

Optical Character Recognition (OCR), which was a hot topic of research in the 1990s, was the forerunner of modern text recognition (Character classification was the first task for which neural networks were evaluated for computer vision tasks). Character segmentation is often assumed in OCR, which is only possible for typewritten text. Although OCR and typewritten text are of little practical or scientific importance in today's world, they depict the growth of machine learning and computer vision through time.

Handwriting recognition can be done at the character level [16], which means that the text is identified one at a time as a single isolated character. This was the initial issue tackled with LeNet [17], and it is still being done for languages like Japanese [18] and Chinese [19]. For alphabetic languages, handwriting recognition can be also performed at the word level [20], [21]. The aim, in this case, is to decode single words that are recognized in the image. This task is performed both on digitalized documents

and in scene images[22]. Furthermore, several works concentrate on line-level HTR, in which the entire text of a line is transcribed, including spaces that are ignored in the word level handwriting recognition. Line-level handwriting recognition can be done on a pre-segmented text [23]–[27] or as part of a joint detection and recognition system in which both line segmentation and detection are performed [29]. Finally, recent work directly addresses handwriting recognition at the paragraph [30] or page level [28]. Layout analysis approaches such as paragraph or line segmentation are usually combined in such works [31]–[33].

Text recognition from a sequential perspective has a lot in common with speech recognition, which is a classic signal processing and machine learning problem. Many text recognition systems were affected by their speech recognition counterparts, and vice versa. The use of Hidden Markov Models (HMMs) guided the initial attempts at line-level text recognition [32]–[35]. A robust feature extraction method was usually required prior to the use of HMMs such as edge detectors [36]. Simple statistical features were often retrieved column-by-column from the initial handwritten line image, resulting in a sequence of features that were then used as input to an HMM.

The foundation of cutting-edge recognition algorithms is a recurrent module that captures the sequential nature of the text. In the last decade, Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), and more recently, CNNs have gained popularity due to their superior performance.

Recurrent Neural Networks (RNNs), and Multidimensional Long Short-Term Memory (MDLSTM) networks, have been extensively used in HTR. MDLSTM networks show state-of-the-art performance on most of the HTR benchmarks. Regular LSTM networks differ from MDLSTM in that the former introduces recurrence along the axis of one-dimensional sequences, whereas the latter introduces recurrence along

two axes which makes it ideal for handling unrestricted two-dimensional input. In line-level HTR, it is common to use the MDLSTM to extract features. The character-level transcription of the input line image is then obtained by converting the 2D data into a 1D sequence. This design is fundamentally at the heart of most of the successful line-level HTR techniques; however, compared to Convolutional Neural Networks (CNNs), MDLSTMs are computationally expensive. Furthermore, a visual comparison of the 2D-LSTM features retrieved in the bottom layers reveals that they are visually similar to the 2D CNN outputs [27].

CNNs are well known for feature representation of input images. Recently, they have been adopted in handwriting recognition models in combination with Recurrent Neural Networks (RNNs), which are responsible for generating output sequence and decoding the underlying text [37]. In [38], three CNN layers are used to extract features from input images which are then fed into two CNN-MDLSTMs for extracting context information. The proposed model in [39] incorporates CNNs with MDLSTMs, but instead of setting CNNs as feature extractors for the input images, layers of LSTMs scan the blocks of input images in different directions; then, CNN layers receive the output of each LSTM layer and again forward to LSTMs. The top-most layer is fully connected rather than convolutional. The softmax layer receives the last activations, which are summed vertically. Connectionist Temporal Classification (CTC) is used to process the output of the softmax layer. A similar idea is proposed in [40], but they work with an optimized version of MDLSTM where the convolutional and recurrent layers have been relocated, and the subsampling processes have been tweaked to improve feature extraction at the bottom levels while lowering the activation volume before reaching the upper layers. An HMM is used in the decoding step to reduce errors generated by the CNN- MDLSTM optical model.

A recent study [41] introduced a convolutional-only architecture for HTR. They use deformable convolutions [42] to tackle the problem of diversity in writing styles, as the deformation of the kernel can be interpreted as geometrical deformations of the same textual elements. Deformable convolutions broaden the definition of convolution by redefining the shape of the convolution as adaptable. Convolution weights are supposed to multiply inputs not on the conventional orthogonal, canonical k × k grid but rather on a learning-based weight-input coordinate correspondence. The state-of-the-art performance level was achieved in [41] by reducing character uncertainty at the network's softmax output.

2.3.1 Handwritten Text Recognition in Historical Arabic Documents

Compared to HTR in other scripts such as Chinese and Latin, little work was put into Arabic text recognition. This is due to some characteristics of the Arabic writing system summarized below:

- The cursive form of Arabic writing and the similarities between distinct letter shapes.
- The Arabic is written from right to left.
- The shape of the character changes depending on where it appears in the word. Each character has two to four different variations.
- Dots appear in fifteen of the letters. They might be on top of or underneath the letter.
- Some of the letters have the same basic shape. The only thing that distinguishes them is the dots.
- If these letters (و, ز, ر, ذ, د, ا) exist in the word, the word will be broken into two or more sub-words separated by spaces, which are usually shorter than the space

between words. To avoid segmentation of a word into two words, this must be considered.

A number of benchmark datasets have been made available to evaluate handwriting recognition (and other related tasks) on historical Arabic documents. Among these, HADARA [5], IBN SINA [4], and VML-HD [6] have been commonly employed in a number of studies. Most of the work on HADARA dataset is on keyword spotting where the objective is to match a query word image in the collection of documents and retrieve all instances of the queried keyword. The IBN SINA dataset is one of the most widely employed Arabic datasets and has been used for evaluation of word spotting as well as word recognition systems. VML-HD is a relatively recent dataset that contains a large vocabulary of subwords written in different writing styles.

Among notable studies on the analysis of Arabic manuscripts, [43] propose a 5-layered convolutional neural network to recognize 68 classes of Arabic subwords extracted from historical collections. The network comprises 2 convolutional and 3 fully connected layers and reports a recognition rate of 81%. The work was later extended in [44] to study the impact of synthesizing and augmenting data to recognize 39 subword classes from 10 pages of the VML-HD dataset. Through a comprehensive series of experiments, the authors concluded that data augmentation results in relatively better performance as compared to synthesizing data. The authors also concluded that 10 handwritten pages proved to be sufficient for training the model and recognizing the subwords.

The authors in [45] proposed an efficient method to reduce the Arabic subword lexicon by exploiting the topology and geometry of subwords. The skeleton of a subword image is extracted and is represented by a directed a-cyclic graph (DAG). The subword DAG is then encoded into a topological signature vector (TSV), which is a

low dimensional feature vector. Given a query subword, the lexicon is reduced by computing the TSV distance between the query and the lexicon subwords and keeping a set of nearest subwords only. Subsequently, the reduced lexicon is fed to another nearest neighbor classifier. An experimental study of the system is carried out on the IBN SINA database and recognition rates of 82.20% and 86.16% are reported with and without lexicon reduction respectively. Authors attributed the errors in the case of lexicon reduction to the differences in the writing styles of different writers.

In another study, the concept of topological signature vector (TSV) introduced in [46] was extended and improved to a weighted TSV (W-TSV) [47]. The introduction of weights proved effective as noise corresponds to smaller weights and thus it can be filtered out easily. The representation was also enriched by introducing three different DAG representations. These included the topological DAG (T-DAG) which preserves the information of the subword shape topology, the length DAG (L-DAG) which adds information about skeletal curve lengths by applying weights to the T-DAG, and finally, the curvature DAG (C-DAG) which carries information about the curved structure of the subword. In a further extension of this work, the authors introduced the concept of Arabic word descriptors (AWD) [47] to create an index for a reference database of the subwords' shapes. The AWD is created from sorted and normalized structural descriptors (SD) of all the subwords using the bag-of-words (BOW) model, hence allowing an efficient shape matching. When a subword is provided as a query, its reduced lexicon is acquired from the labels of the top-ranked entries in the indexed database. The proposed technique improved the lexicon reduction accuracy on IBN SINA and IFN/ENIT[48].

Among other studies, skeleton-based features are employed for Arabic subword recognition with the SVM classifier in [49], and an accuracy of 89.66% is reported in

this study. Fouladi et al. [50] employ localized orientation histograms together with a contour alignment method to recognize subwords in a writer-dependent framework. Experiments on Arabic subwords in the IBN SINA dataset as well as on a collection of Farsi words reported accuracy of around 91%.

A number of studies targeting the IBN SINA dataset are primarily concerned with dictionary reduction. The key idea is to address the problem of high computational complexity due to a large lexicon. The results suggest that when performing lexicon reduction, a reduction of the dictionary would result in reduced recognition accuracy. Hence, the trade-off between high accuracy and reducing the lexicon size needs to be managed [47]. The performance charts for the IBN SINA dataset are mostly dominated by conventional methods that employ handcrafted features with traditional classifiers such as the Nearest Neighbor. Feature learning and end-to-end deep learning-based solutions have not been extensively used for the recognition of Arabic historical documents.

2.3.2 Post Processing

Text recognition requires more than just taking visual cues into account. Even humans, for example, require some language background to resolve potential ambiguities between similar characters, such as "ب" and "ت". To this goal, after visual recognition, a post-processing phase is used to try and combine existing language models, which are calculated using the occurrence frequency of characters' n-grams, given a specified corpus. This can be expanded to include n-grams of whole words. In other words, language models penalize rarely occurring sub-sequences of characters that may correspond to visual recognition artifacts. Language models are used to enhance optical model results.

Even though trainable models, such as RNNs, implicitly learn the language model present in the training corpus, external language models can be actively imposed on the decoding process to aid final decoding. External language models can be generated from an existing corpus of the desired language in the form of n-grams or word-grams. As a result of the additional information derived from language modeling, there is a noticeable improvement since it can remedy probable inaccuracies in visual feature decoding. As old Arabic manuscripts include letters without doting, extracting statistics of language models from the modern Arabic writing corpora might not help in improving OCR in such kinds of documents. A new algorithm to tackle the problem of letter dotting is needed, which we propose in this dissertation in CHAPTER 6.

2.4 Document Forensics

The processing of documents is not limited to layout analysis and recognition or detection. Document forensics include writer identification, automatic signature identification and verification, detection of forged or fake documents, and document authentication. The significance of such tasks is obvious, and automatic forgery detection and authentication would substantially aid the automation of legal forms' processing and verification. In terms of implementation, forensics methods necessitate finding minor patterns that distinguish individual writers or an authentic document from possible forgeries using machine learning algorithms.

2.5 Document Understanding and Document Classification

Companies frequently have significant collections of various document types such as invoices, legal documents, etc. that are difficult to manage. Document understanding tries to digitize and extract knowledge from thousands of documents

such as tables, graphs, form fields, and text passages. It makes it easier to search, query, and monitor knowledge graphs and other extracted data by allowing them to be organized and stored.

2.6 Conclusion

This chapter discussed the background of documents' analysis and recognition and related handwriting recognition literature in Arabic and non-Arabic scripts. From the abovementioned literature, the handwriting recognition systems in Arabic script are gaining less attention due to the challenges related to the nature of Arabic ancient documents and related to Arabic script mentioned earlier. In addition to this, there is a gap in processing ancient documents having an undotted script. This dissertation aims to fill this gap and improve on the work done.

CHAPTER 3: NEURAL NETWORKS BACKGROUND

In this chapter, we describe the theoretical background briefly with a focus on Neural Networks (NNs), which play a significant role in this dissertation. Until the current re-emergence of Neural Networks (NNs), computer vision tasks were guided by a specific methodological overview defined by the following steps: the first is feature extraction, and the second is machine learning. Most of the existing techniques for various of computer vision tasks mainly relied on the feature extraction step with the goal of extracting as much information from an image as possible into a concise descriptor. These handcrafted features were then fed into a machine-learning algorithm to achieve specific tasks such as image classification using Support Vector Machines [51]. In computer vision, statistical features (e.g., statistical distributions of pixels) and structural features (e.g., topological and geometric properties of a shape) have been successfully employed. Because data were usually few and resources were restricted, the features needed to be highly discriminative to improve the generalization. Unsupervised learning, such as Principal Component Analysis (PCA), was used to enhance discriminative representation.

Machine learning techniques like SVMs, HMMs, and NNs were used only as a last step after precisely building the feature extraction process. However, the amount of labeled data combined with the computational capabilities of cutting-edge hardware (most notably GPUs) resulted in extensive neural network usage across all computer vision tasks over the previous decade. Existing resources, in particular, allow for the efficient training of end-to-end deep NNs with hundreds, if not millions, of layers. We examine some fundamental topics of building NNs since deep learning will be a major component of this dissertation.

3.1 Multi-Layer Perceptron (MLP)

NNs are inspired by the brain, it was firstly introduced by McCulloch and Pitts. A linear transform is topped by a step activation function in the McCulloch-Pitts neuron, also known as a linear threshold gate. Rosenblatt presented the Perceptron in 1958 as an improvement over the McCulloch-Pitts Neuron, introducing the idea of trainable weights as well as an adequate training procedure for binary classification. Multiple neurons are stacked to create a multi-layer feed-forward network to expand the neuron notion to classify non-linearly separable classes. Hidden layers are intermediate layers that exist between input and output. **Figure 2** depicts the perceptron and **Figure 3** depicts the multi-layer network.

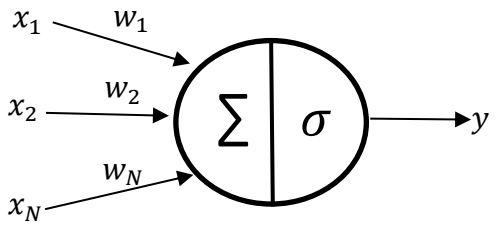


Figure 2. A Perceptron Neuron with dot product and activation function (σ).

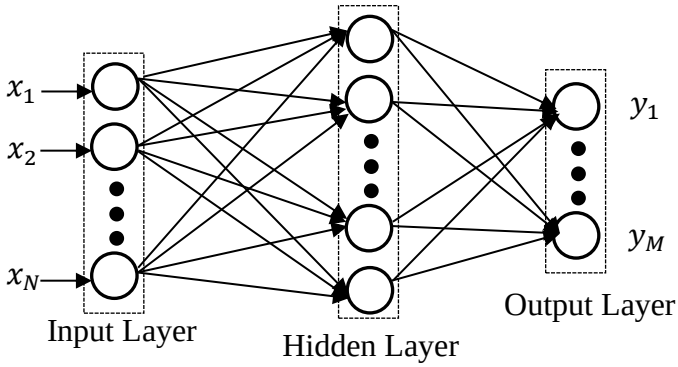


Figure 3. A multi-layer feed-forward neural network with three layers.

The MLP with the linear model in Equation 1 shows how the output from the previous layer x_i^{l+1} contributes to the following layer input $(l+1)$. Then the activation function which, represents the non-linearity, is applied to layer $(l+1)$ neurons.

$$y_i^{l+1} = b_i^{l+1} + \sum_{k=1}^{N_l} x_i^l \, w_{ik}^{l+1}, \quad \forall i \in [1, N_{l+1}], \tag{1}$$

A deep neural network (DNN) is formed when more hidden layers are used. Although the DNNs are more complex and require more computational power, they are still used in some problems such as acoustic scenes and events detection and classification [52].

3.2 Back Propagation

The fundamental issue with training neural networks is the complexity of their stacked-layer structure. The chain rule is used to calculate the gradient of each parameter with respect to the loss function. It is a basic yet powerful concept. The following two steps can be characterized as iterative gradient-based optimization schemes: computing the gradients and then updating the weights. These two processes are repeated in this order until convergence is achieved. The gradient would be close to zero at convergence.

3.2.1 Computing Gradients

Computing gradient in deep architectures is not straightforward. Computing gradient at layer-wise is the solution by moving backward from the loss function toward the input layer and thus named backpropagation.

Consider the following example of a fully-connected intermediate layer: $y = \sigma(w^T x)$, where y is a single-dimensional output, σ is the activation function, w is the layer's weights, and x is the k-dimensional input. Considering the bias is excluded and

the output is one-dimensional for simplicity, we want to calculate the gradients $\vartheta J/\vartheta w$ and $\vartheta J/\vartheta x$ given the gradient at the output $\vartheta J/\vartheta y$. The weights are then updated, and the gradient error is propagated to previous layers. The chain rule for the first and second are expressed as follows:

$$\frac{\vartheta J}{\vartheta w} = \frac{\vartheta J}{\vartheta y}\frac{\vartheta \sigma(w^T x)}{\vartheta (w^T x)}\frac{\vartheta (w^T x)}{\vartheta w} = \frac{\vartheta J}{\vartheta y}\,\sigma'(w^T x)x \qquad (2)$$

$$\frac{\vartheta J}{\vartheta x} = \frac{\vartheta J}{\vartheta y}\frac{\vartheta \sigma(w^T x)}{\vartheta (w^T x)}\frac{\vartheta (w^T x)}{\vartheta x} = \frac{\vartheta J}{\vartheta y}\,\sigma'(w^T x)w \qquad (3)$$

The gradient chain rule can be employed consecutively from the network's output to its input to calculate every parameter gradient with respect to the objective loss function, as shown above.

3.2.2 Updating Weights

Vanilla Gradient Descent employs the following update rule to compute the gradient score of the objective function with regard to the parameters θ of the whole training set: $\theta \leftarrow \theta - \eta.\nabla_\theta J(\theta)$. The convergence rate is controlled by the hyper-parameter η. Large numbers can cause overshooting and divergence, while small values might cause slow convergence.

Due to the fact that calculating the gradients over the whole dataset can impose significant computing overhead, the above formulation is impractical. The standard Stochastic Gradient Descent (SGD) optimization is employed to solve this problem. For each training sample, SGD updates the parameter. Epoch refers to a whole iteration of each data sample. At each epoch, the pairs of inputs and targets are supplied to the SGD in a different sequence. In practice, the SGD mini-batch option is employed, in which the gradients are calculated across batches of samples.

3.3 Recurrent Neural Networks (RNNs)

Since the emergence of multi-layer feed-forward networks, advances in neuronal structure have been made, resulting in two types of neural networks that will be discussed in this dissertation: Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs).

Sequence modeling is a popular research area, with applications in Natural Language Processing (NLP), speech recognition, and stock prediction. Because standard neural network formulations cannot model sequences of data, an alternative is necessary. Recurrent Neural Networks (RNNs) were developed to achieve this goal. RNNs are a type of neural network that uses previous outputs as inputs while maintaining hidden states.

RNNs are distinguished by their memory, allowing them to influence current input and output using prior input information. RNNs' output is dependent on the sequence's prior parts, whereas traditional deep neural networks assume that inputs and outputs are independent of one another. **Figure 4** shows the difference between RNNs and Feedforward Neural Networks. While future occurrences may be useful in deciding the output of a sequence, unidirectional recurrent neural networks are unable to account for them in their predictions.

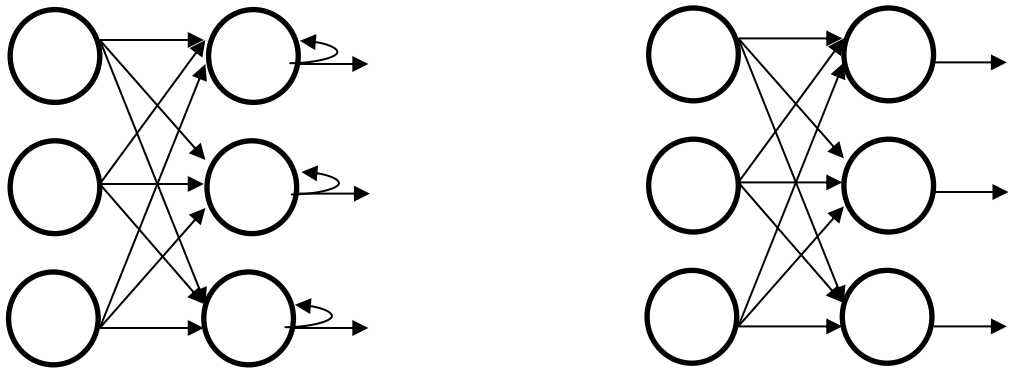


Figure 4. Comparison of RNNs (left) and Feedforward Neural Networks (right).

Assuming that the x_i segments represent an RNN's input sequence and the h_i segments represent its hidden state, the following recurrent formulation emerges:

$$h_i = \sigma(f_h(h_i - 1) + f_x(x_i)) \quad (4)$$

where $\sigma()$ is a non-linear activation function, $f_h()$ is the hidden state transformation function and $f_x()$ is the input transformation function. The linear transformation for $f_h()$ and $f_x()$ is formulated by $f_h(x) = W_h x + b_h$, where W is the weight matrix and b is the bias. The same weights, consisting of the functions $f_h()$ and $f_x()$, are utilized at each step of the recurrent formulation of Equation 4 As a result, an unrolled RNN can be considered as a standard NN with shared weights. There are several variants of RNNs, we specifically differentiate the following cases:

3.3.1 Bidirectional recurrent neural networks (BRNN)

BRNNs are RNNs with different network architectures. Bidirectional RNNs pull in future data to increase accuracy ($l_i = f_l(l_{i+1} + x_i)$), whereas unidirectional RNNs can only draw on previous inputs to create predictions about the current state ($r_i = f_r(r_{i-1} + x_i)$). At each time step i, bi-directional RNNs incorporate both information flows, rightward r_i and leftward l_i: $h_i = f_c(r_i, l_i)$. To better comprehend the context and minimize ambiguity, you may need to learn representations from future time steps.

3.3.2 Long short-term memory (LSTM)

LSTM is a popular RNN architecture introduced as a solution to the vanishing gradient problem. LSTMs use a concept of cell state that serves as a transportation highway for relative information as it travels down the sequence chain.

In LSTM, the forget gate f_t (Equation 5), input gate i_t (Equation 6), output gate

o_t (Equation 7) are using sigmoid function (σ) to output values in range [0,1].

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{(t-1)} + b_{hf}) \quad (5)$$

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

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{(t-1)} + b_{ho}) \quad (7)$$

The candidate memory cell $\tilde{c}_t$ in Equation 8 is calculated using tanh, which outputs values in the range [-1,1]. The output cell c_t in Equation 9 is calculated using input and forget gates with the Hadamard product ($\odot$).

$$\tilde{c}_t = tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{(t-1)} + b_{hg}) \quad (8)$$

$$c_t = f_t \odot c_{(t-1)} + i_t \odot \tilde{c}_t \quad (9)$$

The forget gate is responsible for controlling how much of the old memory cell ($c_{(t-1)}$) is preserved while the input gate is controlling how much of the new candidate cell ($\tilde{c}_t$) is considered (e.g. the previous cell will be considered when f_t is close to 1 while i_t is close to 0). The cell state c_t and the output gate o_t are used to calculate the output hidden state h_t as follows:

$$h_t = o_t \odot tanh(c_t) \quad (10)$$

3.3.3 Gated recurrent units (GRUs)

GRUs contain an update gate that determines how much information from the hidden state should be allowed through (z_t), as well as a reset gate that determines how much information should be rejected from the hidden state (r_t).

$$r_t = \sigma(W_{ir}x_t + b_{ir} + W_{hr}h_{(t-1)} + b_{hr}) \quad (11)$$

$$z_t = \sigma(W_{iz}x_t + b_{iz} + W_{hz}h_{(t-1)} + b_{hz}) \quad (12)$$

Then the candidate hidden state (n_t) is calculated as follows:

$$n_t = tanh(W_{in}x_t + b_{in} + r_t \odot (W_{hh}h_{(t-1)} + b_{hn})) \quad (13)$$

Lastly, the hidden state (h_t) is calculated as follows:

$$h_t = (1 - z_t) \odot n_t + z_t \odot h_{t-1} \quad (14)$$

The candidate state is used if the update gate z_t is near zero. The current hidden state is equivalent to the previous hidden state if the update gate is near 1. LSTM and GRU strive to maintain information from distant time steps without vanishing it across time, but the GRU is cost-efficient compared to LSTM. In the majority of contemporary sequence-related tasks, LSTMs and GRUs are chosen. In CHAPTER 4, we show how the GRU is used in Arabic subword recognition.

3.4 Convolutional Neural Networks (CNNs)

The advent of Convolution Neural Networks, which can successfully process images, is a key part of neural networks' resurrection. Because spatial context must be captured, image-related issues have always been complex. Convolution with a handcrafted kernel, designed to catch specific patterns, was traditionally used to conduct spatial filtering such as edges. CNNs introduced trainable filters that can build discriminative feature maps optimal for the given task. This method transformed the field of computer vision, displacing handcrafted features.

The convolution layer, in essence, converts one image feature map to another while taking into account contextual information about each pixel's surroundings. It's worth noting that convolution is a linear operation that takes the place of traditional NNs' linear projection. Low-level features, such as edges, are generated by layers closer to the input, whereas high-level features, such as an eye or a nose in face detection systems are generated by layers closer to the output. Because large kernel convolutions are expensive, multiple layered convolutions of small kernel 3×3 are recommended [53][54].

Let us define the input tensor to a layer by $X \in R^{H \times W \times C_{in}}$ and a sub-tensor

of it centered at the position i,j by $X_{(i,j)} \in R^{h\times w\times C_{in}}$. Let us define $W_{c_{out}} \in R^{h\times w\times C_{in}}, c_{out} = 1,\dots,C_{out}\ the\ c_{out}'th$ filter of a layer. In CNNs, convolutional neurons convolve X with $W_{c_{out}}$ and add an offset, which equates to doing the following calculation for each point (i,j) of the input tensor X:

$$Y_{c_{out}}(i,j) = \sum_{k,m,c_{in}=1}^{h,w,C_{in}} W_{c_{out,}}(k,m,c_{in})X_{(i,j)}(k,m,c_{in}) + b_{c_{out}}$$

$$= w_{c_{out}}^{T} x_{(i,j)} + b_{c_{out}} \quad (15)$$

the $Y_{c_{out}}(i,j)$ is the $(i,j)'^{th}$ element of the output feature map $Y_{c_{out}}, b_{c_{out}}$ is the bias term, and $x_{(i,j)}$ and $w_{c_{out}}$ are vectorized versions of $X_{(i,j)}$ and $W_{c_{out}}$, respectively. The feature mappings $Y_{c_{out}}, c_{out} = 1,\dots,C_{out}$ are concatenated to generate the tensor $Y \in R^{H\times W\times C_{out}}$, and the layer output is produced using an element-wise activation function. Stacking layers in a specific order is not a requirement of the CNN architecture. A majority of recent architectures, on the other hand, involve complicated information flows with many pathways such as GoogleNet [55], ResNet [56], DenseNet [57], etc.

3.5 Generalized Operational Perceptrons

GOP neurons are a natural superset of linear MLP neurons. They allow for improved input signal encoding using linear and non-linear fusion algorithms, resulting in more compact neural network designs with superior performance. The linear model of MLP in Equation 1 is replaced by a non-linear model in GOPs with three operators: nodal operator Ψ_i^{l+1}, pool operator P_i^{l+1}, and activation operator f_i^{l+1} as follows:

$$y_i^{l+1} = b_i^{l+1} + P_i^{l+1}\left(\Psi_i^{l+1}\left(w_{1i}^{l+1}, x_i^l\right),\dots,\Psi_i^{l+1}\left(w_{ki}^{l+1}, x_k^l\right), n\right),\ \forall i \in [1, N_{l+1}] \quad (16)$$

The nodal operator can be multiplication, harmonic (sinusoid), exponential, quadratic function, Hermitian, Gaussian, Laplacian of Gaussian (LoG), and derivative

of Gaussian (DoG). The pool operator can be summation, maximum, median, and n-correlation. The activation operator can be tanh, and lin-cut.

3.6 Operational Neural Networks (ONNs)

According to recent studies, [58]–[61], CNNs, like their predecessors, Multi-Layer Perceptrons (MLPs), rely on the ancient linear neuron model, so they are successful in learning linearly separable problems very well, but they may completely fail when the problem's solution space is highly non-linear and complex. ONNs [58], [61]–[63], are recently proposed heterogeneous networks with a non-linear neuron model. They can learn highly complex and multi-modal functions or spaces even with compact architectures. Similar to Generalized Operational Perceptrons (GOPs) [58]–[60], [64], [65], operational neurons of ONNs are modeled similar to biological neurons, with nodal (synaptic connections) and pool (synaptic integration in the soma) operators. An operator set is a collection of the nodal, pool, and activation operators, and the operator set library needs to be built in advance to contain all possible operator sets.

ONNs [38], which are derived directly from GOPs, are heterogeneous networks that encapsulate neurons with linear and non-linear operators, bringing them closer to biological systems. In summary, the nodal and pool operators in ONNs extend the exclusive use of linear convolutions in convolutional neurons.

ONNs generalize CNNs transformation in (1) using:

$$Y_{c_{out}}(i,j) = \Psi\left(x_{(i,j)}, w_{c_{out}}\right) + b_{c_{out}}, \tag{17}$$

where Ψ is a nodal function that can be a combination of different functions. During the training, the selection of Ψ is achieved using a search strategy. The ONN layer is a conventional CNN layer when the nodal function is determined to be the dot-product

of its arguments. The operator chosen will be applied to every connection and every kernel element in the network [66].

3.7 Self-Organized ONNs

ONNs, too, have a variety of limits and downsides as a result of such fixed and static architecture. First, only the operators in the operator set library can obviously be used, and if the correct operator set for the learning problem at hand is not in the library, the desired learning performance cannot be achieved. Second, to reduce the search space, one or few operator sets can be assigned to all neurons in each hidden layer, which poses a limited level of heterogeneity. Finally, there is always a need for searching for the best operator sets for each layer, which might be cumbersome, especially for deeper networks.

To tackle the aforementioned problems, [9] proposed Self-ONNs with generative neurons. The generative neuron model allows Self-ONNs to self-organize by iteratively generating nodal operators during the back-propagation (BP) training to maximize the learning performance. Certainly, being able to create any non-linear nodal operator significantly improves both operational diversity and flexibility.

Self-ONNs promise a similar or better performance level than conventional ONNs with an elegant diversity and reduced computational complexity. The self-organization capability of Self-ONNs is enabled by the generative neuron model, in which the nodal operators are iteratively generated during backpropagation training to maximize learning performance. In CHAPTER 5 we show how we adapted Self-ONNs to improve handwriting recognition performance in both Arabic and English datasets.

Instead of searching for the best possible nodal function, during training, each generative neuron in a Self-ONN can iteratively generate any nonlinear function of

each kernel element with a truncated Mac-Laurin series expansion:

$$\Psi\left(x_{(i,j)}, w_{c_{out},1,\ldots,w_{c_{out},Q}}\right) = w^T_{c_{out},1} x_{(i,j)} + w^T_{c_{out},2} x^2_{(i,j)} + \cdots + w^T_{c_{out},Q} x^Q_{(i,j)} =$$

$$\sum_{q=0}^{Q} w^T_{c_{out},q}\, x^q_{(i,j)}\,, \qquad (18)$$

where $x^q_{(i,j)}$ is an element-wise power, $w_{c_{out},q}$ are learnable weights interacting with $x^q_{(i,j)}$. As a result, each neuron undergoes the following transformation:

$$Y_{c_{out}}(i,j) = \sum_{q=1}^{Q} w^T_{c_{out},q}\, x^q_{(i,j)} + b_{c_{out}}, \qquad (19)$$

where $w_{c_{out},q}$, $q = 1,\ldots,Q$, are learned using gradient-based optimization since the purpose is to learn the best suited nodal function.

An illustration of the kernels in CNNs, ONNs, and Self-ONNs is depicted in **Figure 5**. In conventional CNNs (**Figure 5** (left)), a linear transformation is always used in convolution with the input tensor. In ONNs (**Figure 5** (middle)), a selected non-linear operator is used for all kernel elements (e.g. sinusoids with different frequencies). In Self-ONNs (**Figure 5** (right)) the right nodal operator for every kernel element, every neuron, and every synaptic connection is generated during (BP) training. This allows that in Self-ONNs, for a certain kernel element the nodal operator can be linear, while for another may be similar to a sinusoid or any arbitrary non-linear function. This allows neuron-level and even kernel-level diversity and heterogeneity. For more

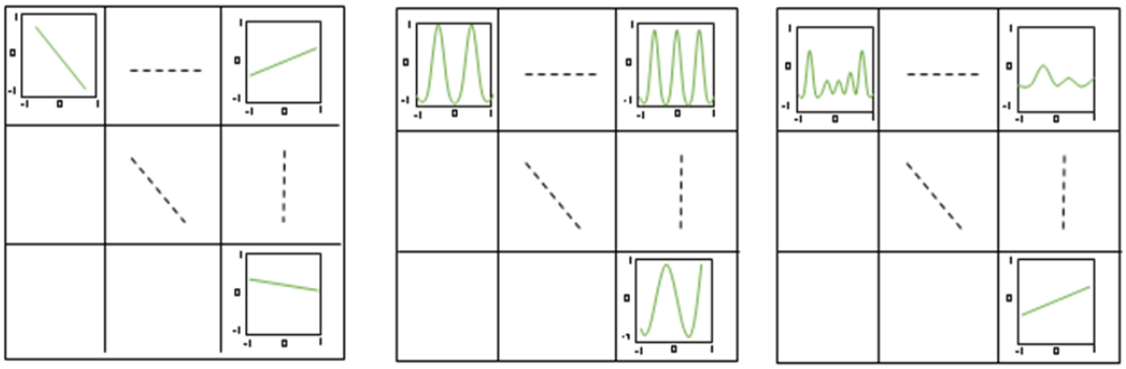


Figure 5. Kernels in CNNs (left), ONNs (middle), and Self-ONNs (right).

detailed information and details on the BP formulations, the readers are referred to [9].

3.8 Conclusion

This chapter presented the theoretical background of NNs, RNNs, and their variants, CNNs, and the recent advances in NNs. In particular, the ONNs and Self-ONNs with generative neurons play a significant role in this dissertation. This dissertation will present the use of these advanced models to tackle many challenges related to handwriting recognition in historical Arabic manuscripts in the coming chapters.

CHAPTER 4: SUBWORD RECOGNITION USING CNN-GRU

Handwriting recognition techniques are typically categorized into segmentation-free (holistic) and segmentation-based (analytical) approaches. Segmentation-based techniques rely on segmenting the words into primitive units (subwords or characters) for recognition (**Figure 6** (a)) while segmentation-free techniques employ words as units of recognition (**Figure 6** (b)). Recognition of subwords has several advantages over holistic word recognition. Firstly, since subwords are disconnected parts of a word, it is easier to extract them as compared to complete words especially in case of cursive scripts like Arabic. Secondly, same subwords are shared across multiple words (**Figure 6** (c)) and the total number of unique subwords is much less than the total number of unique words. In other words, the size of the dictionary of complete words is larger than that of the subwords, hence using subwords as units of recognition reduces the number of unique classes to be recognized. Another advantage of subword recognition is that it can be extended for handwriting recognition in other languages that share the same script as Arabic e.g., Urdu, Persian, etc. Few examples of the words shared between Arabic and Persian are

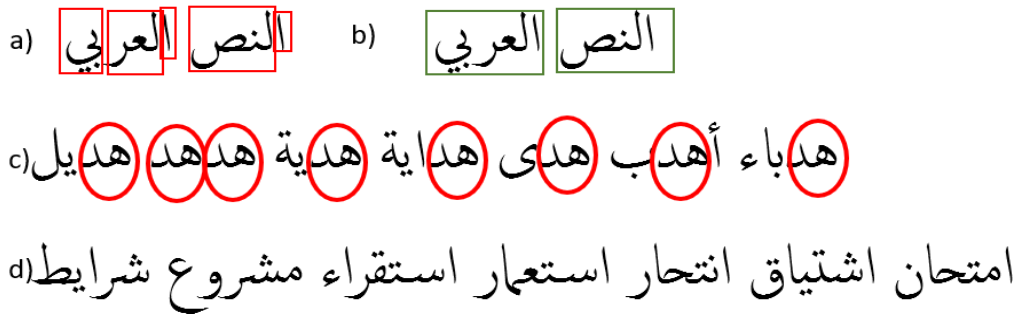


Figure 6. (a) Arabic subwords. (b) Complete Arabic words (c) Arabic words with shared subwords. (d) A sample of Arabic words that are used in other non-Arabic languages.

shown in **Figure 6** (d). Finally, subword recognition is known to report higher recognition rates as opposed to word recognition [67].

We propose a deep learning-based technique to recognize a substantial vocabulary of Arabic handwritten subwords extracted from historical documents. More specifically, we propose a hybrid CNN-GRU model that learns to discriminate between a large number of subword classes. The key highlights of this study are listed in the following.

- An end-to-end system is proposed for recognition of challenging Arabic handwritten subwords extracted from historical manuscripts.
- A hybrid CNN-GRU architecture is introduced with shallow convolutional layers extracting robust features from subwords while the GRU layers learn to map the feature sequences to subword class labels.
- Experimental study is carried out on two benchmark datasets IBN SINA and VML-HD and recognition rates outperforming current state-of-the-art are reported.
- Unlike previous studies which consider only a subset of VML-HD dataset, the complete dataset has been employed for evaluation purposes in our experiments. To the best of our knowledge, no previous study has considered such diverse subword classes to evaluate the proposed models.

4.1 The proposed subword model

This section presents the details of the proposed subword recognition technique that relies on extraction of robust representations using convolutional layers followed by sequence modelling using Gated Recurrent Units (GRUs). An overview of the CNN-GRU architecture employed for recognition of subwords is illustrated in **Figure 7**. CNNs are known to be state-of-the-art feature extractors outperforming hand-

engineered features on a number of classification tasks. Likewise, recurrent neural networks are known to effectively model sequential data.

While the basic RNNs are known to suffer from the vanishing gradient problem when modeling long-term dependencies, variants like LSTMs and GRUs are commonly employed. These variants rely on gates to regulate the flow of information from one time step to another.

The employed architecture comprises of two convolutional layers. The input subword image (40×64×3 for images in the IBN SINA dataset) is fed to the first convolutional layer with 64, 3×3 filters producing an output of 40×64×64. The feature maps of the first convolutional layer are fed as input to the second convolutional layer which produces an output of 40×64×256. A max-pooling layer reduces the spatial dimensions of the output volume by 2 producing a volume of 20×32×256. Batch-normalization is also added to the network to speed up training by normalizing the activations while a dropout layer is included to mitigate over-fitting. Prior to feeding the activations to the GRU layers, the data is reshaped to 20×8192. A stack of four Bi-directional GRU layers is added after the convolutional layers. The outputs are then concatenated and fed to the next GRU layer. The output of the GRU layers is finally flattened and fed to a fully connected softmax layer for classification. The same architecture is employed for the VML-HD dataset, we only changed the input image size to (110×110).

The model is implemented in TensorFlow on TitanX GPU with 12 GB VRAM. Model was trained using ADAM optimizer, the learning rate was set to 0.001 while the batch size was set to 32. The model was trained for 350 epochs.

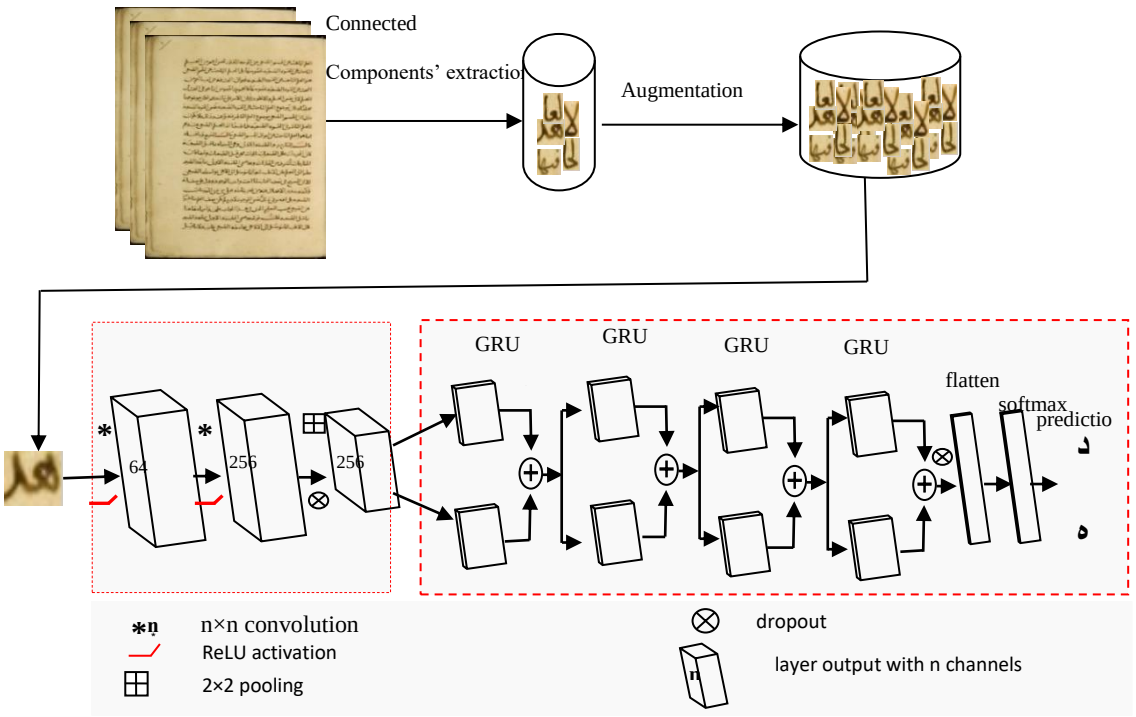


Figure 7. An overview of the CNN-GRU architecture for Arabic subword recognition. The Conv layers serve as feature extractors while the stacked GRU layers serve as recognizers.

4.2 Experiments

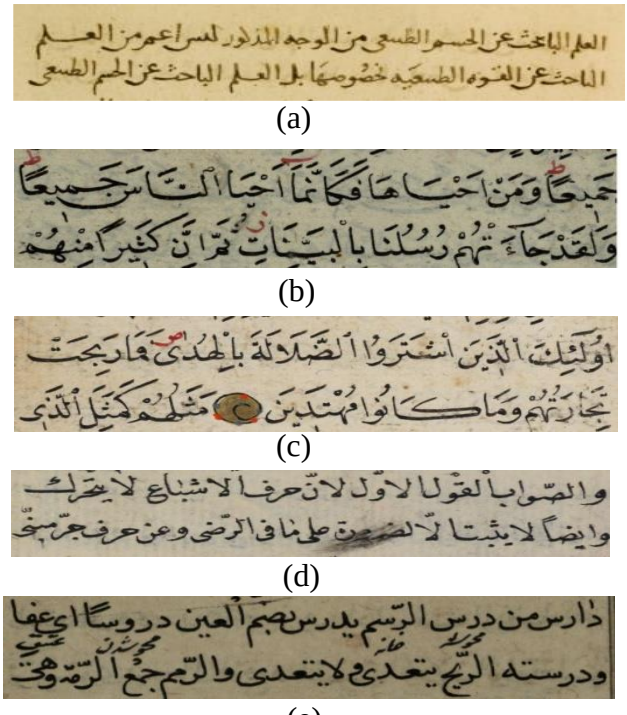
This section presents the details of the experiments along with the reported results. We first introduce the datasets employed in our study followed by a discussion on the results.

4.2.1 Subword-level datasets

One of the well-known datasets compiled for research on historical Arabic manuscripts is the IBN SINA dataset which contains 50 folios of ancient Arabic documents from KitabKashf al-Tamwihatfisharh al-Tanbihat ((**Figure 8** (a)). Following the same experimental protocol as that of previous studies, we employ the first 40 folios for training and the last 10 folios for testing. Furthermore, we also ignore the dots similar to [50]. The second dataset, VML-HD, is recently published and is relatively much larger comprising of 680 pages from five different books written by different writers in diverse writing styles. Since there is no published work using the

full version of this dataset yet, we employed 60% of images for training, 20% for validation and 20% in the test set.

The success of deep learning-based solutions is mostly linked with the amount of data employed for training. Most of the current research in object recognition using deep CNNs makes use of data augmentation to increase the size of training data and avoid over-fitting [68] [69]. Data augmentation can also be employed to address the class imbalance problem [70]. In our study, we employ shear transform and rotation (up to certain degrees) to augment the subword images. For each subword image in the IBN SINA dataset, we generate 36 images resulting in a total of 179,027 images in 933 different categories. Since the VML-HD data is relatively larger and few classes have thousands of examples, we apply data augmentation only to classes having less than 10 samples. This results in a total of 374,161 subwords in 6187 different categories.



(a) (b) (c) (d) (e)

Figure 8. Samples of the datasets used. (a) Sample from the IBN SINA dataset. (b) to (e) Samples from the VML-HD dataset.

4.2.2 Results & discussion

The performance of the proposed CNN-GRU architecture on both datasets is presented in **Table 3**. It can be observed from the reported recognition rates that though IBN SINA is a single writer dataset, the recognition rate is slightly lower (96.13%) as compared to that on the VML dataset (98.60%). This can be attributed to the relatively smaller training set in case of IBN SINA dataset as well as high degradation in the manuscripts. In some cases, small parts of subwords are lost leading to incorrect classification. When compared to other known methods evaluated on the IBN SINA dataset, a significant performance enhancement is observed as opposed to conventional classifiers like SVM and Nearest Neighbour.

Likewise, the previous studies employing VML dataset used only a part of the dataset for evaluation of the proposed techniques. In our experiments, we employ the complete dataset with more than six thousand subword categories and the proposed model still outperforms the existing techniques. The high recognition rates reported on two different datasets support the potential application of such a system in automatic transcription of historical manuscripts. This could eventually lead to true digital libraries making them searchable for the end users.

Figure 9 illustrates the subwords categories which reported a recognition rate of less than 100%. As discussed earlier, the degradation of manuscripts and incorrect segmentation of subwords are the major factors contributing to recognition errors as morphologically similar subword classes tend to overlap. As an example, the subword class `kr' in **Table 4**. has lost the upper stroke of the first letter making it resemble another subword `lr' hence leading to an incorrect classification. Likewise, similar problems can occur with other subword classes resulting in incorrect recognition.

Table 3. Recognition results in IBN SINA and VML-HD datasets.

Study	Method	Dataset	Accuracy
R. Alaasam et al. [43]	CNN	Part of VML-HD	81%
R. Alaasam et al. [44]	CNN	Part of VML-HD	97.82%
Chherawala et al. [45]	1-NN	IBN SINA	86.16%
Moghaddam et al. [49]	SVMs	IBN SINA	89.66%
Chherawala & Cheriet [47]	1-NNs	IBN SINA	86.2%
Fouladi et al. [50]	Contour Matching By Alignment	IBN SINA	91.08%
Proposed Method	CNN and GRU	IBN SINA	**96.13%**
Proposed Method	CNN and GRU	Full set of VML-HD	**98.6%**

Table 4. Classification results. The first three rows show incorrect classification while the last three rows show the correct classification.

Input Image	True class	Predicted Class
 kr (كر)	 Kr (كر)	 lr (لر)
 v (و)	 v (و)	 h (ه)
 rh (رح)	 rh (رح)	 h (ح)
 bma (بما)	 bma (بما)	 bma (بما)
 llh (لله)	 llh (لله)	 llh (لله)
 qSa (قصا)	 qSa (قصا)	 qSa (قصا)

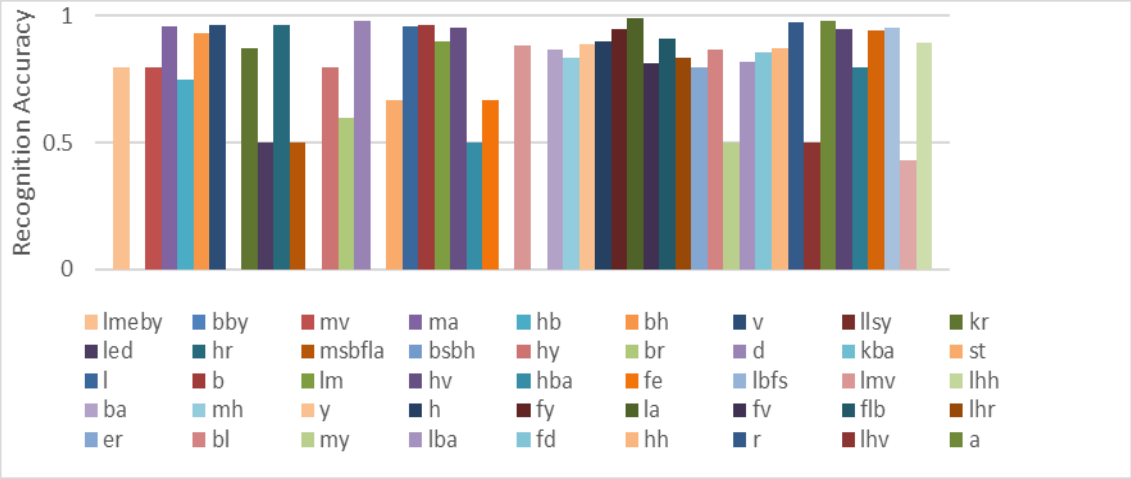


Figure 9. Categories of subwords that gained less than 100% accuracy.

4.3 Conclusion

Recognition of Arabic handwriting from historical manuscripts is a challenging problem due to varying writing styles, document degradation and segmentation issues. The current study addressed this problem using a hybrid CNN-GRU model to recognize Arabic subwords extracted from historical documents. The proposed model was evaluated on two different datasets, IBN SINA and VML-HD and reported high recognition rates of 96.10% and 98.60% on the two datasets respectively. A comparison with existing techniques revealed the superiority of the proposed architecture in terms of recognition rates. The proposed solution can also be adapted to other cursive scripts like Persian, Kurdish and Urdu etc. In our subsequent endeavours on this subject, we aim to include a post-processing step to validate the predictions of the model. Furthermore, in addition to subwords, retrieval using complete words can also be added to allow more useful queries at the application layer.

CHAPTER 5: HANDWRITING RECOGNITION USING SELF-ONNS WITH GENERATIVE NEURONS

The majority of current offline HTR algorithms work at the line-level by converting the text-line image into a series of feature vectors which are fed into an optical model, such as RNN, to recognize handwritten text. Although document-level text detection and localization and paragraph-level joint line segmentation and recognition yielded encouraging results, the best recognition results, however, are still obtained by systems that work at the line-level [71].

In this work, we propose a dedicated use of Self-ONNs in HTR as an alternative to the traditional CNN-based methods. We investigate an adequate level of non-linearity of the Self-ONN layers to provide extensive information on the Self-ONN performance under various topologies. We further investigate the use of deformable convolutions along with the generative neurons in the same network. With such a novel approach, we achieve the state-of-the-art performance with a significant performance gap over all recent methods.

5.1 The Proposed System

To investigate the impact of using a heterogeneous and non-linear network model in HTR, we performed certain modifications over the recently proposed CNN-only HTR system [41] that currently holds the state-of-the-art HTR performance. **Figure 10** depicts an overview of the architecture used in [41]. In this section, we go over the modifications we made on the blocks and some additional system features.

5.1.1 Architecture

The proposed architecture consists of two parts; the backbone and the head. The backbone consists of a group of ResnetBlocks [72] blocks and acts as an optical model responsible for transforming the input images into feature maps. Each block contains either 2D-CNNs or 2D-Self-ONNs with 3 × 3 kernels, 1 × 1 stride, 1 × 1 padding, and 1 × 1 dilation. The number of filters in each group of blocks is twice the number of filters in the previous group of blocks. Each CNN or Self-ONN layer is followed by batch normalization and each group of blocks is followed by max pooling. The original architecture in [41] uses a total of 10 convolutional blocks. Here we only use 7 blocks to further reduce the complexity and we evaluate both configurations. Similar to the way deformable convolutions are injected in [41], we injected operational layers with generative neurons of Self-ONNs by replacing the convolutional layers/neurons in ResNet blocks as demonstrated in **Figure 10** (b).

The feature maps extracted from the convolutional backbone are then fed into the convolutional or operational head to be transformed into character predictions with the help of either 1D-CNN or 1D-Self-ONN. The convolutional or operational head consists of several CNNs or Self-ONNs, each one is followed by batch normalization and a ReLU non-linearity (in the case of CNN) or Tanh (in the case of Self-ONN). We generate a sequence of probability distributions over the potential characters using the softmax function on the final output, which is then propagated into a Connectionist Temporal Classification (CTC) loss [73].

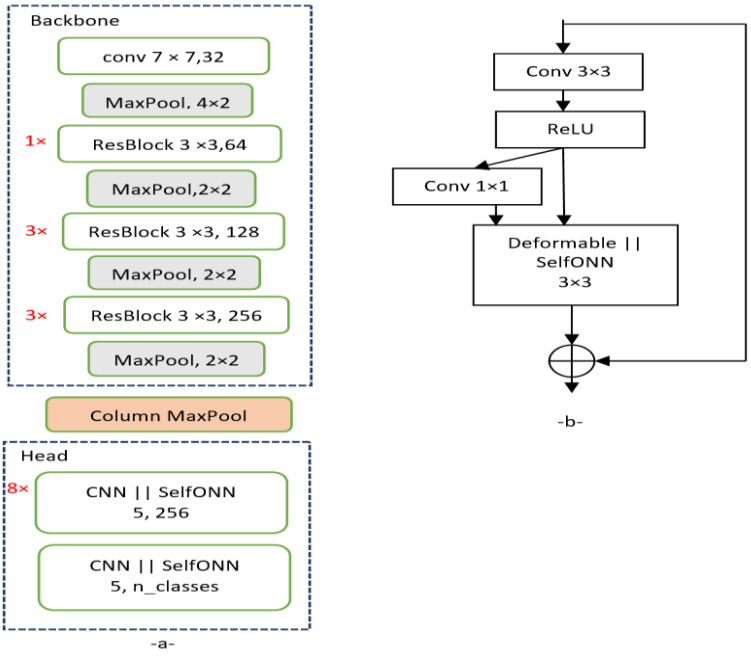


Figure 10. (a) System architecture consisted of a Backbone that contains either CNNs or Self-ONNs and a Head (1D-CNNs or 1D-Self-ONNs). (b) ResBlock variation for Self-ONNs or deformable convolutions.

5.1.2 Implementation Details

To test our approach, we use a widely used line-level dataset IAM [7]containing a total of 9,862 text lines. It includes one set of training, one set of testing, and two sets of validation. We use also HADARA80P [5] dataset which is based on an Arabic historical handwritten book. It contains 1,336 lines. We use 80% of the dataset for training, 10% for validation and 10% for testing as this splitting method adopted in several research papers [74]–[76].

Because the Hadara80P dataset is relatively different and much smaller than the IAM dataset, we train it for 120 epochs only, while for the IAM dataset, we train it for 2000 epochs. Adam optimizer is used to train the model with a maximum learning rate of 4e−5 (for IAM) and 1e-5 (for Hadara80P) and the batch size set to 12.

The Word Error Rate (WER) and Character Error Rate (CER) are the

evaluation metrics used in this study, both of which use the Levenshtein Distance [22] between the predicted text and the target text. The ratio of misrecognized characters is represented by CER, whereas the ratio of misrecognized words is represented by WER.

Three types of errors need to be considered when calculating the CER and WER: the substitution error (S) is the misspelled characters or words, the deletion error (D) is the lost or missing characters or words, and the insertion error (I) is the incorrect inclusion of characters or words. The following formula describes the common calculation of CER:

$$CER = \frac{S+D+I}{N}, \quad (20)$$

where N is the number of characters in the ground truth. The WER formula is similar to CER's one, but at the word level.

$$WER = \frac{S_w+D_w+I_w}{N_w} \quad (21)$$

If the Levenshtein distance between two words is not zero, the word is considered incorrectly classified even if only one character is incorrect. This evaluation includes all symbols including special characters. **Figure 12** and **Figure 11** show samples taken during testing the proposed model along with CER and WER.

5.2 Experimental Evaluation

5.2.1 Evaluating The Self-ONN Optical Model

In the first experiment, we compare the performance of CNNs versus Self-ONNs on the original architecture proposed in [41] which consists of 10 blocks in the backbone and three convolutional layers in the head. **Table 5** illustrates the comparison of accuracy after replacing the CNN layers in the head with Self-ONN

Table 5. Comparison of the performance of CNNs, Self-ONNs, and deformable convolutions on the IAM dataset using the original architecture proposed in [41].

Configuration		Q-order	Best CER		Best WER	
Backbone	Head		CER	WER	CER	WER
CNN	CNN	-	5.424	18.914	5.171	17.894
CNN	Self-ONN	3	5.128	17.982	5.128	17.982
CNN	Self-ONN	5	5.145	17.858	5.145	17.858
CNN	Self-ONN	7	5.202	18.178	5.202	18.178
CNN	Self-ONN	9	5.270	18.258	5.270	18.258
CNN	**Self-ONN**	**3,5,7**	**5.075**	**17.589**	**5.075**	**17.589**
CNN+DeformableCNN	Self-ONN	3,5,7	5.156	17.771	5.151	17.654

layers with Q = 3, 5, 7 and 9 (Q is the order of the Taylor polynomials). We report the performance of the models with the best CER and the WER. Self-ONNs consistently outperform both CNNs and deformable CNNs in terms of CER and WER.

We then trim the model by removing three blocks from the convolutional backbone that leads to a reduction of 6 CNN layers we tested this compact architecture on HADARA80P dataset because it is smaller in size compared to IAM dataset. The results in **Table 6** show that using SelfONNs on the backbone was giving better results than using it on the head. The results also show exceptional accuracy improvement (3.464 % and 1.2 % WER and CER accuracy improvement respectively) when using SelfONNs compared to CNN only architecture.

We further improve the model for IAM dataset and added the removed layers to the head leading to a total of 9 layers in the head. The results are presented in **Table 7**. This new model where layers in the feature extraction part (backbone) are reduced and layers in the classification part (head) are populated leads to a better CER or WER

than the ones reported in [41]. Following this, using operational layers of Self-ONNs in the feature extraction part exhibits even better results than using only CNNs or CNNs with deformable convolutions. Finally, the combination of Self-ONNs with deformable convolutions achieves an exceptional improvement in both CER and WER and thus setting a new state-of-the-art performance level in this domain.

An important observation worth mentioning is that although the use of the three neuron types (convolutional, deformable convolutional, operational) improved the performance, the location of each one in the network architecture also had an important effect. By looking at the results in 1, we find that although the three networks were used (CNN and deformable in the backbone while Self-ONN in the head), the result was not the best.

Table 6. Comparison of the performance of CNNs, Self-ONNs (Q = 3,5,7 in all), and Deformable Convolutions on the HADARA80P dataset using only 7 blocks on the backbone.

Configuration		Best CER		Best WER	
Backbone	Head	CER	WER	CER	WER
CNN	CNN	9.199	35.912	9.191	35.460
CNN+DeformableCNN	CNN	9.639	37.216	9.639	37.216
CNN	Self-ONN	12.486	46.178	12.486	46.178
Self-ONN	**CNN**	**7.977**	**32.038**	**7.991**	**31.996**
Self-ONN+DeformableCNN	CNN	11.764	46.010	12.145	46.681

Table 7. Comparison of the performance of CNNs, Self-ONNs (Q = 3,5,7 in all), and Deformable Convolutions on the IAM dataset using the proposed architecture in Figure 10

Configuration		Best CER		Best WER	
Backbone	Head	CER	WER	CER	WER
CNN	CNN	4.799	16.287	4.731	16.324
CNN+DeformableCNN	CNN	4.914	16.171	4.914	16.171
CNN	Self-ONN	4.737	16.164	4.895	16.2583
Self-ONN	CNN	4.732	16.033	4.794	16.127
Self-ONN+DeformableCNN	**CNN**	**4.576**	**15.488**	**4.532**	**15.080**

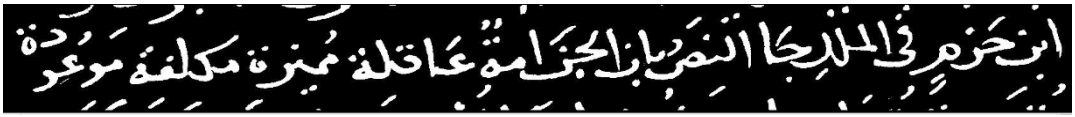
5.2.2 Performance Evaluations

As discussed earlier, several deep learning models of HTR were proposed in the literature including LSTM-Based approaches, attention-based approaches and sequence-to-sequence transition approaches in addition to the recently proposed CNN-only approaches. For a detailed set of comparative evaluations, the performances of the proposed methods are compared against all recent state-of-the-rt methods (lexicon-free in line-level) in IAM dataset and presented in **Table 8**. The proposed method outperforms all prior works. The proposed Self-ONN architecture combined with deformable convolution improved the CER by 0.14% and WER by 1.49% over [41].

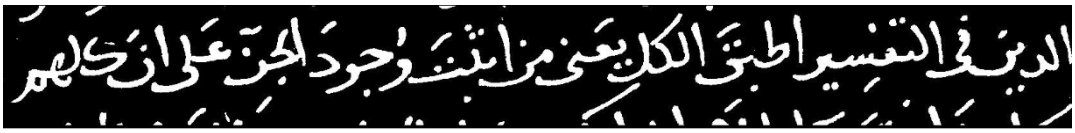
The HADARA80P dataset was mainly used in word spotting systems thus the reported results are at word-level. In [77] they used the line-level dataset with other non-historical Arabic datasets to train their system, however the accuracy of their system on HADARA80P dataset is not reported, instead, they reported the overall accuracy of the system using all datasets.

Table 8.Comparison with the state-of-the-art results on IAM dataset.

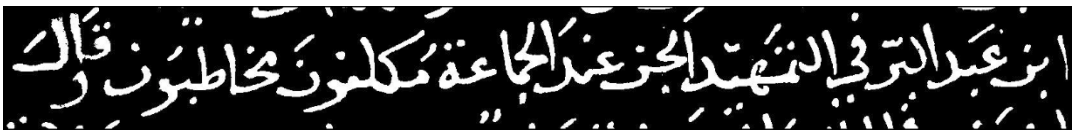
System	Method	CER	WER
Chenetal.[38]	CNN&LSTM	11.15	34.55
Phametal.[39]	CNN&LSTM	10.8	35.1
Khrishnanetal.[78]	CNN	9.78	32.89
Chowdhuryetal.[79]	CNN&RNN	8.10	16.70
Puigcerver[27]	CNN&LSTM	6.2	20.2
Markouetal.[80]	CNN&LSTM	6.14	20.04
Duttaetal.[81]	CNN&LSTM	5.8	17.8
Tassopoulouetal.[82]	CNN&LSTM	5.18	17.68
Yousefetal.[83]	CNN	4.9	-
Michaeletal.[37]	CNN&LSTM	4.87	-
Cojocaruetal.[84]	CNN&DeformableCNN	4.6	19.3
Retsinasetal.[41]	CNN&DeformableCNN	4.67	16.57
Proposed	**CNN&DeformableCNN&Self-ONNs**	**4.53**	**15.08**



Actual Text: ابن حزم فى الملل جا النص بان الجن امة عاقلة مميزة مكلفة موعودة
Predicted Text: ابن حزم فى المالـال جا النص بان الجن امة اعاقلة ماميزة مكلفة موعودا
Character Errors: 5.0

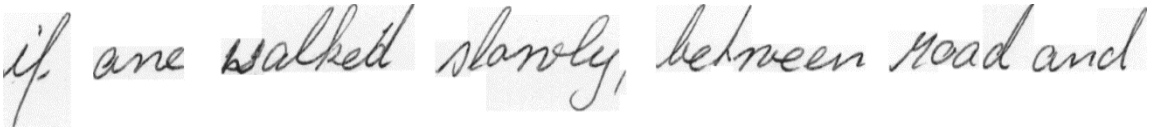


Actual Text: الدين فى التفسير اطبق الكل يعنى من اثبت وجود الجن على ان كلهم
Predicted Text: الدين فى التفسير اطبق الكل يعنى امن اثبت وجود الجن اعلى ان كلهما
Character Errors: 3.0
Word Errors: 3.0

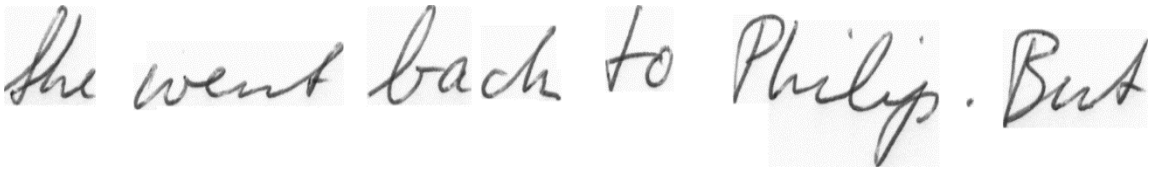


Actual Text: ابن عبد البر فى التمهيد الجن عند الجماعة مكلفون مخاطبون و قال
Predicted Text: ابن اعبد البر فى التمهيد الجن اعند الجماعة مكلفون مخاطبون و قا
Character Errors: 3. 0
Word Errors: 3.0

Figure 11. Sample of Model predictions of HADARA80P dataset with their CER and WER.



Actual Text: if one walked slowly, between road and
Predicted Text: if are walked slanely, between Noad and
Character Errors:6.0
Word Errors:3.0



Actual Text: She went back to Philip. But
Predicted Text: the went back to Philigs. But
Character Errors:3.0
Word Errors:2.0

Actual Text: mantle, and said with such
Predicted Text: mantle, and said with such
Character Errors:0.0
Word Errors:0.0

Actual Text: him go, unable to speak, she
Predicted Text: him go, unable to speak, she
Character Errors:0.0
Word Errors:0.0

Figure 12. Sample of Model predictions of IAM dataset with their CER and WER.

5.3 Conclusion

In this dissertation, a novel approach based on Self-ONNs is proposed for HTR. Reaching the state-of-the-art performance levels, the proposed approach benefits from the superior learning capabilities of the Self-ONNs that are heterogeneous network models with generative neurons. The previous top model proposed in [41] employs an uncertainty reduction method to improve the overall accuracy to 4.55% CER and 16.08% WER on the IAM line dataset. The proposed Self-ONN-based approach surpasses the original model even without employing any uncertainty reduction or any other post-processing whilst the network depth is further reduced. This study shows that the operational layers with generative neurons are able to represent complicated contextual information and handle HTR efficiently. The superiority of these layers is very obvious in the Arabic dataset which is more complex than the English Dataset. In our future work, we aim to explore thoroughly different training strategies for Self-ONNs and investigate the performance of Self-ONNs with RNNs. The optimized PyTorch implementation of Self-ONNs is publicly shared in [85].

CHAPTER 6: UNDOTTED TEXT DISAMBIGUATION USING SELF-ONNS

This chapter discusses a novel way for disambiguating undotted words. Here, we propose a solution assuming that dots are entirely lost, and we are trying to build a system that can read an undotted text and then predict the possible dots of characters in the word. The word I'jam (إعجام) refers to adding points or dots to a consonant to improve clarity and understanding. The proposed system (summarized in **Figure 13**) consists of an OCR model and a dots restoration model; the OCR model receives an image and returns the corresponding transcription. The dots restoration model (trained on a rich Arabic corpus) receives an undotted Arabic text as an input and returns the corresponding dotted text as an output result. The key highlights of this chapter are listed in the following.

- We propose a novel method within our knowledge to restore dots from undotted Arabic script. While the method is providing a solution for handwriting recognition systems that deal with Arabic manuscripts with old undotted scripts, it shows its success in recent Arabic handwritten documents which is having dotted and diacritized scripts.
- We analyze the effects of dots and diacritical marks on the performance of the OCR model.
- In addition to dots restoration, we improve on a recent diacritics restoration model by introducing Self-ONNs with generative neurons proposed recently in [9] to overcome the limitations of CNNs.

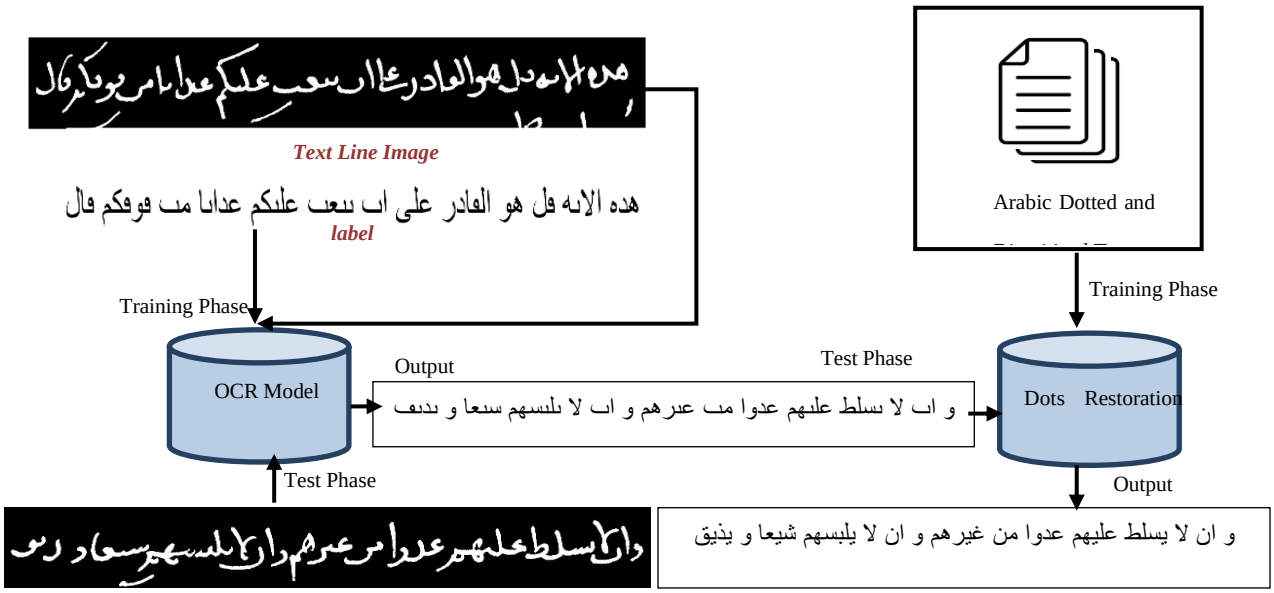


Figure 13. General overview of the proposed OCR and dots restoration system architecture.

6.1 Previous Diacritization Models

The earliest attempt to automatically restore diacritics in Arabic texts relied solely on rules. It either employed digital dictionaries or relied on human input to determine the linguistic rules required for diacritization. This category includes the majority of the online published tools for Arabic diacritizition, which commonly use morphological analyzers [3]. The difficult, costly, and time-consuming effort of generating and maintaining rules that encompass the rich linguistic characteristics of Arabic is a fundamental shortcoming of rule-based approaches. Furthermore, the development of these systems necessitates ongoing review and modifications by language experts [3].

The second category used simple statistical methods. Explicit language rules are not used in these methods. Statistical models, on the other hand, learn diacritization by estimating the likelihood of distribution of a series of words or characters from

diacritized texts. The third group is made up of several sorts of deep learning. The use of various types of Neural Networks to restore diacritization is the most recent trend in the field. Fadel et al. [86] compare the performance of publicly accessible rule-based systems with their neural-based technique Shakkala [87]. Their results demonstrate that the neural Shakkala system surpasses traditional rule-based techniques significantly.

The first RNN proposed to address the diacritization problem as a sequence transcription problem was proposed in [88]. They designed, trained, and tested a bidirectional LSTM network that takes raw undiacritized sequences as input and converts them to diacritized sequences. Prior to, or during data training, they did not perform any lexical, morphological, or syntactical analysis. Instead, they applied error-correction algorithms as a post-processing step on the network's output. Later, in [89], they examined various network architectures and found that the bidirectional LSTM network outperforms the encoder/decoder network and the unidirectional LSTM network. They also fine-tuned their model by employing a deeper network and dropout.

In neural machine translation, non-recurrent architectures such as convolutional and self-attentional have been shown to outperform RNNs. CNNs and self-attentional networks are considered better at modeling long-range dependencies because they can connect distant words over shorter network paths than RNNs [90]. On various Natural Language Processing tasks, Yin et al. [91] compare CNNs, LSTMs, and GRUs. They discovered that CNNs are better at tasks involving semantics, while RNNs are better at tasks involving syntax, particularly for longer sentences.

A combination of 1-D Convolution Bank, Highway network, and Bidirectional GRU network (CBHG) was proposed by [92] for a character-level Neural Machine

Translation (NMT) and adapted by [93]for building Tacotron text-to-speech model. In [94], the same network was adapted to restore diacritics in Arabic text. The performance of the network was compared to the other two networks. The first is a baseline model with an embedding layer of 512 dimensions, three bidirectional LSTM layers with 256 cells each, a fully connected layer to project to the output size, and finally, a softmax layer to output the probability for each diacritic. The second model is built on an encoder-decoder architecture. While the encoder-decoder solves the problem and outperforms many other systems in the literature, they discovered that the CBHG model is substantially faster to train and gets state-of-the-art results.

We further improve on this model and instead of using CNNs we use Self-ONNs (described in CHAPTER 3) with generative neurons demonstrating superior results compared to CNNs. We show that with a smaller number of layers and parameters, our model is able to yield better results than the original CNN-based model. In addition to diacritics restoration, we adapted the model to restore dots in an undotted Arabic text. To our knowledge, this is the first one of its kind model for dots restoration. As the diacritization of text helps in improving text-to-speech systems' results, the dots restoration will help in improving the Arabic HTR significantly.

6.2 Proposed Dots and Diacritics Restoration Method

Our goal in diacritics restoration is to generate a sequence of diacritics y = (y1,..., yn) from a sequence of input characters x = (x1,..., xn), where y1 is the diacritic of the character x1, which is picked from 15 potential values. The same way our dots restoration goal is to generate a sequence of dotted characters y = (y1,..., yn) from a sequence of undotted characters x = (x1,..., xn), where y1 is the dotted character of the undotted character x1, which is picked from 37 potential values. The goal is to

find a target sequence that maximizes the conditional probability of y given a source sentence x. **Table 9** shows the possible characters for our dots restoration and the possible diacritics for diacritics restoration. The first empty class ('') is assigned to punctuations and other non-Arabic characters found in the text.

In this study, we are aiming to improve on a recently proposed model for diacritics restoration in [95] and reduce the network complexity and depth significantly. We adapted the same model for dots restoration. The model (shown in **Figure 14**) consists of a bank of 1D convolutional filters, highway networks [96], and a bidirectional gated recurrent unit (GRU) Recurrent Neural Network. The input sequence is first convolved with K sets of 1-D filters, with the k-th set containing Ck filters of length (1, 2, . . . ,K). Uni-grams, bigrams, and k-grams are all represented by these filters. With a stride of 1, the output of the convolutional layer is sent to a max-pooling layer over time. The output of the max-pooling layer is fed into a series of 1-D convolution layers with residual connections to the original inputs. These convolution layers' output is sent into a multi-layer highway network. The last layer is a bidirectional GRU that extracts the input sequence's sequential features. To output the dots or diacritics, a fully-connected projection layer and a softmax layer on top of the CBHG module is added. We modified the model with a smaller number of layers (3 convolutional or operational layers only). In the case of injecting Self-ONN layers, we use Tanh activation function instead of Relu.

Table 9. Possible target Arabic letters and diacritics

All possible Letters										
‘’	ء	آ	أ	إ	ا	ئ	ؤ	ب	ة	ت
ث	ج	ح	خ	د	ذ	ر	ز	س	ش	ص
ض	ط	ظ	ع	غ	ف	ق	ك	ل	م	ن
ه	و	ى								
All possible diacritics										
‘’	ً	ٌ	ٍ	َ	ُ	ِ	ّ	ْ	ٓ	ٔ

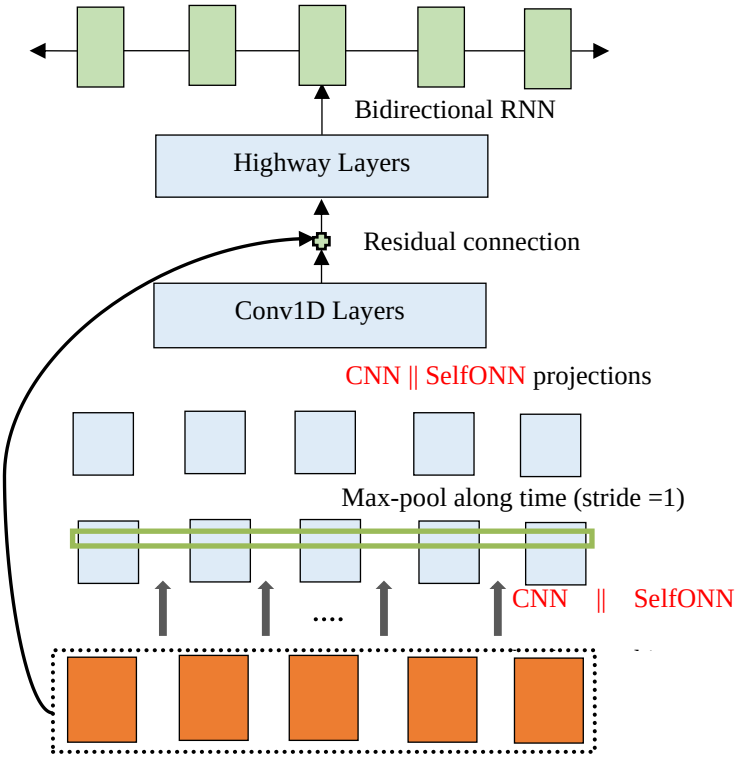


Figure 14. The architecture of dots restoration model taken from [110].

6.3 OCR Model

To evaluate the effect of dots on the OCR, we use a state-of-the-art model proposed in [41] and used for line recognition in English and French datasets IAM[7] and RIMES [97].

The architecture of the model proposed in [41] consists of a convolutional backbone

and a convolutional head. The convolutional backbone is comprises a collection of Resnet blocks [72] that serve as an optical model for converting input images into feature maps. We reduced the number of blocks in the backbone from 10 blocks to 7.

The feature maps acquired from the convolutional backbone are then sent into the convolutional head, which uses a 1D-CNN to translate them into character predictions. Each CNN in the convolutional head is followed by batch normalization and a ReLU non-linearity. We use the softmax function on the final output to construct a sequence of probability distributions over the potential characters, which is then propagated into a Connectionist Temporal Classification (CTC) loss [73].

6.4 Experiments

6.4.1 Dots and diacritics datasets

The dataset used in our diacritization and dots restoration is the cleaned version of Tashkeela corpus[1] prepared by [86]. They cleaned the dataset by solving some diacritization errors, such as misplaced diacritics and eliminating the first diacritic. They also cleaned up the data by deleting English characters, adding whitespaces to separate numbers from words, and removing multiple whitespaces. They also did some file formatting, like deleting tags from HTML files and eliminating URLs. The dataset consists of 55K lines containing around 2.3M words. Furthermore, 5K lines from the dataset are chosen at random and distributed evenly between the validation and testing datasets. More statistics on the dataset size and diacritics usage after the cleaning process can be seen in [86]. Because the dataset is having dotted and

[1] https://sourceforge.net/projects/tashkeela

diacritized Arabic letters, we remove the diacritics and use Algorithm 1 to convert the dotted characters into its corresponding undotted characters before training. We treated the variants of Hamza letters (آ، أ، ؤ، إ، ئ) as the dotted letters. The result of this algorithm is shown in **Table 10**.

We also use HADARA80P [5] dataset which is prepared for OCR tasks. This dataset was taken from the "badalu almaaun fi fadlu altaaun" manuscript, which dates back to the eighth century. The HADARA80P contains 80 images of a handwritten book and is annotated at the line and word level. We use 80% of lines' images for training, 10% for validation, and 10% for testing. We binarize the images and remove dots and diacritics using methods in [98]. We use the ground truth of the dataset to test our dots restoration model.

Algorithm 1 DotsRemoval Algorithm

Input: T - The Text to be Undotted
Output: UL- The Undotted Text and **DL** – The Dotted Text
1: **function** DotsRemoval(T)
2: Let **DL** be a list of dotted Arabic letters (اآأإبتثنيجخذزشضظغفقةؤئ)
3: Let **UL** be a list of undotted Arabic letters (ااااسسححدرسصطعههوى)
4: **for** c = 0 to T.length do
5: **if T**[c] in **DL**
6: add corresponding undotted character in **UL**
7: add T[c] in **DL**
8: **else**
9: add **T**[c] in **DL**
10: add **T**[c] in **UL**
11: **end if**
12: **end for**
13: **assert**(DL.length= UL.length)
14: **return DL,UL**
15: **end function**

Table 10.Results of removing dots before training using Algorithm1.

Dotted text	Undotted Text
قوله لأقل من ستة أشهر منه أي من وقت العتق ، فلو جاءت به لستة أشهر فصاعدا منه لا يعتق إلا أن يكون حملها توأمين جاءت بأولهما لأقل من ستة أشهر ثم جاءت بالثاني لستة أشهر أو أكثر ، أو تكون هذه الأمة معتدة عن طلاق أو وفاة فولدت لأقل من سنتين من وقت الفراق ، وإن كان لأكثر من ستة أشهر من وقت الإعتاق حينئذ فيعتق لأنه كان محكوما بوجوده حين أعتقه حتى ثبت نسبه ، وعلى هذا فرع ما لو قال ما في بطنك حر ثم ضرب بطنها فألقت جنينا ميتا ، إن ضربها بعد العتق لأقل من ستة أشهر تجب دية الجنين لأبيه إن كان له أب حر لأنه حر ، وإن لم يكن تكون	قوله لاقل مب سبه اسهر مبه اى مب وقب العبف ، فلو حاءب به لسبه اسهر فصاعدا مبه لا بعبف الا اب بكوب حملها بوامبب حاءب باولهما لاقل مب سبه اسهر بم حاءب بالبابى لسبه اسهر او اكبر ، او بكوب هده الامه معبده عب طلاف او وفاه فولدب لاقل مب سببب مب وقب الفراف ، واب كاب لاكبر مب سبه اسهر مب وقب الاعباف حببد فبعبف لابه كاب محكوما بوحوده حبب اعبفه حبى ببب بسبه ، وعلى هدا فرع ما لو فال ما فى بطبك حر بم صرب بطبها فالفب حببا مبا ، اب صربها بعد العبف لاقل مب سبه اسهر بحب دبه الحبب لابه اب كاب له اب حر لابه حر ، واب لم بكب بكوب

6.4.2 Evaluation

Diacritic Error Rate (DER) and Word Error Rate (WER) are two typical measures for evaluating diacritization systems. Both errors can be generated either with or without considering the word's last character (Case-Ending (CE)) in the calculation because this case largely relies on grammatical rules. The error rates for both measurements without CE are mostly lower than the error rates with CE. The DER calculates the percentage of characters that were not appropriately diacritized given all characters and their correct corresponding diacritics. All characters, including punctuations and spaces, are calculated using Equation 22. Characters that can be diacritized with several diacritics are handled as a single diacritic. The last character of each word is not taken into account when calculating the error rate without CE.

$$DER = \frac{D_{ic}}{D_{ic}+D_c} \times 100 \qquad (22)$$

where, Dic is the number of incorrect diacritics, while Dc is the number of correct diacritics.

The WER calculates the percentage of words with at least one diacritical error. In order to measure the percentage of unequal words, we compare all words from the

original file with the words from the predicted file using the equation:

$$WER = \frac{W_{ic}}{W_{ic}+W_c} \times 100 \quad (23)$$

where, W_{ic} is the number of unequal words while the W_c represents the number of equal words.

To evaluate our dots restoration model, we calculate Dotting Error Rate (DotER) similar to DER calculation. The WER represents the percentage of words with at least one dotting error. In the case of DotER the D_{ic} is representing is the number of incorrect dotted letter.

The OCR model is evaluated using the Word Error Rate (WER) and Character Error Rate (CER), both of which use the Levenshtein Distance [99] between the predicted text and the target text. The ratio of misrecognized characters is represented by CER, whereas the ratio of misrecognized words is represented by WER.

6.5 Results and Discussion

6.5.1 Diacritics Restoration

As we mentioned earlier, we reduced the model proposed in [94]. Instead of using 18 CNN layers, we used only three layers; then we replaced the CNN layers with SelfONN layers as described in **Table 11** with different Q settings. The results show that using only one SelfONN layer for feature extraction followed by CNN layers is the optimum configuration. We reported the results of DER and WER with case-ending and DER* and WER* without case-ending.

The results in **Table 12** show that Self-ONN is significantly achieving better error rates than those achieved by the full CNN model, except for the WER without case-ending, where the difference between the two models is 0.04, which is a slight

decrease in accuracy. According to the results shown in **Table 13**, we show that with a smaller number of parameters, the proposed model surpasses the CBHG model and improves most error rates.

Table 11. Results of reducing the number of convolutional layers from 18 to 3 layers and replacing CNN layers with SelfONN layers.

	DER	WER	DER*	WER*
CNN layer followed by two SelfONN layers with Q=7	2.38	9.3	1.8	5.66
3 SelfONN layers Q = 3,5,7	2.35	9.16	1.77	5.53
SelfONN layer with Q = 3 followed by 2 CNN layers	2.12	8.14	1.6	4.95

Table 12. Results of the CNN-based model in [110] and the proposed SelfONN model.

	DER	WER	DER*	WER*
CBHG [94] with 18 CNN Layers	2.16	8.23	1.63	**4.91**
SelfONN layer with Q = 3 followed by 2 CNN layers	**2.12**	**8.14**	**1.6**	4.95

Table 13. Total number of parameters and results of the proposed SelfONN models versus the original model in [94]

Model	Number of Parameters	DER	WER	DER*	WER*
CBHG [94]	15150353	2.16	8.23	1.63	**4.91**
CNN layer followed by 2 SelfONN layers with Q=7	6000401	2.38	9.3	1.8	5.66
3 SelfONN layers Q = 3,5,7	5934865	2.33	9.11	1.74	5.45
SelfONN layer with Q = 3 followed by 2 CNN layers	**4951825**	**2.12**	**8.14**	**1.6**	4.95

6.5.2 Dots Restoration

We adapted diacritization model to predict possible letter dotting. We use model architecture with the best results for diacritization, which is an updated version of CBHG [95] by using one SelfONN layer followed by two CNN layers. We use the same dataset used for diacritization in train and test splits and the results shown in **Table 14** reveal that our proposed method is successful in generating dotted text from a completely undotted text with high accuracy. We trained our model using only Tashkeela and tested it using the Tashkeela test set and Hadara test set. The results show that the model is able to predict undotted characters (with 70.07 % accuracy of word dotting and 89.46% accuracy of character dotting) in the HADARA80P although not including it in the training set. The model's accuracy on the HADARA80P test set improved dramatically (reaching 99.59 % accuracy of word dotting and 99.89% accuracy of character dotting) when adding the HADARA80P training set Tashkeela. Table 15 shows samples where the model fails to predict correct letter dotting.

Table 14. The results of the dots prediction model with 1 SelfONN and 2 CNNs.

Train Set	Test Set	WER	DotER
Tashkeela	Tashkeela	5.04	1.65
Tashkeela	HADARA80P	29.93	10.54
HADARA80P + Tashkeela	HADARA80P	0.41	0.11

Table 15. Samples of misclassified characters.

Original	Target	Prediction
موابد	موائد	موايد
ومربد	ومزيد	ومريد
مبرله	منزلة	منزله
بحل	نخل	تخل
فرس	فرس	فرش
سسب	سنين	ستين
بمر	ثمر	تمر

6.5.3 OCR Results

We conduct different experiments to test the effect of dots and diacritics on the OCR results. Figure 15 shows the image before and after removing dots and diacritics at the image and label level. We evaluate our model in different scenarios:

- Using the binary image without removing the dots and diacritics (**D**otted **I**mages and **D**otted **L**abels (DIDL)).
- Using the binary image after removing dots and diacritical marks while keeping them in the label (**U**n-Dotted **I**mages and **D**otted **L**abels (UIDL)).
- Using the binary image after removing dots and diacritical marks and removing them from the label (**U**nDotted **I**mages and **U**nDotted **L**abels (UIUL)).

The results shown in Table 16 show the effects of removing dots and diacritics at image and label levels on the OCR performance. The best results are achieved when removing these marks at the image and label level. A significant gap in both CER and WER accuracy between DIDL and UIDL scenarios, as depicted in Figure 16. This gap confirms the need for a robust dots restoration model we propose in this dissertation. In addition, in the case of totally undotted and undiacritized manuscripts such as the one in Figure 1, the need becomes more clear.

We implemented our experiments using PyTorch machine learning library. Our reported experiments were run on 2.60Ghz Intel® Xeon(R) CPU E5-2623 v4 with 125.8GB RAM and NVIDIA GeForce GTX 1080 graphic card.

Table 16. Results of testing the OCR model on different image and label setups.

	CER	WER
DIDL	9.19	35.46
UIDL	6.61	27.93
UIUL	5.75	25.03

(a)

(b)

Figure 15. (a) Image with dots and diacritical marks along with dotted ground truth (label). (b) Image without dots and diacritical marks along with undotted ground truth (label).

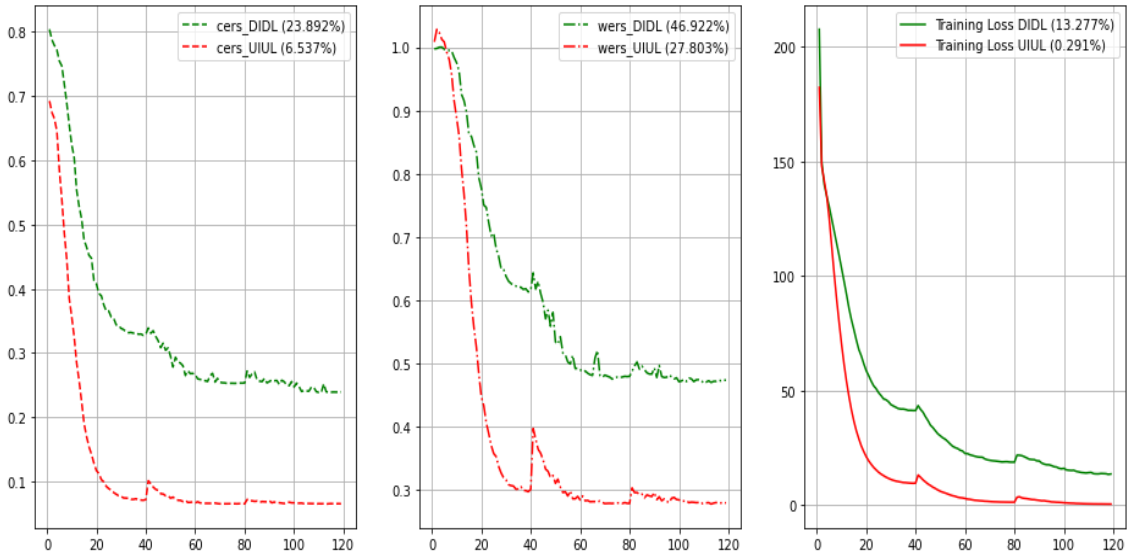


Figure 16. CER, WER, and Loss results of the OCR model using DIDL and UIDL scenarios.

6.6 Conclusion

In this chapter, we propose a compact model to tackle the problem of Arabic diacritization, which is considered among the challenging tasks in Arabic NLP. Our proposed Self-Organized ONN with generative neurons model can solve both diacritization and letter dotting with an exceptional accuracy. To our knowledge, this is the first dots prediction model and is expected to improve the current OCR systems, especially in historical Arabic documents. We tested our model on the HADAR80P dataset, which contains manuscripts with minor degradation; thus, not too many letters lost their dots and diacritical marks. We show the effect of dots and diacritical marks' removal on the OCR improvement. A further experiment is needed in a wide range dataset that includes the first Islamic centuries, where the scripts had no dots and no diacritical marks, such as KERTAS dataset [100].

CHAPTER 7: CONCLUSIONS AND FUTURE WORK

After investigating different aspects of document recognition, in this chapter, we summarize the research done in this dissertaion and identify some research directions that can assist in developing efficient handwriting recognition or keyword spotting systems capable of processing a large number of documents.

7.1 Conclusions

The first chapter of this dissertation highlighted the motivation behind this dissertation and its major contributions. CHAPTER 2 covered the related work and the state-of-the-art methods in handwriting recognition systems. CHAPTER 3 described the theoretical background of Neural Networks and the recent advanced models used to overcome the limitations of MLP and CNNs.

In CHAPTER 4, an efficient subword recognition model based on CNN and GRU is presented. The model is tested in a wide range of Arabic subwords from both IBN SINA and VML-HD datasets and showed high accuracy. Compared to previous results in both datasets , the proposed model results show better performance with a 5.05 % accuracy gap in the IBN SINA dataset [50] and a 0.78 % accuracy gap in the VML-HD dataset [44]. Another advantage of our approach is that it can be extended for handwriting recognition in other languages that share the same script as Arabic, e.g., Urdu, Persian, etc.

CHAPTER 5 presented an advanced use of Self-ONNs in HTR. The proposed method improved on the state-of-the-art CNN-only approach with the help of Self-ONNs. This approach achieved state-of-art results on the IAM English dataset [41] with 0.14% CER and 1.49% WER reductions. Testing the model using the line-level

HADARA80P Arabic dataset produced an exceptional result with 1.2% CER and 3.4 % WER reductions. The exceptional margin between the results of the CNN model versus the Self-ONN model in the noisy HADARA80P dataset confirms the superior learning capabilities of Self-ONNs in such old and degraded manuscripts.

CHAPTER 6 presented a novel method within our knowledge to restore dots from undotted Arabic script. While the method provides a solution for handwriting recognition systems that deal with Arabic manuscripts with old undotted scripts, it shows its success in recent Arabic handwritten documents with dotted and diacritized scripts. In addition to image processing accuracy, using our dots and diacritics restoration model, the integrity of the historical information extracted is preserved. This validates historical data and can provide us with new discoveries of the past.

7.2 Future Directions

After exploring different aspects of handwriting recognition in historical documents, we highlight here some research directions that will help in building efficient HTR systems that are capable of processing vast amounts of documents.

- The amount of digitized Arabic documents is much greater than those prepared for research due to the difficulty of dealing with ancient manuscripts in terms of segmentation and preparing accurate transcription. Preparing a comprehensive dataset that covers different eras with a full transcription at the word, line, and paragraph level would help in boosting the performance of current recognition systems.
- This work has so far been limited to the recognition of segmented words or lines extracted from historical documents. In fact, this is the most crucial step in building a successful recognition system. The challenge of

simultaneous object detection and classification is addressed by state-of-the-art deep learning approaches [101], [102]. These approaches could be adapted to cope with the nature of text images.

- The NLP community has made tremendous progress in developing complicated contextual language models that rely on deep learning rather than typical statistical language models. Contextual word embeddings and missing word completion algorithms are two examples of modern NLP methodologies [103], [104]. Such approaches are in accordance with the required recognition task, offering additional contextual information to the visual recognition. In the case of highly degraded documents where a word might be partially or fully missing, such NLP approaches could be adopted. Words that are more contextually consistent with the rest of the sentence will be chosen.
- In order to build a real-life HTR application running on mobiles, building lightweight neural networks are needed. A deeper study of model compression is required which is a broad area of research.

List of publications related to the Dissertation

1. Hassen, H., & Al-Maadeed, S. (2017, April). Arabic handwriting recognition using sequential minimal optimization. In *2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR)* (pp. 79-84). IEEE.
2. Al-Nuzaili, Q., Al-Maadeed, S., Hassen, H., & 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.
3. Mohammed, H. H., Subramanian, N., & Al-Madeed, S. (2021). Learning-free handwritten word spotting method for historical handwritten documents. *IET Image Processing*, 15(10), 2332-2341.
4. Hassen, H., Al-Madeed, S., & Bouridane, A. (2021). Subword Recognition in Historical Arabic Documents using C-GRUs. *TEM Journal*, *8*(10), 4.
5. Hassen, H., Al-Madeed, S., & Bouridane, A. (2022). WSNet – Convolutional Neural Network-based Word Spotting for Arabic and English Handwritten Documents. *TEM Journal*
6. Hassen, H., J Malik, Al-Madeed, S Kiranyaz (2022). 2D Self-organized ONN model for Handwritten Text Recognition. Submitted to Applied Soft Computing journal.
7. Mohammed, H. H., Al-Maadeed, S. I'jam: An Approach for Dots and Diacritics Restoration Based on Self-Organized ONNs. Submitted to IEEE Access journal

Other papers using machine learning

1. Al-Thani, H., Hassen, H., Al-Maadced, S., Fetais, N., & Jaoua, A. (2018, August). Unsupervised technique for anomaly detection in Qatar stock market. In *2018*

International Conference on Computer and Applications (ICCA) (pp. 116-9). IEEE.

2. AlMeer, M. H., Hassen, H., & Nawaz, N. (2019, September). ROM-Based Deep Learning Inference for Sleep Stage Classification. In *Proceedings of SAI Intelligent Systems Conference* (pp. 877-889). Springer, Cham.
3. AlMeer, M. H., Hassen, H., & Nawaz, N. (2019). ROM-based inference method built on deep learning for sleep stage classification. *TEM Journal*, *8*(1), 28.
4. Akbari, Y., Hassen, H., Subramanian, N., Kunhoth, J., Al-Maadeed, S., & Alhajyaseen, W. (2020, February). A vision-based zebra crossing detection method for people with visual impairments. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)* (pp. 118-123). IEEE.
5. Chowdhury, M. E., Rahman, T., Khandakar, A., Al-Madeed, S., Zughaier, S. M., Hassen, H., & Islam, M. T. (2021). An early warning tool for predicting mortality risk of COVID-19 patients using machine learning. *Cognitive Computation*, 1-16.
6. Akbari, Y., Hassen, H., Al-Maadeed, S., & Zughaier, S. M. (2021). COVID-19 Lesion Segmentation Using Lung CT Scan Images: Comparative Study Based on Active Contour Models. Applied Sciences, [1]

References

[1] R. Saabni and J. El-Sana, “Keywords image retrieval in historical handwritten Arabic documents,” *Journal of Electronic Imaging*, vol. 22, no. 1, p. 013016, 2013, doi: 10.1117/1.jei.22.1.013016.

[2] M. Ibn Khedher, H. Jmila, and M. A. El-Yacoubi, “Automatic processing of Historical Arabic Documents: A comprehensive Survey,” *Pattern Recognition*, 2020, doi: 10.1016/j.patcog.2019.107144.

[3] M. Almanea, “Automatic Methods and Neural Networks in Arabic Texts Diacritization: A Comprehensive Survey,” *IEEE Access*, 2021.

[4] R. Farrahi Moghaddam *et al.*, “IBN SINA: A Database for Research on Processing and Understanding of Arabic Manuscripts Images,” *Proceedings of the 8th IAPR International Workshop on Document Analysis Systems - DAS ’10*, pp. 11–18, 2010, doi: 10.1145/1815330.1815332.

[5] W. Pantke, M. Dennhardt, D. Fecker, V. Margner, and T. Fingscheidt, “An Historical Handwritten Arabic Dataset for Segmentation-Free Word Spotting - HADARA80P,” in *2014 14th International Conference on Frontiers in Handwriting Recognition*, Sep. 2014, vol. 2014-Decem, pp. 15–20. doi: 10.1109/ICFHR.2014.11.

[6] M. Kassis, A. Abdalhaleem, A. Droby, R. Alaasam, and J. El-Sana, “VML-HD: The historical Arabic documents dataset for recognition systems,” in *Arabic Script Analysis and Recognition (ASAR), 2017 1st International Workshop on*, 2017, pp. 11–14.

[7] U.-V. Marti and H. Bunke, “The IAM-database: an English sentence database for offline handwriting recognition,” *International Journal on*

Document Analysis and Recognition, vol. 5, no. 1, pp. 39–46, Nov. 2002, doi: 10.1007/s100320200071.

[8] H. Hassen, S. Al-Madeed, and A. Bouridane, "Subword Recognition in Historical Arabic Documents using C-GRUs," *TEM Journal*, vol. 10, no. 4, pp. 1630–1637, 2021, doi: 10.18421/TEM104-19.

[9] S. Kiranyaz, J. Malik, H. ben Abdallah, T. Ince, A. Iosifidis, and M. Gabbouj, "Self-organized Operational Neural Networks with Generative Neurons," *Neural Networks*, vol. 140, pp. 294–308, Aug. 2021, doi: 10.1016/j.neunet.2021.02.028.

[10] A. Herout, M. Dubská, and J. Havel, "Review of Hough transform for line detection," in *Real-time detection of lines and grids*, Springer, 2013, pp. 3–16.

[11] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, "Word spotting and recognition with embedded attributes," *IEEE Trans Pattern Anal Mach Intell*, vol. 36, no. 12, pp. 2552–2566, 2014.

[12] G. Retsinas, G. Louloudis, N. Stamatopoulos, and B. Gatos, "Keyword spotting in handwritten documents using projections of oriented gradients," in *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, 2016, pp. 411–416.

[13] S. Sudholt and G. A. Fink, "Phocnet: A deep convolutional neural network for word spotting in handwritten documents," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2016, pp. 277–282.

[14] V. Frinken, A. Fischer, and H. Bunke, "A novel word spotting algorithm using bidirectional long short-term memory neural networks," in *IAPR*

Workshop on Artificial Neural Networks in Pattern Recognition, 2010, pp. 185–196.

[15] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character HMMs," in *Pattern Recognition Letters*, 2012, vol. 33, no. 7, pp. 934–942. doi: 10.1016/j.patrec.2011.09.009.

[16] N. D. Cilia, C. de Stefano, F. Fontanella, and A. S. di Freca, "A ranking-based feature selection approach for handwritten character recognition," *Pattern Recognition Letters*, vol. 121, pp. 77–86, 2019.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] T. Clanuwat, A. Lamb, and A. Kitamoto, "Kuronet: Pre-modern Japanese kuzushiji character recognition with deep learning," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, pp. 607–614.

[19] M. Jaderberg, K. Simonyan, A. Zisserman, and others, "Spatial transformer networks," *Adv Neural Inf Process Syst*, vol. 28, pp. 2017–2025, 2015.

[20] A. K. Bhunia, A. Das, A. K. Bhunia, P. S. R. Kishore, and P. P. Roy, "Handwriting recognition in low-resource scripts using adversarial learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4767–4776.

[21] F. P. Such, D. Peri, F. Brockler, H. Paul, and R. Ptucha, "Fully convolutional networks for handwriting recognition," in *2018 16th*

International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 86–91.

[22] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 11, pp. 2298–2304, 2016.

[23] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout improves recurrent neural networks for handwriting recognition," in *2014 14th international conference on frontiers in handwriting recognition*, 2014, pp. 285–290.

[24] P. Voigtlaender, P. Doetsch, and H. Ney, "Handwriting recognition with large multidimensional long short-term memory recurrent neural networks," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2016, pp. 228–233.

[25] T. Bluche and R. Messina, "Gated convolutional recurrent neural networks for multilingual handwriting recognition," in *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, 2017, vol. 1, pp. 646–651.

[26] A. Chowdhury and L. Vig, "An efficient end-to-end neural model for handwritten text recognition," 2019. doi: 10.48550/arXiv.1807.07965.

[27] J. Puigcerver, "Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Nov. 2017, vol. 1, pp. 67–72. doi: 10.1109/ICDAR.2017.20.

[28] B. Moysset, C. Kermorvant, and C. Wolf, "Full-Page Text Recognition:

Learning Where to Start and When to Stop," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2017, vol. 1, pp. 871–876. doi: 10.1109/ICDAR.2017.147.

[29] T. Bluche, J. Louradour, and R. Messina, "Scan, Attend and Read: End-To-End Handwritten Paragraph Recognition with MDLSTM Attention," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2017, vol. 1, pp. 1050–1055. doi: 10.1109/ICDAR.2017.174.

[30] T. Bluche, "Joint line segmentation and transcription for end-to-end handwritten paragraph recognition," in *Advances in Neural Information Processing Systems*, 2016, vol. 29, pp. 838–846.

[31] M. Alberti, L. Vogtlin, V. Pondenkandath, M. Seuret, R. Ingold, and M. Liwicki, "Labeling, cutting, grouping: An efficient text line segmentation method for medieval manuscripts," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2019, pp. 1200–1206. doi: 10.1109/ICDAR.2019.00194.

[32] S. Espana-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, "Improving offline handwritten text recognition with hybrid HMM/ANN models," *IEEE Trans Pattern Anal Mach Intell*, vol. 33, no. 4, pp. 767–779, 2010.

[33] T. Plötz and G. A. Fink, "Markov models for offline handwriting recognition: a survey," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 12, no. 4, p. 269, 2009.

[34] H. Bunke, S. Bengio, and A. Vinciarelli, "Offline recognition of unconstrained handwritten texts using HMMs and statistical language

models," *IEEE transactions on Pattern analysis and Machine intelligence*, vol. 26, no. 6, pp. 709–720, 2004.

[35] A. L. Koerich, R. Sabourin, and C. Y. Suen, "Large vocabulary off-line handwriting recognition: A survey," *Pattern Analysis & Applications*, vol. 6, no. 2, pp. 97–121, 2003.

[36] V. Basavaraja, P. Shivakumara, D. S. Guru, U. Pal, T. Lu, and M. Blumenstein, "Age estimation using disconnectedness features in handwriting," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, pp. 1131–1136.

[37] J. Michael, R. Labahn, T. Gruning, and J. Zollner, "Evaluating Sequence-to-Sequence Models for Handwritten Text Recognition," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, Sep. 2019, pp. 1286–1293. doi: 10.1109/ICDAR.2019.00208.

[38] Z. Chen, Y. Wu, F. Yin, and C.-L. Liu, "Simultaneous Script Identification and Handwriting Recognition via Multi-Task Learning of Recurrent Neural Networks," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Nov. 2017, vol. 1, pp. 525–530. doi: 10.1109/ICDAR.2017.92.

[39] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout Improves Recurrent Neural Networks for Handwriting Recognition," in *2014 14th International Conference on Frontiers in Handwriting Recognition*, Sep. 2014, pp. 285–290. doi: 10.1109/ICFHR.2014.55.

[40] D. Castro, B. L. D. Bezerra, and M. Valenca, "Boosting the Deep Multidimensional Long-Short-Term Memory Network for Handwritten Recognition Systems," in *2018 16th International Conference on*

Frontiers in Handwriting Recognition (ICFHR), Aug. 2018, pp. 127–132. doi: 10.1109/ICFHR-2018.2018.00031.

[41] G. Retsinas, G. Sfikas, C. Nikou, and P. Maragos, "Deformation-Invariant Networks For Handwritten Text Recognition," in *2021 IEEE International Conference on Image Processing (ICIP)*, Sep. 2021, pp. 949–953. doi: 10.1109/ICIP42928.2021.9506414.

[42] J. Dai *et al.*, "Deformable Convolutional Networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 764–773. doi: 10.1109/ICCV.2017.89.

[43] R. Alaasam, B. Kurar, M. Kassis, and J. El-Sana, "Experiment study on utilizing convolutional neural networks to recognize historical Arabic handwritten text," in *2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR)*, 2017, pp. 124–128.

[44] R. Alaasam, B. K. Barakat, and J. El-Sana, "Synthesizing versus Augmentation for Arabic Word Recognition with Convolutional Neural Networks," in *2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*, 2018, pp. 114–118.

[45] Y. Chherawala, R. Wisnovsky, and M. Cheriet, "TSV-LR: topological signature vector-based lexicon reduction for fast recognition of pre-modern Arabic subwords," *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, pp. 6–13, 2011, doi: 10.1145/2037342.2037345.

[46] Y. Chherawala and M. Cheriet, "W-TSV: Weighted topological signature vector for lexicon reduction in handwritten Arabic documents," *Pattern Recognition*, vol. 45, no. 9, pp. 3277–3287, 2012, doi:

10.1016/j.patcog.2012.02.030.

[47] Y. Chherawala and M. Cheriet, “Arabic word descriptor for handwritten word indexing and lexicon reduction,” *Pattern Recognition*, vol. 47, no. 10, pp. 3477–3486, 2014, doi: 10.1016/j.patcog.2014.04.025.

[48] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, H. Amiri, and others, “IFN/ENIT-database of handwritten Arabic words,” in *Proc. of CIFED*, 2002, vol. 2, pp. 127–136.

[49] R. F. Moghaddam, M. Cheriet, T. Milo, and R. Wisnovsky, “A prototype system for handwritten sub-word recognition: Toward Arabic-manuscript transliteration,” *2012 11th International Conference on Information Science, Signal Processing and their Applications, ISSPA 2012*, pp. 1198–1204, 2012, doi: 10.1109/ISSPA.2012.6310473.

[50] K. Fouladi, B. N. Araabi, and E. Kabir, “A fast and accurate contour-based method for writer-dependent offline handwritten Farsi/Arabic subwords recognition,” *International Journal on Document Analysis and Recognition*, vol. 17, no. 2, pp. 181–203, 2014, doi: 10.1007/s10032-013-0210-7.

[51] C. Cortes and V. Vapnik, “Support-vector networks,” *Mach Learn*, vol. 20, no. 3, pp. 273–297, 1995.

[52] A. Mesaros *et al.*, “Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 2, pp. 379–393, 2017.

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Adv Neural Inf Process Syst*,

vol. 25, pp. 1097–1105, 2012.

[54] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[55] C. Szegedy *et al.*, “Going deeper with convolutions,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June, pp. 1–9. doi: 10.1109/CVPR.2015.7298594.

[56] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[57] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[58] S. Kiranyaz, T. Ince, A. Iosifidis, and M. Gabbouj, “Generalized model of biological neural networks: Progressive operational perceptrons,” in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 2477–2485. doi: 10.1109/IJCNN.2017.7966157.

[59] S. Kiranyaz, T. Ince, A. Iosifidis, and M. Gabbouj, “Progressive operational perceptrons,” *Neurocomputing*, vol. 224, pp. 142–154, 2017.

[60] D. T. Tran, S. Kiranyaz, M. Gabbouj, and A. Iosifidis, “Heterogeneous Multilayer Generalized Operational Perceptron,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 3, pp. 710–724, Mar. 2020, doi: 10.1109/TNNLS.2019.2914082.

[61] S. Kiranyaz, J. Malik, H. ben Abdallah, T. Ince, A. Iosifidis, and M. Gabbouj, “Self-organized Operational Neural Networks with Generative Neurons,” *Neural Networks*, vol. 140, pp. 294–308, Aug. 2021, doi: 10.1016/j.neunet.2021.02.028.

[62] J. Malik, S. Kiranyaz, and M. Gabbouj, “Operational vs Convolutional Neural Networks for Image Denoising,” *arXiv preprint arXiv:2009.00612*, 2020.

[63] S. Kiranyaz, J. Malik, H. ben Abdallah, T. Ince, A. Iosifidis, and M. Gabbouj, “Exploiting heterogeneity in operational neural networks by synaptic plasticity,” *Neural Computing and Applications*, pp. 1–19, 2021.

[64] D. Thanh Tran, S. Kiranyaz, M. Gabbouj, and A. Iosifidis, “Progressive Operational Perceptron with Memory,” *arXiv e-prints*, p. arXiv--1808, 2018.

[65] D. T. Tran, S. Kiranyaz, M. Gabbouj, and A. Iosifidis, “Knowledge transfer for face verification using heterogeneous generalized operational perceptrons,” in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 1168–1172.

[66] M. Soltanian, J. Malik, J. Raitoharju, A. Iosifidis, S. Kiranyaz, and M. Gabbouj, “Speech Command Recognition in Computationally Constrained Environments with a Quadratic Self-Organized Operational Layer,” in *2021 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2021, pp. 1–6. doi: 10.1109/IJCNN52387.2021.9534232.

[67] H. Davoudi, M. Cheriet, and E. Kabir, “Lexicon reduction of handwritten Arabic subwords based on the prominent shape regions,” *International Journal on Document Analysis and Recognition*, vol. 19, no. 2, pp. 1–15,

2016, doi: 10.1007/s10032-016-0262-6.

[68] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, “Deep Big Simple Neural Nets Excel on Handwritten Digit Recognition,” *Neural computation, 22(12)*, pp. 3207–3220, 2010, doi: 10.1162/NECO_a_00052.

[69] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Advances In Neural Information Processing Systems*, pp. 1–9, 2012, doi: http://dx.doi.org/10.1016/j.protcy.2014.09.007.

[70] S. Afzal *et al.*, “A data augmentation-based framework to handle class imbalance problem for Alzheimer’s stage detection,” *IEEE Access*, vol. 7, pp. 115528–115539, 2019.

[71] E. Chammas, C. Mokbel, and L. Likforman-Sulem, “Handwriting recognition of historical documents with few labeled data,” in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, 2018, pp. 43–48.

[72] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[73] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification,” in *Proceedings of the 23rd international conference on Machine learning - ICML ’06*, 2006, pp. 369–376. doi: 10.1145/1143844.1143891.

[74] S. Lu *et al.*, “Codexglue: A machine learning benchmark dataset for code

understanding and generation," *arXiv preprint arXiv:2102.04664,* 2021.

[75] G. Qin and J. Eisner, "Learning how to ask: Querying LMs with mixtures of soft prompts," *arXiv preprint arXiv:2104.06599,* 2021.

[76] D. H. Kim and T. MacKinnon, "Artificial intelligence in fracture detection: transfer learning from deep convolutional neural networks," *Clin Radiol,* vol. 73, no. 5, pp. 439–445, 2018.

[77] F. Stahlberg and S. Vogel, "QATIP - An Optical Character Recognition System for Arabic Heritage Collections in Libraries," 2016. doi: 10.1109/DAS.2016.81.

[78] P. Krishnan, K. Dutta, and C. V. Jawahar, "Word Spotting and Recognition Using Deep Embedding," in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, Apr. 2018, pp. 1–6. doi: 10.1109/DAS.2018.70.

[79] A. Chowdhury and L. Vig, "An efficient end-to-end neural model for handwritten text recognition," *arXiv preprint arXiv:1807.07965,* 2018.

[80] K. Markou *et al.*, "A Convolutional Recurrent Neural Network for the Handwritten Text Recognition of Historical Greek Manuscripts," in *International Conference on Pattern Recognition,* 2021, pp. 249–262. doi: 10.1007/978-3-030-68787-8_18.

[81] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar, "Improving CNN-RNN Hybrid Networks for Handwriting Recognition," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Aug. 2018, pp. 80–85. doi: 10.1109/ICFHR-2018.2018.00023.

[82] V. Tassopoulou, G. Retsinas, and P. Maragos, "Enhancing Handwritten Text Recognition with N-gram sequence decomposition and Multitask

Learning," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 10555–10560. doi: 10.48550/arXiv.2012.14459.

[83] M. Yousef, K. F. Hussain, and U. S. Mohammed, "Accurate, data-efficient, unconstrained text recognition with convolutional neural networks," *Pattern Recognition*, vol. 108, p. 107482, Dec. 2020, doi: 10.1016/j.patcog.2020.107482.

[84] I. Cojocaru, S. Cascianelli, L. Baraldi, M. Corsini, and R. Cucchiara, "Watch Your Strokes: Improving Handwritten Text Recognition with Deformable Convolutions," in *2020 25th International Conference on Pattern Recognition (ICPR)*, Jan. 2021, pp. 6096–6103. doi: 10.1109/ICPR48806.2021.9412392.

[85] Self-ONNs, "http://selfonn.net/," 2021.

[86] A. Fadel, I. Tuffaha, M. Al-Ayyoub, and others, "Arabic text diacritization using deep neural networks," in *2019 2nd international conference on computer applications & information security (ICCAIS)*, 2019, pp. 1–7.

[87] Z. Barqawi, "Shakkala, Arabic text vocalization." 2017. [Online]. Available: https://github.com/Barqawiz/Shakkala

[88] G. A. Abandah, A. Graves, B. Al-Shagoor, A. Arabiyat, F. Jamour, and M. Al-Taee, "Automatic diacritization of Arabic text using recurrent neural networks," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 18, no. 2, pp. 183–197, 2015.

[89] G. Abandah and A. Abdel-Karim, "Accurate and fast recurrent neural network solution for the automatic diacritization of Arabic text," *Jordan. J. Comp. Inform. Technol*, vol. 6, pp. 103–121, 2020.

[90] G. Tang, M. Müller, A. Rios, and R. Sennrich, “Why self-attention? a targeted evaluation of neural machine translation architectures,” *arXiv preprint arXiv:1808.08946*, 2018.

[91] W. Yin, K. Kann, M. Yu, and H. Schütze, “Comparative study of CNN and RNN for natural language processing,” *arXiv preprint arXiv:1702.01923*, 2017.

[92] J. Lee, K. Cho, and T. Hofmann, “Fully character-level neural machine translation without explicit segmentation,” *Trans Assoc Comput Linguist*, vol. 5, pp. 365–378, 2017.

[93] Y. Wang *et al.*, “Tacotron: Towards end-to-end speech synthesis,” *arXiv preprint arXiv:1703.10135*, 2017.

[94] M. A. H. Madhfar and A. M. Qamar, “Effective Deep Learning Models for Automatic Diacritization of Arabic Text,” *IEEE Access*, vol. 9, pp. 273–288, 2020.

[95] Y. Wang *et al.*, “Tacotron: Towards end-to-end speech synthesis,” *arXiv preprint arXiv:1703.10135*, 2017.

[96] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Highway networks,” *arXiv preprint arXiv:1505.00387*, 2015.

[97] E. Grosicki, M. Carre, J.-M. Brodin, and E. Geoffrois, “Results of the rimes evaluation campaign for handwritten mail processing,” in *Document Analysis and Recognition, 2009. ICDAR’09. 10th International Conference on*, 2009, pp. 941–945.

[98] T. Faisal and S. AlMaadeed, “Enabling indexing and retrieval of historical Arabic manuscripts through template matching based word spotting,” in *2017 1st International Workshop on Arabic Script Analysis and*

Recognition (ASAR), 2017, pp. 57–63.

[99] V. I. Levenshtein and others, “Binary codes capable of correcting deletions, insertions, and reversals,” in *Soviet physics doklady*, 1966, vol. 10, no. 8, pp. 707–710. doi: 1966SPhD...10..707L.

[100] K. Adam, A. Baig, S. Al-Maadeed, A. Bouridane, and S. El-Menshawy, “KERTAS: dataset for automatic dating of ancient Arabic manuscripts,” *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 21, no. 4, pp. 283–290, 2018.

[101] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Adv Neural Inf Process Syst*, vol. 28, 2015.

[102] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[103] A. Radford, T. Narasimhan, T. Salimans, and I. Sutskever, “[GPT-1] Improving Language Understanding by Generative Pre-Training,” in *Preprint*, 2018, pp. 1–12. [Online]. Available: https://gluebenchmark.com/leaderboard%0Ahttps://gluebenchmark.com/leaderboard%0Ahttps://gluebenchmark.com/leaderboard%0Ahttps://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf

[104] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2019, vol. 1, pp. 4171–

4186.