DOCTOR OF PHILOSOPHY

A Novel Approach to Printed Arabic Optical Character Recognition

Alghamdi, Mansoor

Award date:
2019

Awarding institution:
Bangor University

Link to publication

PRIFYSGOL
BANGOR
UNIVERSITY

School of Computer Science

A Novel Approach to Printed Arabic Optical Character Recognition

Mansoor Ali Alghamdi

Submitted in partial satisfaction of the requirements for the degree of Doctor of Philosophy in Computer Science

Supervisor: Dr William J. Teahan

Abstract

Optical character recognition (OCR) is essential in various real-world applications, such as digitizing learning resources to assist visually impaired people and transforming printed resources into electronic media. Considering the Arabic language, the need to extend digital Arabic content on the Internet has motivated more research on Arabic text recognition. However, Arabic OCR still poses significant challenges, owing to the special characteristics of Arabic script. This research aims to develop an effective printed Arabic OCR system.

Performance evaluation of OCR systems is an essential task for OCR systems development. However, studies in Arabic OCR suffer from the lack of proper performance evaluation metrics, the availability of evaluation tools and effective performance evaluation of current OCR systems. Thus, this work proposes a standard protocol with an automated evaluation tool, which has a new set of metrics, for measuring the effectiveness of Arabic OCR systems. In addition, the effectiveness of the state-of-the-art printed Arabic text recognition systems have been experimentally evaluated.

In this work, we describe the implementation of a printed Arabic OCR system. The implantation of this system is divided into five stages: pre-processing, feature extraction, character segmentation, classification and post-processing. Unlike other typical Arabic OCR systems, the developed system performs the feature extraction stage prior to the character segmentation stage.

In the pre-processing stage, a novel thinning algorithm is developed in order to produce skeletons for Arabic text images. An evaluation experiment is conducted to evaluate the performance of the new algorithm against other well established thinning algorithms with respect to several new performance metrics. In all performance tests, the new algorithm produces the best results. In the second stage, a new chain-code representation technique using an agent-based model for extracting features from non-dotted Arabic text images has been proposed. The feature extraction method achieved an accuracy of 98.1%. Based on the extracted features, a character segmentation technique for segmenting connected Arabic words into characters was developed. The character segmentation technique produced a recall of 84.2% and a precision of 77.3%. In the classification stage, the Prediction by Partial Matching (PPM) compression based method is applied as a classifier to recognise Arabic text. Experimental evaluation on a public dataset reveals that the system has an accuracy of 77.3% for paragraph-based text images. In the final stage, a post-processing technique based on a PPM model is applied for correcting the OCR output. By applying the post-processing method, the recognition accuracy improves to 86.9%. The experimental results show that the system substantially improves upon the state-of-the-art when compared with four well-known Arabic OCR systems using the automated evaluation tool.

Acknowledgments

First, I would like to express my deep gratitude to my supervisor Dr. William J. Teahan who was abundantly helpful and offered invaluable assistance, support and guidance. I could not have imagined having a better supervisor for my PhD study.

Many thanks to my beloved wife Maryam for her understanding and endless love, throughout the duration of my studies.

Special thanks to my son Meshari, who is the pride and joy of my life, for putting up with me when I was busy working on my thesis instead of playing with him.

Last but not the least, I would like to thank my parents, brothers and sisters who deserve special mention for their support. Especially my mother is the one who truthfully raised me with her caring and gently love and her every day prayers. I would not be in this position without her.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

BACC	Bangor Arabic Compression Corpus
GED	Graph Edit Distance
HHM	Hidden Markov Models
NLP	Natural Language Processing
NNs	Neural Networks
OCR	Optical Character Recognition
PATRION	Printed Arabic Optical Character Recognition
PPM	Prediction by Partial Matching
SVM	Support Vector Machines

Chapter 1: Introduction

1.1 Background and Motivation

Optical character recognition (OCR) is a technique that transforms a printed or handwritten text image into an electronic textual format. OCR development is considered as a challenging task in the field of pattern recognition. Many OCR approaches have been proposed for Latin and non-Latin scripts. Previous research on text recognition has focused primarily on Latin scripts, such as English, and it has not been until the last two decades that recognition of non-Latin scripts, such as Arabic, has been researched (Alginahi, 2013). Furthermore, owing to the special characteristics of Arabic script, OCR for Arabic text still poses significant challenges (Slimane et al., 2013).

OCR applications can be categorized into two types, namely online text recognition and offline text recognition (Ahmad, 2016). Figure 1.1 illustrates the different types of text recognition systems. Offline text recognition systems perform on text that has been previously written or printed on a page. Online text recognition systems, by contrast, recognise text captured in real-time. Offline text recognition is further classified into two types: handwritten text recognition and printed text recognition. In contrast, online text recognitions systems are limited to recognising handwritten text. Although handwritten script is significantly more challenging than printed Arabic text for OCR, Arabic printed text OCR still poses significant challenges (Slimane et al., 2013). Therefore, this study will deal only with the printed text recognition type.

Generally, the process for developing a typical OCR system involves five stages: pre-processing, segmentation, feature extraction, classification and post-processing, (Khorsheed, 2002). A text image may need to go through the pre-processing stage for enhancing the readability of the image. During the segmentation stage, the text image is segmented into small patterns, like segmenting words into characters. Then, features are extracted from the segmented patterns to be utilized by the next stage which is classification. The classification stage is responsible for assigning a pattern into a pre-classified class based on the extracted features of the pattern. The final stage is the post-processing stage which improves the recognition accuracy by detecting and correcting linguistic misspellings in the produced OCR text without human intervention.

Arabic OCR is highly desirable in various real-world applications, such as digitising learning resources to assist visually impaired people, bank cheque processing and mail sorting (Al-Badr and Mahmoud, 1995; Alginahi, 2013). Furthermore, there are many initiatives for Arabic digital content enrichment (Saad and Ashour, 2010). One of these initiatives is King Abdullah's Initiative for Arabic Content (Jambi, 2014). Recently, Crown Prince Mohammed Bin Salman has established the global center for combating extremist ideology. One of its objectives is to analyse online Arabic text, such as text from Twitter that may have extremist content (Etidal, 2018). Twitter has a character limit of 280 characters, so users are not able to add long tweets. However, users publish their tweets on text images in order to avoid cramming their text into 280 characters. Therefore, robust and efficient Arabic OCR is required to support the initiative by increasing Arabic content on the Internet and to analyse online text images.

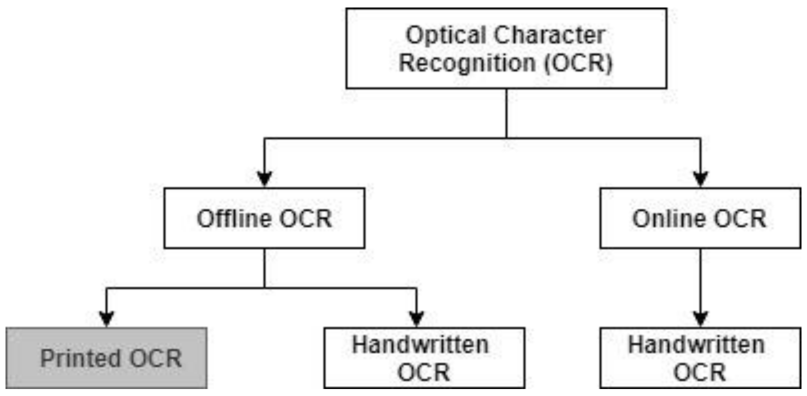


Figure 1.1: Categories of OCR systems
Note: The shaded text box is the scope of this research.

1.2 Research Questions

The following are the specific research questions being investigated in this study in order to help achieve the primary aim of the project:

- Is the current OCR methodology which involves the five sequential stages (pre-processing; feature extraction; segmentation; classification and post-processing) the most effective for designing Arabic OCR?
- Are there alternative methodologies that might yield better results for Arabic OCR?

- Are the general standard performance measurements of character accuracy sufficient to assess how Arabic OCR systems are coping with the challenges of Arabic script?
- Are the available OCR systems sufficient to recognise printed Arabic text images?
- Can Arabic text be effectively recognised by considering Arabic text as non-dotted text and then corrected later on in order to recover the dotted from?
- Can a compression based method be effectively applied as a classifier for the recognition of the text features?
- Can a compression based method be applied to Arabic OCR output that contributes to significant improvement in Arabic text recognition accuracy?

1.3 Aims and Objectives

The primary aim of this study is to develop a novel OCR system for printed Arabic script that substantially improves upon the state-of-the-art when compared with presently available systems. A further aim is to develop new approaches that make use of compression-based language modelling as these have proven to be very effective for many natural language processing (NLP) applications in the past but they have not yet been applied to the specific problem of Arabic OCR.

In order to fulfill these aims, the following objectives for this work are to:

- perform a comprehensive review of printed Arabic text recognition (see Chapter 2);
- develop an automated tool for evaluating the performance of Arabic OCR systems (see Chapter 3);
- evaluate the effectiveness of the state-of-the-art printed Arabic text recognition systems (see Chapter 3);
- design and implement a new thinning algorithm for Arabic text and evaluate the effectiveness of the new algorithm by comparing it with well-known thinning algorithms (see Chapter 4);

- develop a new chain-code representation technique using an agent-based model for extracting features from non-dotted Arabic text images prior to the character segmentation phase (see Chapter 5);
- develop a new character segmentation technique for segmenting connected Arabic words into characters based on the extracted features (see Chapter 6);
- develop a new compression-based method for classifying the segmented Arabic characters (see Chapter 7);
- use compression-based post-processing techniques (e.g., adding dots to Arabic text and correcting the OCR output) (see Chapter 8);

1.4 Contributions

The main contribution of this study is the development of a novel OCR system for printed Arabic script that substantially improves upon the state-of-the-art. In the following are the further contributions of this work to the field of printed Arabic text recognition:

1. A novel printed Arabic OCR application to recognise multi-font types, multi-size and multi-styles Arabic text images has been developed. The effectiveness of this system was evaluated by comparing the output produced with ground truth and by comparing it with other Arabic OCR systems.
2. A comprehensive review of printed Arabic text recognition has been performed. This review analyzed the challenges and open problems in the field of printed Arabic OCR. This work resulted in the following publication:
    - Alghamdi, M. and Teahan, W., 2018. Printed Arabic Script Recognition: A Survey. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 9(9).
3. An automated evaluation tool with new objective metrics for the evaluation of Arabic OCR performance has been developed. Also, a standard protocol for measuring the effectiveness of Arabic OCR systems with a comprehensive automated evaluation of the current Arabic OCR systems has been provided. This work resulted in the following publications:

- Alghamdi, M.A., Alkhazi, I.S. and Teahan, W.J., 2016. Arabic OCR evaluation tool. In *Computer Science and Information Technology (CSIT), 2016 7th International Conference on* (pp. 1-6). IEEE.
- Alghamdi, M. and Teahan, W., 2017. Experimental evaluation of Arabic OCR systems. *PSU Research Review*, *1*(3), pp.229-241.

4. A novel thinning algorithm for Arabic text has been developed and evaluated by comparing it with alternative thinning algorithms. Also, a new set of performance metrics for the evaluation of thinning algorithm is proposed. This work resulted in the following publication:

   - Alghamdi, M.A. and Teahan, W.J., 2017. A new thinning algorithm for Arabic script. *International Journal of Computer Science and Information Security*, *15*(1), p.204-211.

5. A new technique based on a cognitive insight used for extracting features from non-dotted Arabic text images has been developed and evaluated. The method has been tested on a sample of 100 Arabic words images and has produced an average edit distance accuracy of 98.07%.
6. A new character segmentation technique based on the features extracted from Arabic text has been developed and evaluated. The method has been tested on a sample of 100 Arabic words images and has produced a recall of 84.21% and a precision of 77.25%.
7. A novel compression-based method has been applied as a classifier for the first time to Arabic textual feature recognition and has been evaluated. The method has produced an average edit distance accuracy of 77.31%.
8. A post-processing technique based on the PPM compression scheme has been applied to two problems: adding dots to Arabic text and correcting the OCR output. The post-processing technique improves the edit distance accuracy of the printed Arabic OCR system from 77.3% to 86.9% which is a significant improvement.
9. A new methodology for implementing printed Arabic OCR has been developed. This methodology performs the feature extraction stage prior to the character segmentation stage in order to overcome the challenges of Arabic character segmentation.

1.5 Thesis Outline

The organisation of the thesis is illustrated in Figure 1.2. This thesis is divided into nine chapters as follows.

Chapter 1, entitled “Introduction”, presents the background and motivation of this work and highlights the aim and objectives of this work along with the research questions. Also, it reviews the contribution of this work.

Chapter 2, entitled “Literature Review”, reviews the literature related to printed Arabic text recognition research. It discusses techniques that have been utilized for developing printed Arabic OCR with emphasis on the issues related to Arabic script. It also addresses the challenges and open problems in printed Arabic text recognition.

Chapter 3, entitled “Experimental Evaluation of Arabic OCR systems”, introduces an automated evaluation tool with new objective metrics for the evaluation of Arabic OCR performance. Also, it describes an experiment to automatically evaluate four well-known Arabic OCR systems using a set of performance metrics. The evaluation experiment is conducted on a publicly available printed Arabic dataset comprising 240 text images with a variety of resolution levels, font types, font styles and font sizes.

Chapter 4, entitled “Thinning”, proposes a new thinning algorithm for Arabic script. The chapter also addresses several objective performance metrics for assessing statistically the effectiveness of thinning algorithms. An experiment is conducted to evaluate the new thinning algorithm against two well established thinning algorithms with respect to the proposed objective performance metrics.

Chapter 5, entitled “Feature Extraction”, proposes a new approach based on a cognitive insight for extracting features from non-dotted Arabic text images. It describes the rationale behind using this approach and explains how the feature of Arabic text images are extracted.

Chapter 6, entitled “Character segmentation”, presents a new character segmentation technique for segmenting connected Arabic words into characters.

Chapter 7, entitled “PPM Classification”, describes the use of the Prediction by Partial Matching (PPM) compression based method for textual feature recognition. The chapter implements the PPM model to recognise isolated Arabic character images and Arabic text images.

Chapter 8, entitled “PPM Post-processing”, introduces a post-processing method based on a PPM model for correcting Arabic OCR output. It also investigates the effectiveness of using the PPM based correction method for correcting the Arabic OCR output.

Chapter 9, entitled “Conclusion and Future Work”, summarises the main conclusions of this study and provides several directions for future research.

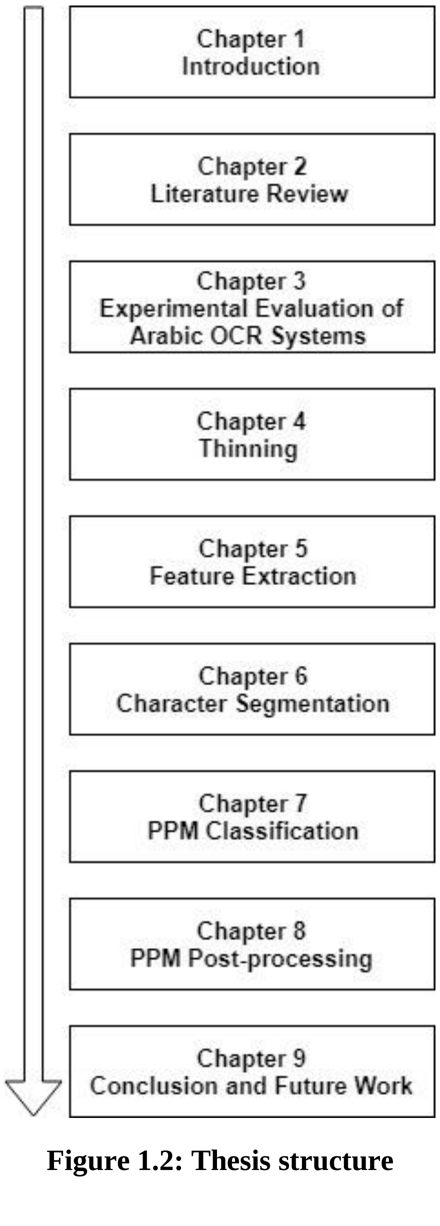


Figure 1.2: Thesis structure

Chapter 2: Literature Review

2.1 Introduction

OCR is a technique that transforms a printed or handwritten text image into an electronic format. OCR development is considered a challenging task in the field of pattern recognition. Many OCR approaches have been proposed for Latin and non-Latin scripts. However, printed Arabic OCR still poses great challenges because of the special characteristics of Arabic script (Slimane et al., 2013; Ahmed et al., 2019)

Arabic OCR is highly desirable in various real-world applications, such as digitising learning resources to assist visually impaired people, bank cheque processing and mail sorting (Alginahi, 2013; Al-Badr and Mahmoud, 1995). Furthermore, there are many initiatives for Arabic digital content enrichment (Saad and Ashour, 2010). One of these initiatives is King Abdullah's Initiative for Arabic Content. Therefore, a robust and efficient Arabic OCR is required to support this initiative by increasing Arabic content on the Internet.

Numerous methods have been proposed for recognising printed Arabic script from an image, yet we are unaware of comprehensive surveys of printed Arabic OCR during the last fifteen years. Two surveys have been conducted on printed Arabic OCR (Al-Badr and Mahmoud, 1995; Khorsheed, 2002). However, these reviews do not reflect the current progress in printed Arabic OCR. Therefore, establishing a guide and baseline for future directions remains important for Arabic OCR researchers.

This work will establish this guide and baseline for Arabic OCR researchers by providing a comprehensive literature review of printed Arabic text recognition research. It reviews techniques that have been utilized for developing printed Arabic OCR with emphasis on the issues related to Arabic script. It also addresses the challenges and open problems in printed Arabic text recognition.

Part of this work in this chapter has been published in the following paper:

- Alghamdi, M. and Teahan, W., 2018. Printed Arabic Script Recognition: A Survey. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 9(9).

This chapter is organised as follows. In Section 2.2, Arabic script characteristics and challenges are discussed. Section 2.3 presents the methodologies of printed Arabic

OCR, with subsections that review the five stages of the development of printed Arabic OCR: preprocessing, segmentation, feature extraction, classification and post-processing. Section 2.4 discusses performance evaluation issues of printed Arabic OCR. Section 2.5 concludes with a discussion about challenges and open problems. A summary is provided in the last section.

2.2 Arabic script characteristics and challenges

There is no doubt that printed Arabic OCR faces a number of challenges and there is still an intensive need for more research (Al-Helali and Mahmoud, 2017). However, most challenges facing the development of Arabic OCR are due to the characteristics of Arabic script. Arabic script has some features that distinguish it from other languages. Compared to English, the most obvious feature of Arabic script is that it is written cursively from right to left in both printed and handwritten. The greatest challenges are due to the more complex characteristics of Arabic script. In the following section, the characteristics of Arabic script that may complicate recognition will be discussed:

- *Shapes and positions*

The Arabic alphabet has 28 basic letters (see Table 2.1). However, an Arabic letter may contain four dissimilar shapes in relation to its location inside a word: whether it is an isolated letter, an initial letter (in which a letter is inked from the right side, an ending letter (in which a letter is linked form the left side) or a middle letter (in which a letter is linked from the right and left sides). Thus, the number of letters to be recognized will increase from 28 letters to 125 letters. Also, Arabic script contains loop shaped letters.

- *Overlapping characters and Ligatures*

Characters in an Arabic word might be overlapped vertically with or without touching each other (see Figure 2.1). In particular, some characters are combined and written as a ligature such as (لا) which is a combination of two letters Lam (ل) and Alf (ا). However, ligatures occur in Arabic script depending on the type of fonts being used (AlSalamah and King, 2018). For instance, in Traditional Arabic font, there are about 220 ligatures whereas Simplified Arabic incorporates about 150 ligatures (El-Mahallawy, 2008).

- *Diacritics*

Characters in an Arabic word can exist with diacritics or short vowels such as Fat-hah, Dhammah, Mada’ah, Kasrah and Sukkun, as illustrated in Figure 2.1. These can be placed either over or below the letters as strokes. In addition, Tanwen is considered as a diacritic which is indicated by double Fat-hah, double Dhammah and double Kasrah. One more diacritic that Arabic script has is Shaddah which is similar to the number 3 as it is rotated 90° clockwise.

Isolated	Initial	Middle	End
ا	ا	ـا	ـا
ب	بـ	ـبـ	ـب
ت	تـ	ـتـ	ـت
ث	ثـ	ـثـ	ـث
ج	جـ	ـجـ	ـج
ح	حـ	ـحـ	ـح
خ	خـ	ـخـ	ـخ
د	د	ـد	ـد
ذ	ذ	ـذ	ـذ
ر	ر	ـر	ـر
ز	ز	ـز	ـز
س	سـ	ـسـ	ـس
ش	شـ	ـشـ	ـش
ص	صـ	ـصـ	ـص

Isolated	Initial	Middle	End
ض	ضـ	ـضـ	ـض
ط	ط	ـطـ	ـط
ظ	ظ	ـظـ	ـظ
ع	عـ	ـعـ	ـع
غ	غـ	ـغـ	ـغ
ف	فـ	ـفـ	ـف
ق	قـ	ـقـ	ـق
ك	كـ	ـكـ	ـك
ل	لـ	ـلـ	ـل
م	مـ	ـمـ	ـم
ن	نـ	ـنـ	ـن
ه	هـ	ـهـ	ـه
و	و	ـو	ـو
ي	يـ	ـيـ	ـي

Table 2.1: Arabic characters with different positions and shapes

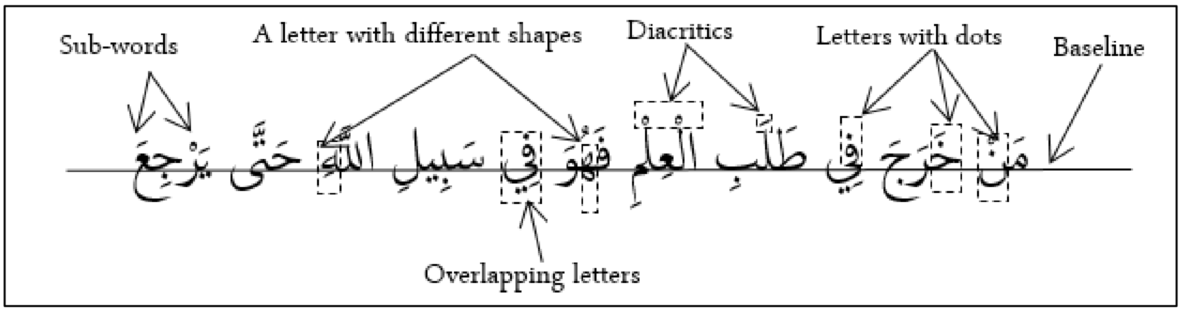


Figure 2.1: Arabic script characteristics

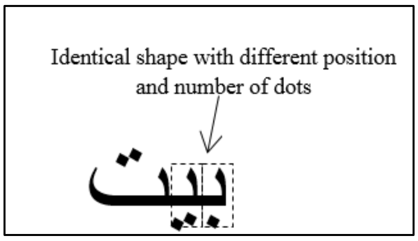


Figure 2.2: Two characters (*Ba* and *Ya*) with an identical shape and a different number of dots

- *Cursive*

As mentioned above, Arabic script is a cursive script which means that a word is composed of connected characters using a horizontal line called the *baseline*. However, six characters (ا، د، ذ، ر، ز، و) of the Arabic alphabet are not linked with succeeding letters. This can present a challenge because these characters can divide a word into one or more units as sub-words (see Figure 2.1).

- *Presence of dots*

In old Arabic script, the characters were not originally dotted. However, due to the difficulty of recognising Arabic characters by non-native Arabic learners, the characters of Arabic script have become dotted (Baik, 1992). The Arabic alphabet relies on number and position of dots in order to differentiate between similar letters (see Figure 2.2). Fifteen characters in the Arabic alphabet have dots. They can be placed below the character, above it or in the middle. Ten of these characters have one dot, three have two dots and two have three dots, as shown in Table 2.1.

2.3 General Arabic OCR methodology (model)

This section will focus on the methodologies used by printed Arabic OCR systems. Published approaches and systems for Arabic OCR indicate that the process of implementing Arabic OCR consists of five phases: (1) pre-processing; (2) feature extraction; (3) segmentation; (4) classification and (5) post-processing, see Figure 2.3.

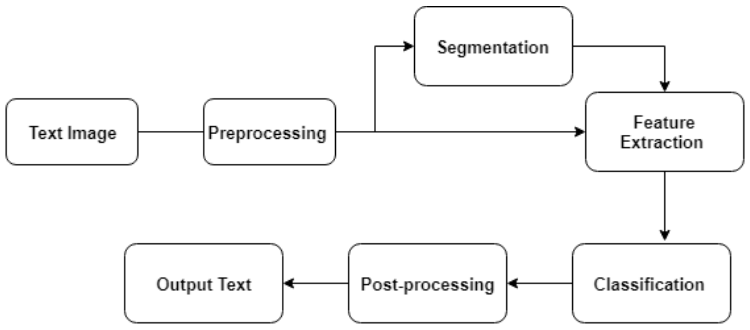


Figure 2.3: General printed Arabic OCR methodology

2.3.1 Preprocessing phase

This is the first phase of OCR methodology which is responsible for enhancing the readability of the input image. Preprocessing is a combination of algorithms that are applied to the input image in order to reduce noise and alterations, thus simplifying the subsequent phases of OCR methodology (Abu-Ain et al., 2018; Al-Badr and Mahmoud, 1995). There are various factors that affect the quality of the input image. Al-Badr and Mahmoud (1995) list the history of image, the printing process, the kind of font, the quality of paper, the condition of the image and the image acquisition as the vital factors that influence the input image quality.

Khorsheed and Clocksin (2000) emphasize that the downstream OCR accuracy relies on the quality of the input image. Furthermore, Al-Badr (1995) states that OCR systems, which report high recognition accuracy on some input images, will report less recognition accuracy on input images that are poor in quality. Thus, the preprocessing phase is a critical stage in OCR development that simplifies the data for the subsequent phases to operate accurately. Generally, several preprocessing operations are employed on the input image: binarization, layout analysis, thinning, smoothing and filtering, size and slant normalization, slant detection, skew detection and baseline detection. The selection of these operations to be applied in the preprocessing relies upon the conditions of the input image, such as the amount of noise and skew in the input image (Kannan and Subramanian, 2015). In the following section, the preprocessing techniques which are applied in Arabic OCR will be clarified.

Binarisation

For character recognition, the binarisation process involves converting an input gray scale image into a binary image, in which a pixel has only two values 0 and 1. The binary image has the critical information, such as the shape of characters. It has been found that increasing processing speed and reducing storage capacity are the key benefits of binarisation technique (El-Mahallawy, 2008; Lawgali, 2015). Al-Badr and Haralick (1998) suggest selecting the most appropriate method for binarisation might separate connected objects or joining isolated objects. A number of studies have confirmed the efficiency of computing the histogram of the gray scale of an image and then detecting a cut-off point as the binarisation method (Al-Badr and Haralick, 1998; Jumari and Ali, 2012). However, some researchers work on recognition without applying binarisation methods, such as AbdelRaouf (2012) and Pavlidis (1993).

Size normalization

Since Arabic characters differ in size, as described earlier, size normalization is commonly applied to characters or words by scaling the characters or the words to an adjusted size. This process is crucial for the recognition or classification phase, since some recognition methods are sensitive to dissimilarity in size and position, such as template matching and correlation approaches (Al-Badr and Haralick, 1998). AbdelRaouf (2012) classified normalization methods into two approaches: moment-based normalization; and nonlinear normalization. It is argued that normally a character is normalized to a standard size for classification (Cheriet et al., 2007). However, in terms of word normalization, applying normalization to a word instead of a character will result in losing critical information (Cheriet et al., 2007; Cho et al., 1995).

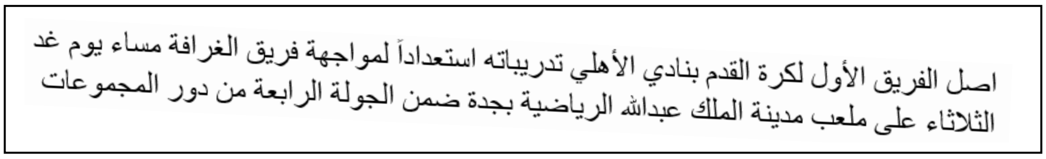
De-noising

Noise may be presented during the acquisition process via scanners which results in distortions and variations in the input text image. Besides this, very small items in the text image can be reflected as noise (Lawgali, 2015), which are byproducts of image scanning or binarisation and which are not parts of the text. Such noise may have a major impact on the performance of OCR systems (Drira and Lebourgeois, 2012).

Noise removal is an operation for enhancing the visual quality of the input image (Shi et al., 2012).

As a solution, several techniques have been introduced that are considered as noise removal methods (Buades et al., 2005). These methods like dilation algorithms, which are applied to broken letters, and erosion algorithms, which are applied to text images with touching letters (AbdelRaouf, 2012), are conditioning processes in terms of OCR development (Al-Badr and Mahmoud, 1995; Gonzalez and Woods, 2002). In addition, the median filter approach is commonly used in both printed text images and handwritten text images. For example, Al-ani et al. (2014) apply this approach for removing noise in printed Arabic text images. Another example of a study that applies a median filter algorithm in handwritten Arabic text images is a study by Al-Shatnawi et al. (2011). Ahmed et al. (2012) applied a morphological noise removal method for Arabic printed OCR proposed by Mahmoud (1994). However, Ahmed et al. (2012) discovered that letter holes could be filled while applying this method, with lower thresholds, to Arabic text images.

Figure 2.4. An example of Arabic text skew



اصل الفريق الأول لكرة القدم بنادي الأهلي تدريباته استعداداً لمواجهة فريق الغرافة مساء يوم غد
الثلاثاء على ملعب مدينة الملك عبدالله الرياضية بجدة ضمن الجولة الرابعة من دور المجموعات

In fact, the review suffers from the fact that some printed Arabic OCR studies applied noise removal algorithms without providing information of the applied algorithm, for instance, in Taha et al. (2012). Such approaches, however, should be selected carefully when considering OCR systems. That is, because of the similarity between Arabic letters, any alteration of a letter might change it to another letter. Thus, a perfect noise removal method is the method that is able to eliminate noise while preserving the shape of the character (Drira and Lebourgeois, 2012).

Skew detection and correction

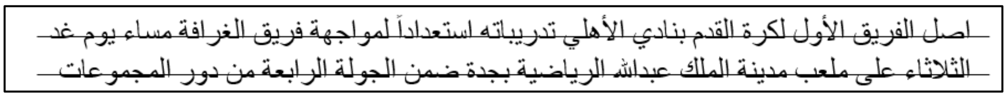
Initially, a text image has zero rotation, yet when physically scanning the image manually, rotation of images up to 20^{o} might occur (Khorsheed, 2002). This rotation is called skew which results in non-zero skew text images (see Figure 2.4). The skew can lead to incorrect recognition and baseline detection (Al-Shatnawi and Omar, 2009). According to Abuhaiba (2003), it is impossible to segment a text if the text is rotated. As a result, detecting and correcting the skew is critical to OCR applications that rely on segmentation approaches to recognize characters.

The process of estimating the skew angle is known as skew detection, whereas the process of rotating the image with the purpose of correcting the skew is called skew correction. A wide variety of skew detection and correction methods have been proposed. Sun and Si (1997) group these methods into five groups: projection profile, Hough transform, Fourier transform, nearest neighbor clustering and correlation. The Hough transform is the standard approach for detecting the skew (Parker, 2011). A method based on the projection profile was introduced by Baird (1995). Hull (1998) has provided a comprehensive review of twenty–five skew detection and correction approaches. The author concludes that further work on more sophisticated methods is still required. The Radon transform method has shown its efficiency for skew correction (Dong et al., 2005). Some methods are designed for specific applications and image type. For example, a new method has emerged for Arabic text images from Al-Shatnawi and Omar (2009). Alginahi (2010) concludes that selecting a skew detection and correction method relies on the image type.

Baseline detection

As described in the previous section, Arabic characters are joined through a horizontal line called the baseline (see Figure 2.5). Graphically, the baseline can be described as the line which has the maximal amount of black pixels (Shafait et al., 2006). This line contains critical information about the text, such as text orientation and position of connection points between Arabic letters (Al-Badr and Mahmoud, 1995). Thus, detecting the baseline is beneficial for many OCR stages, for instance, skew normalization (Omar et al., 2000), segmentation (Amin, 1998; Arica and Yarman-Vural, 2002) and structural features extraction such as the character's dots (El-Hajj et al., 2005).

It has been reported that most Arabic OCR has applied baseline detection methods as a preprocessing step (Al-Shatnawi et al., 2011) .The baseline detection techniques for Arabic script have been classified into four groups by Al-Shatnawi and Omar (2008); namely, horizontal projection methods, the word skeleton method, contour tracing and principle component analysis. Among these, the horizontal projection technique is widely implemented for determining the baseline in Arabic OCR, such as in Pechwitz and Margner (2002) and Al-Badr and Mahmoud (1995). Several studies implement a horizontal projection approach in OCR systems for detecting the baseline, such as in Zidouri et al. (2003), El-Hajj et al. (2005), Ahmed et al. (2012) and Al-rashaideh, (2006). It has been emphasized that the horizontal projection method is simple and efficient for Arabic printed text (Pechwitz and Margner, 2002; Zeki, 2005). However, this method is applicable only for noise-free images, as it fails for unclean images (Bukhari et al., 2011).



اصل الفريق الأول لكرة القدم بنادي الأهلي تدريباته استعداداً لمواجهة فريق الغرافة مساء يوم غد
الثلاثاء على ملعب مدينة الملك عبدالله الرياضية بجدة ضمن الجولة الرابعة من دور المجموعات

Figure 2.5: Baseline detection of a printed Arabic text image

Another baseline detection approach is the x-y cut proposed by Nagy et al. (1992) which is based on a horizontal projection method. This method works well for Arabic noisy images, though it fails in the presence of large amounts of noise and skew (Bukhari et al., 2011). Consequently, Bukhari et al. (2011) proposed using a ridge-based text line detecting approach for Arabic text. The former method's efficiency has been tested and recommended for different types of Arabic text images, since it was found to achieve above 96% text line detection accuracy (Bukhari et al., 2011).

Al-Shatnawi and Omar (2008) summarize the state-of-the-art of baseline detection methods in Arabic script. In summary, for printed Arabic text, the standard horizontal projection method is sufficient for detecting the baseline, since the baseline in printed text is straight whereas for handwriting, the baseline is not straight, thus more sophisticated approaches should be considered (Parvez and Mahmoud, 2013).

Thinning and skeletonization

Producing skeletons is a critical pre-processing operation for OCR in which extracting features from the skeleton of a character is essential (Parker, 2011). The method for producing the skeleton of a pattern image is called thinning. Thinning "skeletonization" can be defined as the process of unpeeling as many pixels as possible without distorting the general shape of the character (Cheriet et al., 2007). In other words, it involves operations that can be implemented in order to produce the skeleton of object images. Thinning techniques are classified into iterative approaches and non-iterative approaches (Abu-Ain et al., 2013). The former can be either sequential methods, which perform by peeling the counter pixels individually, or parallel methods which perform simultaneously on all the counter pixels until obtaining a skeleton (Saeed et al., 2010). The latter utilize other techniques, such as distance transforms, to produce a skeleton without examining all pixels (Saeed et al., 2010).

In general, thinning is usually applied in OCR systems as a method for reducing the amount of data of a pattern that needs to be considered for next processing stage, thereby saving storage space for the structural information of the pattern (Naccache and Shinghal, 1984; Al-Shatnawi et al., 2011). Furthermore, thinning algorithms have contributed to facilitate feature extraction of a pattern which is the core task in OCR to identify the pattern from another, since the relevant information of a pattern is not related to the thickness of the pattern (Devi, 2006; Alginahi, 2010). Therefore, it is claimed that the effectiveness of an OCR performance is heavily reliant on how effective the thinning algorithm is (Ali, 2012). The authors in (Cheriet et al., 2007) and (Chatbri and Kameyama, 2014) agree that the main features of a desirable thinning algorithm are: ensuring the peeling is as thin as possible, connected and preserving the topology of a pattern.

A number of thinning algorithms have been proposed, such as in (Guo and Hall, 1992), (Hilditch, 1983), (Wang and Zhang, 1989); for a comprehensive survey refer to (Lam and Lee, 1992). However, these algorithms have not been proposed specifically for Arabic script; rather they are proposed for general purposes. Additionally, direct adoption of such thinning algorithms, which are developed for other languages, may be not applicable to Arabic because of its characteristics (Al-shatnawi and Omar, 2014; Cowell and Hussain, 2001; Hosseini, 1997).

According to Cowell and Hussain (2001), various problems have arisen when applying thinning algorithms to Arabic characters. One of the main obstacles with applying thinning algorithms to Arabic text is reducing the number of dots for similar characters, where the number of dots can differentiate between them (Al-Badr and Mahmoud, 1995; Al-shatnawi and Omar, 2014). It is believed that any deletion of character dots will result in misrepresenting these characters (Hosseini, 1997). Therefore, some researchers extract dots of Arabic characters before applying thinning algorithms, in order to overcome this problem (Melhi et al., 2001; Al-Badr and Haralick, 1998).

Another problem of thinning algorithms when considering Arabic script concerns preserving the connectedness of Arabic text. Some thinning approaches may fail in preserving the Arabic text connectivity which will lead to challenges in text recognition (Al-shatnawi and Omar, 2014). Owing to these reasons, it is claimed that thinning is responsible for many recognition errors in OCR systems (Al-Badr and Mahmoud, 1995; Al-shatnawi and Omar, 2014). Therefore, thinning algorithms must be capable of both preserving dots and the connectedness of Arabic script.

As stated, there has been relatively few publications on developing thinning algorithms for Arabic (Ali, 2012; A. M. Al-Shatnawi et al., 2011; Al-ani et al., 2014). For instance, (Cowell and Hussain, 2001; Altuwaijri and Bayoumi, 1995) introduce thinning algorithms for Arabic script. However, the proposed algorithms can only deal with isolated Arabic characters. Furthermore, one study by Ali (Ali, 2012) provides a thinning algorithm for Arabic handwritten script. Unfortunately, none of previous studies have considered the challenge of Arabic script discussed above, such as dots and connectivity preservation, when developing the thinning algorithm.

2.3.2 Segmentation phase

After the preprocessing phase, an enhanced text image in the sense of low noise and variation, and a necessary amount of character information (Al-Badr and Mahmoud, 1995), has been produced. During the segmentation phase, the text image is segmented into small components, with a page being segmented into lines, a line into words and a word into letters (Lorigo and Govindaraju, 2006; Naz et al., 2014; Plamondon and Srihari, 2000). Segmentation is a crucial step in Arabic OCR system development because of the fact that it plays a vital role in ensuring the success of the subsequent feature extraction and classification stages (Alginahi, 2013; Zeki, 2005; Stolyarenko and Dershowitz, 2011). However, Zeki (2005) stresses that misrecognition can arise by applying a poor segmentation method. As a result, this stage will have a critical impact on the recognition rate of the text (El-Mahallawy, 2008).

As explained previously, one of the main challenges facing Arabic OCR development is the cursiveness of Arabic script. Segmentation of Arabic text thus can be more difficult and time consuming for the development of Arabic OCR systems (Alginahi, 2013). Correspondingly, segmentation has been considered as the main contribution for increasing the recognition error rate in Arabic OCR systems (Amara et al., 2016; Nawaz et al., 2003; Zeki, 2005).

Generally, segmenting a text image can be graded into two types: external segmentation; and internal segmentation (Arica and Yarman-Vural, 2001). While the former type deals with the isolation of different writing objects such as, paragraphs, sentences and words, the latter deals with the isolation of characters (Alkhateeb, 2010; Arica and Yarman-Vural, 2001).

2.3.2.1 External segmentation

External segmentation refers to the document layout analysis, in particular page decomposition. Document layout analysis is accomplished in order to identify the physical structure of a page (O'Gorman, 1993). As far as offline OCR development is concerned, page analysis is a basic step which segments the image into its different logical parts with the identical type of information, such as graphs, text and tables. Page layout analysis is performed in two approaches: structural analysis by which a page is decomposed into blocks of the page elements, such as paragraphs and words; and functional analysis by which a page is decomposed into functional elements such as title and abstract (Alkhateeb, 2010; Arica and Yarman-Vural, 2002; O'Gorman, 1993).

With respect to Arabic document processing, page decomposition refers to the isolation of text lines of a texture region and the segmentation of words and sub-words (El-Mahallawy, 2008; Khorsheed, 2002; Srihari and Ball, 2012), since it is restricted to text images (Khorsheed, 2002). According to Khorsheed (2002), Amin and Masini (1986) and Mari (1989), applying a fixed threshold to Arabic text documents to determine text lines is the standard method. However, this method fails with a skewed text image (Amin, 1998).

Methods based on histogram projection are considered as conventional approaches for isolating lines and words in Arabic text documents (Amin and Masini, 1986; Ymin and Aoki, 1996). Several studies have relied on horizontal projection techniques for segmenting Arabic text images into lines, such as Cheung et al. (2001), Khorsheed (2007), Amin and Masini (1986), Al-Yousefi and Udpa (1992), Abuhaiba et al. (1994), Taha et al. (2012) and Supriana and Nasution (2013). It is recommended horizontal projection be applied for text images because of its advantages in reducing computational load and its simplicity of implementation (Cheung et al., 2001). Moreover, horizontal projection is an appropriate method for locating text lines in Arabic printed text, since the text lines in printed text are straight (Parvez and Mahmoud, 2013).

For line segmentation, researchers in Arabic OCR determine words in a line of text by inspecting the vertical projection (Sarfraz et al., 2003; Taha et al., 2012). (See Figure 2.6). This method depends on the estimation of the minimum space between words. However, it was pointed out in the Arabic script characteristics section above that some Arabic characters are not linked with succeeding letters, thus this results in a word having with one or more connected components (sub-words), as shown in Figure 2.6. To overcome this issue, methods based on vertical projection consider that the width of spaces between sub words is smaller than the width of the spaces between words (Sarfraz et al., 2003).

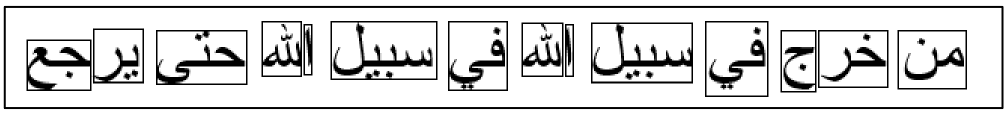


Figure 2.6: An example of line segmentation

Generally, it is relatively easy to segment a text line into words in printed text images, compared to handwritten text images which involve overlapping and touching characters by using vertical projection histogram profiles (Alginahi, 2010; Lawgali, 2015; Al-Shatnawi et al., 2011) . However, some Arabic fonts contain characters that vertically overlap, such as the Traditional font type. Thus, Arabic script even in printed form can contain touching and overlapping characters, so algorithms that have been designed to overcome this challenge for handwritten script may be utilized for printed Arabic. For example, AlKhateeb et al. (2009) have developed a method based on the connected components that analyses the distance between connected components in order to segment handwritten words.

2.3.2.2 Internal segmentation

Internal segmentation deals with segmenting a word into characters. When reviewing segmentation methods in the literature, a major complication arises concerning the classification of word segmentation approaches. For instance, Khorsheed (2002) classifies Arabic OCR systems based on word segmentation into ‘segmentation based systems’, which is based on analytical techniques where a word is segmented into characters, and ‘segmentation–free systems’, which is based on recognizing a word as a unit without segmentation. Amin (1998) and Xiu et al. (2006) discuss word segmentation in terms of implicit and explicit segmentation. Others classify word segmentation in terms of techniques which have been applied to segmenting a word, such as Lawgali (2015), Zeki et al. (2011) and Alginahi (2013). Al-Shatnawi et al. (2011) organize segmentation methods for Arabic script into holistic approaches and analytical approaches.

Mostly, Arabic OCR systems have been developed by two main paradigms: holistic approaches (segmentation–free) which require a large lexicon of Arabic words, and analytical approaches (segmentation based) where a word is segmented into units and each unit is recognized separately.

a- Holistic approach

Segmentation-free or holistic Arabic OCR systems perform the recognition on the entire word as a unit without segmenting the word or recognizing characters separately (Al-Badr and Mahmoud, 1995). Several studies have investigated the holistic approach for printed Arabic script OCR such as Erlandson et al. (1996), Al-Badr and Haralick (1998) and Khorsheed and Clocksin (2000). OCR systems based on a holistic approach require tracing the feature of the entire word and dealing with words instead of characters. As a result, this approach is restricted to recognizing a word against a lexicon (Al-Badr and Mahmoud, 1995; Jain et al., 2018). Moreover, this approach has the challenge of how to deal with the large lexicon size of Arabic words (Nashwan et al., 2017). It is claimed that systems based on this type of segmentation are not useful for general text recognition. Choudhary (2014) suggests this approach for systems in which a lexicon is statically defined, such as bank cheque recognition where vocabulary is limited.

b- Analytical approach

For the analytical or segmentation-based approach, Arabic OCR systems segment words into smaller units like characters (see Figure 2.7). In the typical Arabic OCR system, the analytical approach is divided into two approaches: explicit segmentation and implicit segmentation.

1. Explicit segmentation

The explicit segmentation approach, which is also called dissection segmentation, attempts to segment a word into smaller units. These units could be characters, strokes or loops. Naz et al. (2015) argue that there are two classes of explicit segmentation, which are: direct segmentation and indirect segmentation. In the former, a word is directly segmented into characters exploiting a set of heuristics, while in the latter, a word is divided into smaller segments which can be characters or marks that over segmented characters, such as strokes.

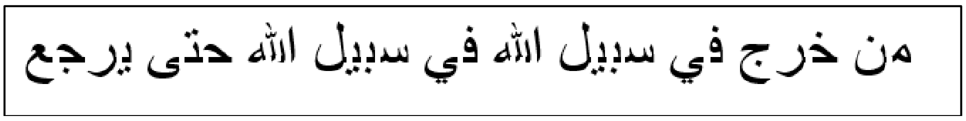


Figure 2.7: Segmenting Arabic words into their characters

Projection analysis is considered as one of the earliest applied dissection methods on Arabic character segmentation (Amin and Masini, 1986; Ymin and Aoki, 1996; Zeki, 2005). The projection method of the text image aims to reduce 2D information into 1D in order to simplify the character segmentation process. A method based on a modulated histogram of the image has been proposed by Najoua and Noureddine (1995). However, this method has been tested on specific Arabic fonts which do not contain overlapping and ligatures. Consequently, this method would not be appropriate for Arabic fonts that have ligatures, such as traditional Arabic font (Alginahi, 2013; Amara et al., 2016).

Another histogram projection method is presented for printed Farsi word segmentation by Parhami and Taraghi (1981) which is also applicable to Arabic script, as Arabic script is similar to Farsi script (Alginahi, 2013). However, this method is font dependent and ineffective in segmenting small font sizes. Although many of the other techniques based on projection analysis have been devolved for Arabic script such as by Zheng et al. (2004), Amara et al. (2016) and Lorigo and Govindaraju (2005), it seems that no projection based segmentation algorithm is accurate in segmenting Arabic text (Parvez and Mahmoud, 2013).

Instead of applying projection analysis methods, contour–based algorithms, which are used for dissection segmentation that rely on the skeleton or contour of Arabic words, are used to simplify the Arabic word segmentation such as in Xiu et al. (2006). Other methods rely on white space and pitch finding techniques for segmenting Arabic words (Al-Yousefi and Udpa, 1992; Zeki, 2005). However, a major criticism of the explicit approach is that it is expensive because of the requirement of finding the optimum word from the arrangement of segmented units (Naz et al., 2015). Cheung et al. (2001) conclude that an accurate segmentation may not be acquired by relying on dissection segmentation approaches.

2. Implicit segmentation

In OCR systems based on implicit segmentation, the segmentation phase and recognition phase are performed simultaneously (Alginahi, 2013). In other words, a word is segmented into characters while being recognized without segmentation in advance (Zeki, 2005). Straight segmentation and recognition based segmentation are also referred to as implicit segmentation (Zeki, 2005). This segmentation approach searches the text image for components that match predefined classes. The principle of implicit segmentation is to utilize a sliding widow to segment the word image into frames of fixed width on which classification relies to make a decision (Koteswara Rao and Negi, 2016). Owing to challenges in segmentation of cursive scripts such as Arabic, researchers use the implicit segmentation approach in order to overcome the problems of word segmentation (Choudhary, 2014). In principle, by applying this type of segmentation, there is no need for a specific dissection algorithm for Arabic script segmentation and the accuracy performance relates to the classification performance (Zeki et al., 2011). Thus, some researchers implement techniques based on implicit segmentation in order to improve recognition accuracy of Arabic OCR, such as Cheung et al. (2001), Al-Badr and Haralick (1995) and Chen and DeCurtins (1993).

2.3.3 Feature extraction phase

Once the text image is segmented into isolated regions (such as character, part of character), the next step is feature extraction which is the process of obtaining distinguishing attributes of the segmented character to be utilized by the next phase which is classification (Srinivas et al., 2008). Feature extraction is the most significant level that heavily influences overall OCR performance (Al-Shatnawi et al., 2011; Lawgali, 2015; Naz et al., 2014). The feature extraction stage is correlated with other OCR stages, such as preprocessing and classification stage. In other words, Due Trier

(1996) points out that the selection of feature extraction methods depends on the output of the preprocessing stage. For instance, some techniques for feature extraction work on skeletons, whereas others work on grayscale images. Moreover, the set of features extracted must match the specification of the selected classifier (Al-Badr and Mahmoud, 1995).

In terms of OCR performance, feature extraction plays a critical role in achieving high accuracy performance (Lawgali, 2015), since the feature extraction stage contributes to the success of the classification step (Naz et al., 2014). However, selection of feature types is a major issue in OCR development (Moubtahij et al., 2014). Researchers recommend that the feature extraction methods should be independent of scalable font characteristics such as font styles, font types, font sizes and should be able to describe and distinguish different patterns effectively (Moubtahij et al., 2014; Sharma et al., 2012). In other words, Al-Muhtaseb and Qahwaji (2011) emphasize that the key purpose of selecting good features is to maximize the effectiveness and the efficiency of the OCR system minimizing the complexity and processing time simultaneously.

Among OCR system development, researchers propose various types of features. Such features can be categorized into three groups: structural features; statistical features; and global transformation features (Khorsheed, 2002). In the following, these features will be discussed in the context of recognizing Arabic script.

2.3.3.1 Structural features

Structural features illustrate a text image in terms of its topological and geometrical characteristics by using its local and global properties (Al-Badr and Mahmoud, 1995; Moubtahij et al., 2014). In case of Arabic script, lines, dots, loops, holes, strokes and zigzags are some structural features (Moubtahij et al., 2014; Abd et al., 2012; Saabni, 2015; Elgammal and Ismail, 2001; Khorsheed and Clocksin, 1999; Ahmad et al., 2013). Considering Arabic script characteristics, some characters have common primary shapes and they can only be differentiated by the number and location of their dots. Thus, Lorigo and Govindaraju (2006) claim that structural features have been commonly used for Arabic script in order to capture the dot information of characters explicitly.

On the other hand, Ghosh et al. (2010) argue that structural feature methods are not capable of discriminating between characters having similar shapes. Similarly, a study by Moubtahij et al. (2014) reports that relying on the structural features of Arabic script may result in misrecognition, owing to the small difference between Arabic letters. Khorsheed (2002) comments that extracting structural features of Arabic characters is a challenging task. Furthermore, it is claimed that Arabic OCR systems using structural feature methods suffer from exhaustive processing time (Almohri et al., 2007). Likewise, various studies have reported that another complication of applying structural features is that it involves expensive preprocessing techniques, such as skeletonization which may result in character shape distortion and loss of structural feature data (Naz et al., 2014; Lorigo and Govindaraju, 2006; Hossain, 2012). Therefore, research on Arabic OCR has been carried out on other feature extraction approaches, as will be discussed below, that are effective in reducing process time and improving performance accuracy (Almohri et al., 2007).

2.3.3.2 Statistical features

Statistical features are derived from statistical representation of patterns which provide a measurable event of interested patterns. Researchers in Arabic OCR systems adopt different approaches to produce statistical features. Some examples of the approaches, which have been applied for representing Arabic characters, are zoning, moments, characteristic loci, histograms and crossing (Moubtahij et al., 2014; Khorsheed, 2002; Al-Badr and Mahmoud, 1995; A. M. Al-Shatnawi et al., 2011).

The zoning method divides the character image into several overlapping and non-overlapping regions. Then, the density of each region pixel is analysesd and used as a feature (Moubtahij et al., 2014; Kumar and Bhatia, 2014).

According to Ahmed et al. (2012), the moment method is a common statistical feature approach that has been applied in patter recognition applications. Moments, including Legendre moments, Zernike moments, central moments, pseudo-Zernike moments and Hu moments, extract geometric features in an image, such as, the shape area of a pattern and the center of the mass (AbdelRaouf, 2012; Aboaisha et al., 2012; Kef et al., 2012). Several studies in printed Arabic script, such as, Abd and Paschos (2007), Oujaoura et al. (2012), Shaaban (2008) and Elrube et al. (2010), have applied moment invariants as a feature vector.

In short, it is claimed that statistical features for pattern representation are easy to extract (Moubtahij et al., 2014; Al-Badr and Mahmoud, 1995). Moreover, such features can be effective in recognition systems and providing high speed and low complexity implementation (Naz et al., 2014; Pati and Ramakrishnan, 2005). However, special attention to the prepossessing techniques should be given, since misrecognition may accrue due to poor prepossessing techniques (Khorsheed, 2002). Nevertheless, the fundamental issue is to determine a set of statistical features, which need to be the most representative data of a pattern, maximizing the performance accuracy and minimizing the processing time simultaneously.

As a result, researchers call for investigating other statistical features which maximize the performance accuracy and minimize the processing time (Al-Badr and Mahmoud, 1995; Arica and Yarman-Vural, 2001).

2.3.3.4 Global transformation features

The global transformation method is applied to convert a skeleton or contour of a pattern by a linear transform into a form that reflects the most relevant features of the transformed pattern (Arica and Yarman-Vural, 2001). Numerous global transformation methods have been used in developing Arabic OCR systems. An example of such methods is the Fourier descriptor which represents the characteristic of a pattern in a frequency domain (Zhang and Lu, 2001). The Fourier descriptor has been applied to Arabic script, such as in Khorsheed and Clocksin (2000) and Mahmoud (1994). Another method is the Hough transform which detects lines in binary images and then defines the parameters of the lines (Cheriet et al., 2007). Amor and Amara (2006) and Touj et al. (2005) utilized the Hough transform for extracting features from Arabic script. Also, some other global transformation methods that have been applied for Arabic OCR for feature extraction are the direction codes method such as Freeman's chain code in Taha et al. (2012), Wavelets in Amor and Amara (2012) and Walsh transformation in Oujaoura et al. (2012).

Overall, it is claimed that global transformation feature techniques have several advantages over structural and statistical approaches. For example, they are applicable for new fonts and easily implemented. Another advantage is that they are robust to noise and variation. However, they might require the implementation of other features in order to obtain high accuracy performance.

In conclusion, the feature extraction stage plays a critical role in Arabic OCR development in which distinguishing attributes are extracted and it is clear that each

Arabic OCR developer needs to apply different feature extraction approaches. Still, good features are required, which assist in distinguishing a character from other characters and maximize the accuracy performance simultaneously. Furthermore, these features must be selected specifically for a selected classifier. Some researchers apply different feature extraction methods in combination. However, this may cause extra complications for the implementation (Khorsheed and Clocksin, 2000).

2.3.4 Classification phase

The classification phase has the responsibility for assigning a pattern into a pre-classified class based on the features of the pattern which have been extracted in the previous phase (Cheriet et al., 2007). The pre-classified classes can be words, sub-words, characters or strokes, based on the OCR approach used (Al-Helali and Mahmoud, 2017). Several different classification approaches have been used for many applications, such as text classification, speech recognition and text recognition. Different classifiers have been used for printed Arabic text recognition, such as Hidden Markov Models (HMM), Support Vector Machines (SVM), K-nearest neighbour and Neural Networks (NNs).

SVM, which is a binary classifier, has been used in the implementation of printed Arabic OCR systems (Abd and Paschos, 2007; Abd et al., 2012; and Mehran et al., 2005). (For a comprehensive review of applying SVM to Arabic OCR, refer to Amara et al. (2014)). However, classifiers based on SVM are mostly applied to a small set of data due to the high complexity of training and processing time (Shafii, 2014; Arora et al., 2010). Another classification technique that has been applied to printed Arabic OCR are HMM based techniques. HMMs are statistical models that are considered as being one of the most efficient for recognition applications especially for speech recognition (AbdelRaouf, 2012). Therefore, researchers in OCR have implemented HMMs for OCR in order to obtain high performance OCR systems, such as in Khorsheed (2007), Khorsheed (2015), Awaida and Khorsheed (2012), Slimane et al. (2013), Shaker (2018), Rahal et al. (2018) and Al-Muhtaseb et al. (2008).

Neural Networks (NNs) based methods have been successfully applied in many NLP applications, such as text recognition (Al-Ayyoub et al., 2018). NNs refer to the non-linear system that may be constructed according to a specific network topology

(Khorsheed, 2002). There are different architectures of NNs which consist of an input layer, an output layer and a single or multiple hidden layers (Hamza, 2008). According to Alginahi (2004), NNs based methods can successfully improve computation time since they consist of parallel processing consisting of several connections of the same neural processes. Moreover, NNs have the capability to adhere to data changes as they are adaptive (Alginahi, 2013). Extraction features from raw images can be done automatically by using NN based methods (Sawy et al., 2017). However, NNs require large training data (Nashwan et al., 2017).

Several NN methods, such as Deep Neural Networks (DNNs), Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs), have been used as classifiers for Arabic text recognition and resulted in achieving promising accuracies. Owing to the challenge of Arabic segmentation, researchers developed free-segmentation Arabic OCR systems that employ NN based methods (Nashwan et al., 2017). NN based methods such as RNNs have the ability to recognise unsegmented characters as they work on the sequence of characters (Younis and Alkhateeb, 2017). For example, Radwan et al. (2018) proposed an OCR system for Arabic text using multichannel neural networks in order to overcome the challenge of character segmentation of Arabic text. They implemented three neural networks. In the first network, the font pitch of an Arabic word is predicted. In the second network, the Arabic word is segmented into its characters. Then, the segmented characters are fed as input to a CNN by which salient features are automatically extracted. The system was evaluated on a word-level dataset and achieved an accuracy rate of 94.3% for an Arabic transparent font of 18 font pitch.

Ko et al. (2018) proposed an OCR system for recognising printed Arabic text using a deep-learning approach. They used the multi-dimensional bi-direction long short-term memory (MD-LSTM) with the use of connectionist temporal classification. The system was trained on one-million Arabic text line images. The system achieved an accuracy rate of 99.9%. Similarly, Rashid et al. (2013) introduced a printed Arabic OCR system based on the MD-LSTM approach. The system achieved an accuracy rate of 99.1% on a word-level dataset. In addition, multiple classifiers has been utilized in order to improve the accuracy of classification; for review see (Rahman and Fairhurst, 2003).

There are a number of studies that compare the performance of different classifiers. For instance, Amara, Ghedira, Zidi, & Zidi (2015) conduct a performance evaluation experiment on multi-class support vector machines using an Arabic character dataset. In addition, in (Pal et al., 2009), performance of different classifiers have been evaluated on Devanagari characters. However, the performance of classification methods on text recognition systems are affected by various factors, such as the pre-processing method, feature extraction techniques and training and testing datasets (Liu and Fujisawa, 2008). Therefore, it is challenging to assess the performance of a classifier based on reported performance accuracies in different studies, since the studies do not use standard pre-processing methods, feature extraction techniques and datasets.

Interestingly, compression based methods (models) for classification have been successfully applied in different text classification applications (Alkhazi et al., 2017). For instance, model-based text classification using the PPM compression method, which will be discussed in the next sub-section, has been considered for a number of different classification applications. The PPM model has been found to be competitive in different classification tasks (Teahan, 2018). For example, Teahan & Harper (2003) use PPM models for topic classification and report results that PPM outperforms other classification methods, such as SVM and Naïve Bayes classifiers. In addition, the PPM based compression models have been used accurately for gender classification (Altamimi and Teahan, 2017), emotion classification (Almahdawi and Teahan, 2017), and text classification (Teahan, 2018; Alkhazi and Teahan, 2017). Although many studies consider the use of compression-based methods as classifiers, no prior study exists on the use of compression-based models as a classifier in Arabic text recognition. Although it has been shown in the literature above that Arabic OCR systems that applied NNs produced competitive results, they require a large amount of training data. However, the compression based method used in this thesis can be used as a classifier with comparatively smaller training data (as described in Chapter 7).

2.3.4.1 Prediction by Partial Matching (PPM)

PPM is an online adaptive text compression system that foretells the following character or symbol by using the previous context with given fixed length. Models that set their prediction probabilities on a few preceding characters are called order n

finite-context models, where n indicates the number of preceding characters used. The model's order represents the maximum length of context used to predict the next character. To predict upcoming characters, PPM uses a set of fixed order context models with different values of n.

For each model, a record is kept of the frequency counts all symbols that have followed every subsequence observed in the input from 1 to length n with the number of times that each has occurred. Prediction probabilities are estimated by using these frequency counts. The probabilities associated with each symbol that has followed the preceding n symbols in the past are used to predict the forthcoming symbol. Therefore, a single probability distribution is obtained from each model.

PPM combines these distributions into a single model and arithmetic coding is used for encoding the symbol according to its associated probability. The combination of different probability distributions into one model is achieved through the insertion of the *escape* symbol. PPM starts with the model's maximum order first to encode the forthcoming symbol (Teahan et al., 2000). However, if a specific model does not contain the input symbol, which means that the input symbol has not been seen previously, an *escape* symbol is inserted at the transition from the current order model to the lower order model (a model of order n-1). That is, if the model does not contain the symbol, then the encoder will back off to a default order of -1 (Teahan et al., 2000).This process of escaping will be repeated until the model finds that symbol or prediction. If necessary, when completely unseen symbol is encountered, the encoder will back off to the lowest order context of -1 in which all symbols are encoded with a probability of $\frac{1}{|A|}$, where A is the size of the alphabet that is used.

PPM uses a Markov-based character n-gram procedure which applies a back-off mechanism alike to that suggested by Katz (Katz, 1987). Nevertheless, PPM refers to the backing-off as "escaping" and it was developed before Katz's proposed mechanism. It was first proposed by Cleary and Witten (Cleary, John and Witten, 1984) in 1984 when they developed the character-based PPM variants, PPMA and PPMB. Then came Moffat and Howard, in 1990 and 1993, who introduced two further variants of PPM, PPMC and PPMD (Wu, 2007). The main distinction among these variants of PPM, is the estimation of the escape probability that the smoothing mechanism requires for backing off to a reduced model's order. Experiments for character streams have shown that PPMD usually delivers better compression results when compared to other variants of PPM (Khmelev and Teahan, 2003).

The following equation is used to determine the probability p of the following character φ using PPMD (Teahan, 2000):

$$p(\varphi) = \frac{2c_d(\varphi)-1}{2T_d} \tag{2.1}$$

where the coding order currently used is indicated by d, the number of times where the current context has happened or occurred in total is represented by T_d and the number of times that the context has followed by the symbol φ is represented by c_d (φ). The estimate of the escape probability e by PPMD is as follows:

$$e = \frac{t_d}{2T_d} \tag{2.2}$$

where t_d represents the number of types of unique symbols that occur following the current context.

For example, if a specific context has occurred three times before, with three characters *a*, *b* and *c* following it once, according to equation 2.1 and 2.2, the probability for each character will be $\frac{1}{6}$ and the escape probability will be $\frac{3}{6}$. In the escape probability, 3 represents the number of types as there are three types, *a*, *b* and *c*.

In most experiments, the use of 5 as a maximum order has proven to be efficient, as PPMD starts with the model's maximum order first to encode the forthcoming symbol (Teahan et al., 2000).

If the forthcoming symbol was predicted by the current model, then its probability in the current maximum order, 5 in this case, will be used to transmit it. If the forthcoming symbol was not found in the model, then the encoder will escape to the next lower order model, 4 in this case. This process of escaping will be repeated until the model finds that symbol or prediction. If the model does not contain the symbol, then the encoder will back off to a default order of -1 (Teahan et al., 2000).

To explain character-based encoding in more detail, Table 2.2 presents the way PPM models a given string. The example in this case is using the PPMD prediction method to model the string '*tobeornottobe*'. The model in this example uses a maximum model order of 2 for illustration purposes (although normally it would be order 5). The first and the second columns contain nine and six different predictions, respectively. Each column contains an escape probability whose count *c* is equivalent to the number of unique symbols that have been seen in the given string. Therefore, the escape symbol the third column (order 0) has a count of 6 because six unique symbols have been seen in the given string. The final column at the bottom of the table that has the lowest order context of -1 will be used when a symbol is completely unseen. In the table, *c* indicates the count, *p* symbolizes the probability and |*A*| is the size of the alphabet that is used (Teahan et al., 2000).

Let the imminent character for this example be letter *o*. The letter has been seen once before (‘be’ → o) for the order two context ‘be’ and therefore it has a probability of $\frac{1}{2}$, applying equation (2.1) since the count is 1, and as a result $(-\log_2 \frac{1}{2} = 1)$. Therefore, the letter *o* will be encoded using 1 bit. But if the upcoming letter in the order two context had not been seen before, (i.e. suppose the next letter was *t* rather *o*), then the model would need to escape to a lower order, the escape probability will be $\frac{1}{2}$, and the model will back off to the order 1 context.

When the model backs off, the new order will be used to estimate the probability, and in this case, there is no letter *t* that comes after *e*. As a result, the model will encode another escape using a probability of $\frac{1}{2}$, and the context will be reduced to the null (order 0) context. Letter *t* will be encoded using this order, where the probability will be $\frac{5}{26}$. The total cost of predicting the last letter will be $\frac{5}{26} \times \frac{1}{2} \times \frac{1}{2}$, which in this case will be over 4 bits. Moreover, if the following letter has not been seen before in the context, such as letter *x*, the escape probability will be encoded three times from the maximum order of 2 to -1 with the following probabilities:$\frac{1}{2} \times \frac{1}{2} \times \frac{3}{13} \times \frac{1}{256}$ since order -1 will be used to encode this letter since we are encoding English characters using ASCII (with an alphabet size of 256), and this will require over 12 bits (Teahan et al., 2000).

Order 2		
Prediction	c	p
‘be’ → o	1	1/2
→ esc	1	1/2
‘eo’ → r	1	1/2
→ esc	1	1/2
‘no’ → t	1	1/2
→ esc	1	1/2
‘ob’ → e	2	3/4
→ esc	1	1/4
‘or’ → n	1	1/2
→ esc	1	1/2
‘ot’ → t	1	1/2
→ esc	1	1/2
‘rn’ → o	1	1/2
→ esc	1	1/2
‘to’ → b	2	3/4
→ esc	1	1/4
‘tt’ → o	1	1/2
→ esc	1	1/2

Order 1		
Prediction	c	p
‘b’ → e	2	3/4
→ esc	1	1/4
‘e’ → o	1	1/2
→esc	1	1/2
‘n’ → o	1	1/2
→ esc	1	1/2
‘o’ → b	2	3/8
→ r	1	1/8
→ t	1	1/8
→ esc	3	3/8
‘r’ → n	1	1/2
→ esc	1	1/2
‘t’ → o	2	1/2
→ t	1	1/6
→ esc	2	1/3

Order 0		
Prediction	c	p
→ b	2	3/26
→ e	2	3/26
→ n	1	1/26
→ o	4	7/26
→ r	1	1/26
→ t	3	5/26
→ esc	6	3/13

Order -1		
Prediction	c	p
→ A	1	1/|A|

Table 2.2: Processing the string ‘*tobeornottobe*’ using PPM
(Teahan et al., 2000)

2.3.5 Post-processing phase

Post-processing is the final stage of the development of Arabic OCR. The objective of this step is to enhance the recognition accuracy by detecting and correcting linguistic misspellings in the produced OCR text without human intervention. Research studies on Arabic OCR have implemented post-processing methods in order to improve the output, such as Doush and Trad (2016) and Magdy and Darwish (2008). It is worth mentioning that three main elements should be considered in correcting OCR output: non-word errors correction; isolated word errors correction; and context–based word correction (Parvez and Mahmoud, 2013). Generally, post-processing methods can be categorized into two main approaches: lexicon-based methods; and context-based (statistical) methods (Taghva and Stofsky, 2001).

The typical technique for correcting the mistakes of Arabic OCR outputs is the lexicon-based method which requires the utilization of an Arabic dictionary, such as in Hassin et al. (2004) and Aljarrah et al. (2012). This technique corrects errors without considering any contextual information in which the errors appear. Therefore, a problem might occur with using this approach when a word is misrecognized by an OCR system and is also in the lexicon (these are called real-word errors) such as, *Fear* for *Tear*. This occurs in many languages such as Arabic in which a large fraction of three characters sequences are corrected words. Consequently, only non-word errors can be corrected, since this method is comparing the recognized words with the words that are in the dictionary. Also, this approach requires a wide-ranging lexicon that consists of all single words. However, the Arabic language has various dialects and it is also a triglossic language with three forms – modern standard Arabic and classical Arabic (Damien et al., 2009) and mixtures of the two. Therefore, this approach is less appropriate for Arabic language since building a single lexicon for Arabic language is more complicated.

On the other hand, context-based (statistical) methods take into account the contextual information in which the misrecognized words appear. A few studies have implemented statistical language models for improving the recognition accuracy of Arabic OCR systems, such as in Prasad et al. (2008), Doush et al. (2018) and Natarajan et al. (2008). Using such methods will help overcome the problem of correcting real-word errors. Moreover, they are also useful in correcting word errors that might have several potential corrections, since these techniques can correct word errors based on grammatical concepts and semantic context (Bassil and Alwani, 2012). Teahan et al. (1998) have applied a statistical approach by using the PPM compression-based language model for correcting English text. The PPM algorithm predicts a probability distribution for the upcoming characters by using the previous characters, as described above in Section 2.3.4.1. For Arabic text, a study by Alhawiti, (2014) investigated the possibility of using the PPM language model for correcting Arabic OCR output text in order to improve the overall accuracy of the OCR system. This study had shown a significant improvement of the OCR system's performance when only correcting single-character errors.

Recently, there have been several attempts to provide systems for correcting Arabic OCR output. For instance, Doush and Trad (2016) propose a system for Arabic OCR output correction based on Google online suggestions within Microsoft Office Word. Bassil and Alwani (2012) describe a context-based technique for detecting and correcting Arabic OCR errors. Although there are some studies on applying context-based methods for correcting Arabic OCR output, more research is needed on investigating the use of Arabic contextual information for OCR output correction (Krayem, 2013).

2.4 Performance evaluation

OCR performance evaluation can be classified into two types: black-box evaluation and white-box evaluation. In the former, an entire OCR engine is treated as an indivisible unit, so the submodules of the OCR system are not known to the evaluator, whereas with the white-box evaluation, each submodule of the OCR system is evaluated if the submodules are accessible (Rice, 1996). Performance evaluation of OCR systems is essential for monitoring progress of OCR systems development, assessing the effectiveness of OCR algorithms, identifying open areas for further research and providing scientific explanation for the performance of OCR systems (Mihov et al., 2005; Kanungo et al., 1999a). Therefore, to produce an efficient Arabic OCR system, effective performance evaluation of current OCR systems is essential.

Despite the significance of Arabic OCR system performance evaluation, relatively little work has been published on empirical analysis of the effectiveness of Arabic OCR systems. For instance, two studies provide an evaluation of two Arabic OCR systems (Kanungo et al., 1999a; Kanungo et al., 1999b). However, as these studies were conducted in 1999, over 17 years ago, they do not reflect current progress in the field of Arabic OCR development (Alginahi, 2013). Some competitions have been held by international conferences for printed Arabic text recognition in order to evaluate different OCR systems. For example, a competition was presented at the 11th International Conference on Document Analysis and Recognition (ICDAR-2011) (Slimane et al., 2011). The main objective of this competition was to assess the impact of font pitch sizes on the performance of OCR systems. A recent competition was organized by the 14th International Conference on Document Analysis and Recognition (ICDAR-2017) (Slimane et al., 2018). In both competitions, only a word-level dataset, in which each text image contains one Arabic word, was used for evaluating printed OCR systems and technique. A more recent empirical study provides a comparative evaluation of the most common Arabic OCR systems (Saber et al., 2016). However, the study by these authors only investigated the effectiveness of input quality images on the performance of Arabic OCR systems. In addition, exploring Arabic OCR systems in relation to their sensitivity to different levels of page quality may not be adequate in fairly assessing their success, as some OCR systems include a combination of image enhancement techniques. To the best of the author's knowledge, no established work has gaged the current progress in the enhancement of Arabic OCR in terms of the challenges of Arabic script.

Moreover, the performance assessments of Arabic OCR systems are only reported by their developers: their results are derived from different datasets that might be small or might be used in developing the systems (Al-Badr and Mahmoud, 1995; Alginahi, 2013). Consequently, as these performance tests are statistically invalid, they cannot be used to compare the performance between Arabic OCR systems (Margner and El Abed, 2009; Alginahi, 2017). Evaluating the performance of Arabic OCR systems is also challenging as no standard dataset is available nor is a set of performance metrics freely available to the community of Arabic OCR developers (Al-Muhtaseb and Qahwaji, 2011; Abdelraouf et al., 2008; Al-Badr and Mahmoud, 1995; Ahmad et al., 2016). In addition, most reports on the performance of Arabic OCR systems are in terms of the general, standard performance measurement of character accuracy, such as in Dahi et al. (2015) and Ahmad et al. (2016). However, this performance metric is insufficient to assess how Arabic OCR systems are coping with the challenges of Arabic script.

On the other hand, to obtain statistically meaningful results when evaluating the performance of OCR systems, a great number of text images are needed to be tested (Rice, 1996). Although tools exist for other languages (e.g. English (Carrasco, 2014), (Nartker et al., 2005)), none exists for Arabic. Therefore, a programmed test tool is needed to automate the evaluation test. Additionally, conducting the performance evaluation under an automated tool will result in eliminating human error, improve speed, precision and reduce repetition (Saber et al., 2016). From the experimental studies on Arabic OCR evaluation, only two studies—(Saber et al., 2016, 2014)—indicate that an automated tool was utilised to evaluate the performance of Arabic OCR systems. Unfortunately, the utilised tool is not available for researchers and is limited in accuracy metrics.

2.5 Challenges and open problems

This chapter has overviewed the main stages used in printed Arabic OCR. It main aim is to reveal the current status of printed Arabic OCR. Although there are various attempts to solve the problems of Arabic text recognition, there is still a crucial need for more research.

In an attempt to evaluate the status of printed Arabic OCR and support the claim that more research is needed in many areas, we used Google scholar to search for scientific research publications, in March 2018, using phrases that are related to Arabic text recognition. The findings are summarized in Table 2.3. The table shows the search phrases used and the search results returned by Google Scholar. It is apparent from the table that there is a lack of Arabic OCR research as comparatively very little research has focused on Arabic OCR compared to studies in OCR for other languages. For example, there were 322,000 results returned for the more general search query ‘OCR’, whereas there were only 956 results returned for the more specific search query ‘Arabic OCR’.

In order to provide a measure of the coverage of research in a particular area, we can estimate the probability that a particular research paper will be in a more specific topic area compared to the more general topic area. For example, we may be interested in the general topic area “single font OCR” and wish to see how much research has been published in the more specific topic area “single font Arabic OCR” in comparison. We can estimate the probability p that the more general topic will be concerned with the more specific topic as $p = s / g$ where s is the count of the number of papers found for the specific topic compared to the count g of the number of papers found for the more general topic. Then we can define the ‘Information Coverage’ I associated with the specific topic in relation to the more general topic as $I = – \log p$. If this value is high compared to other specific vs. general comparisons for the same overall topic (e.g. in relation to Arabic OCR vs. OCR in general), then this reflects that research may be under-represented in this area.

This analysis has been done using the values from Table 2.3 and graphed in Figure 2.8. In the Figure, we see that except for papers on Arabic OCR concerning easy fonts and diacritics, the remaining topics have higher Information Coverage values meaning that there have been less papers published in these areas proportionately compared to papers published in the more general (non-Arabic) areas. We can use Figure 2.8 to help gage the present status of printed Arabic OCR research as it highlights some open areas which need more research. This is based on the number of publications for single, omni and multi font OCR concerning various elements that are related to text recognition concerning easy fonts, complicated fonts, diacritics, page layout, multi-language and noisy documents. In particular, for Arabic text images which contain complicated fonts, there are still many gaps in the research. Furthermore, for single, omni and multi font Arabic OCR on multi-language text images, intensive further research is needed.

Figure 2.9 plots the number of papers per 5-year period for the top 100 Google Scholar searches using the Arabic OCR related phrases. A number of striking results are apparent in Figure 2.9. For example, publications for Arabic OCR peak after 2005. Also, the numbers of papers for printed Arabic OCR decrease since 2005 (this could be because researchers have focussed on handwritten Arabic OCR). Furthermore, it is apparent that the smallest numbers of papers in the period of Arabic OCR research are papers related to noisy documents.

Note that the earliest research papers for the ‘Arabic OCR’ search query are from 1985, which is a result of the top-100 ranking returned by Google Scholar. If, however, we restrict the range of years for which we search, we find that the first papers returned by Google Scholar appear in the 1970 to 1980 period. In contrast, Bader (1995) states that text recognition research first originated in 1940, and papers related to the 'text recognition' query appear in Google Scholar from the 1960s.

	Google Scholar search phrase	Number of papers
1	+"Arabic OCR"	956
	+"OCR"	322,000
2	+"OCR" + "Arabic printed text"	247
	+"OCR" + "printed text"	7,190
3	+"Arabic OCR" + "diacritics"	302
	+"OCR" + "diacritics"	1,800
4	+"Arabic OCR" + "page layout"	54
	+"OCR" + "page layout"	3,360
5	+"Arabic OCR" + "multi-language "	20
	+"OCR" + "multi-language"	866
6	+"Arabic OCR" + "Omni font"	60
	+"OCR" + "Omni font"	380
7	+"Arabic OCR" + "single font"	61
	+"OCR" + "single font"	717
8	+"Arabic OCR" + "multi-font "	149
	+"OCR" + "multi-font"	1,270
9	+"Arabic OCR" + "noisy document"	12
	+"OCR" + "noisy document"	515
10	+"Arabic OCR" + " Simplified Arabic " + "single font"	20
	+"Arabic OCR" + " Simplified Arabic " + "Omni font"	22
	+"Arabic OCR" + " Simplified Arabic " + "multi font"	48
11	+"Arabic OCR" + " Advertising Bold " + "single font"	5
	+"Arabic OCR" + " Advertising Bold " + "Omni font"	5
	+"Arabic OCR" + " Advertising Bold " + "multi font"	15
12	+"Arabic OCR" + "diacritics" + "single font"	30
	+"Arabic OCR" + "diacritics" + "Omni font"	36
	+"Arabic OCR" + "diacritics" + "multi font"	62
13	+"Arabic OCR" + " page layout " + "single font"	8
	+"Arabic OCR" + " page layout " + "Omni font"	9
	+"Arabic OCR" + " page layout " + "multi font"	11
14	+"Arabic OCR" + " multi-language " + "single font"	17
	+"Arabic OCR" + " multi-language " + "Omni font"	9
	+"Arabic OCR" + " multi-language " + "multi font"	29
15	+"Arabic OCR" + " noisy document " + "single font"	1
	+"Arabic OCR" + " noisy document " + "Omni font"	2
	+"Arabic OCR" + " noisy document " + "multi font"	3

Table 2.3: Google Scholar search results for Arabic OCR-related phrases

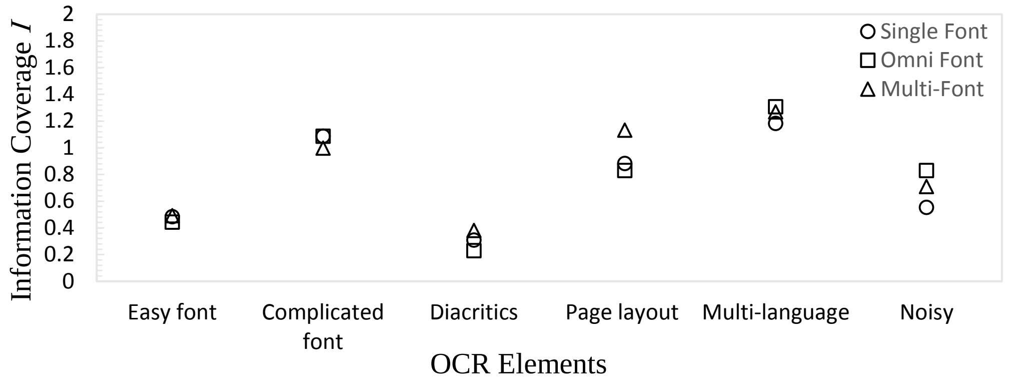


Figure 2.8: Present status of printed Arabic OCR based on the number of publications for different OCR elements

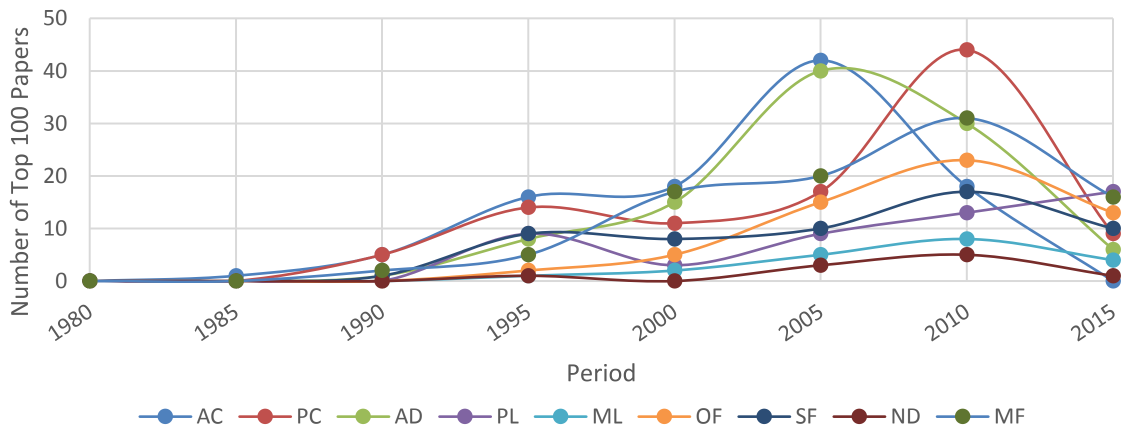


Figure 2.9: Number of papers per 5-year period in the top 100 results returned by Google Scholar for different Arabic OCR related search phrases

Notes: AC = 'Arabic OCR'; PC = 'Arabic printed text'; AD = 'Arabic diacritics'; PL = 'Arabic OCR + page layout'; ML= 'Arabic OCR + multi-language'; OF = 'Arabic OCR + omni font'; SF = 'Arabic OCR + single font'; MF = 'Arabic OCR + multi font'; ND = 'Arabic OCR + noisy document'.

From the review of each stage used in Arabic OCR, the following observations have been noted:

1- All the reviewed research of printed Arabic OCR have used the general OCR methodology which involves the five stages; pre-processing; feature extraction; segmentation; classification and post-processing. However, the following are still open questions: 'Is the current OCR methodology the most effective for designing Arabic OCR?' and 'Are there alternative methodologies that might yield better results for Arabic OCR'?

2- From the review of each OCR stage, it is apparent that the most challenging task in the development of Arabic OCR is the segmentation task. Although previous studies have presented different segmentation techniques for Arabic OCR, these studies have not provided the accuracy performance of these techniques. Since only the overall OCR performances have been reported, it is difficult to gain an insight into which segmentation techniques perform better for printed Arabic OCR. A performance evaluation tool should be developed to assess the different segmentation techniques.

3- The pre-processing stage review given in this study reached the conclusion that direct adoption of pre-processing methods which are designed for general purposes might be not applicable for Arabic script. Thus, developing pre-processing methods that consider the specific characteristics of Arabic script is needed.

4- Most of the proposed methods for feature extraction in Arabic OCR have been adopted from methods that have been developed for other languages without considering the characteristics of Arabic script. Such methods may not be the most appropriate for accurate recognition. The characteristics of Arabic script should be taken into consideration when selecting a feature extraction method that is able to distinguish between Arabic characters.

5- The studies on performance evaluation of printed Arabic OCR have used a black-box evaluation method which can only provide the overall performance of OCR systems. For more insight into which OCR stage is causing the most problems, a white-box evaluation, where each component of the system is accessible, is required.

2.6 Summary

This thesis has provided a comprehensive literature review of printed Arabic text recognition. At first, the specific characteristics of Arabic script that challenge the recognition process have been discussed. Then, the general methodology of printed Arabic OCR has been presented. This methodology was divided into five stages: preprocessing; segmentation; feature extraction; classification; and post-processing.

Techniques applied at each stage of Arabic OCR have been discussed. Also, the issues related to the performance evaluation have been reviewed. Finally, we analyzed the challenges and the remaining problems in the field of printed Arabic OCR.

Chapter 3: Experimental Evaluation of Arabic OCR Systems

3.1 Introduction

Performance evaluation of OCR systems is an essential task for OCR systems development. However, relatively little work has been published on empirical analysis of the effectiveness of Arabic OCR systems. Moreover, studies in Arabic OCR suffer from the lack of proper performance evaluation metrics and the availability of automated evaluation tools. The review in chapter 2 showed that printed Arabic OCR poses great challenges owing to the special characteristics of Arabic text. Although the literature provides typical performance metrics, such as character accuracy and word accuracy for OCR performance evaluation, these metrics are insufficient to assess Arabic OCR systems in terms of the challenges of Arabic text.

Thus, this chapter first aims to propose an automated software tool with a new set of objective accuracy metrics for the evaluation of Arabic OCR systems with respect to the challenges of Arabic text. It also proposes a standard protocol which we hope will be used as a benchmark by researchers in comparing between OCR algorithms. It then aims to experimentally evaluate the effectiveness of the state-of-the-art printed Arabic text recognition systems to provide a better insight into their performances.

The work in this chapter has been published in the following papers:

- Alghamdi, M.A., Alkhazi, I.S. and Teahan, W.J., 2016. Arabic OCR evaluation tool. In *Computer Science and Information Technology (CSIT), 2016 7th International Conference on* (pp. 1-6). IEEE.
- Alghamdi, M. and Teahan, W., 2017. Experimental evaluation of Arabic OCR systems. *PSU Research Review*, *1*(3), pp.229-241.

This chapter is organised as follows: in section 3.2, a new set of objective accuracy metrics for Arabic OCR evaluation is discussed. The developed tool for the evaluation of Arabic OCR performance is described in section 3.3. The most common Arabic OCR systems and an experimental protocol are presented in section 3.4 and 3.5 respectively. The experimental results are then presented and discussed in section 3.6. In the final section, a summary and conclusion is provided.

3.2 Metrics for Arabic OCR system performance evaluation

Generally, Arabic OCR systems are evaluated in terms of character accuracy with this obtained by identifying the differences between an observed variable, which is the output text of the OCR system, and a reference variable, which is the original text called ‘ground-truth’ (Kanai et al., 1993; Teahan et al., 1998). These differences can be determined by the edit distance which is the minimum number of edit operations required to correct the OCR output text to be matched with the ground-truth text. These edit operations are: character insertion, character deletion and character substitution; for more details refer to Levenshtein (1966).

For example, the edit distance between the two strings, the ground truth string (جامعة بانجور) and the OCR-generated string, (حامعة ببانجور) is 2. The required operations to transform the OCR-generated string into the correct one are: (1) substitute the underlined letter in the OCR output (حـ) for (جـ); and (2) delete the underlined letter (بـ). This number, which is the minimum number of single character edit operations, is known as the Levenshtein distance (Levenshtein, 1966). Regarding OCR evaluation studies, it is common to determine the edit distance in different levels: word level and character level. However, character level accuracy is useful for predicting improvements in OCR systems in which OCR developers are interested, whereas word level accuracy is useful for analysing the ease of human readability, as Kanungo et al. (1999b) emphasise. Word accuracy is outside the scope of this of this study.

In the following sub-sections, we suggest various objective performance metrics with respect of the characteristics of Arabic text for evaluating Arabic OCR systems which can provide us with more insight into the effectiveness of Arabic OCR systems.

a. Overall character accuracy

Accuracy in an OCR-generated text in respect to the ground truth text is computed by Levenshtein edit distance; that is, the minimum number of primitive operations that are required to correct the OCR output text to be matched with the ground truth text. These operations are substitution, deletion and insertion.

Overall character accuracy determines the accuracy of Arabic OCR over all of the tested text images. Character accuracy is the percentage of ground-truth characters that are recognised correctly on an Arabic text image, by comparing the ground-truth text file with the OCR output text file. The character accuracy is determined by Equation (3.1):

$$\frac{m-e}{m}\times 100 \tag{3.1}$$

where m is the number of characters in the ground-truth text file and e is the edit distance. The cost for each edit operation is defined as 1.

b. Character accuracy based on character class

Considering Arabic script, some characters have features that allow them to be classified into a particular class. Consequently, it will be valuable to analyse the accuracy of each class. To determine the accuracy of this metric, Arabic characters have been classified into various classes as below:

1. Character accuracy based on character position

As mentioned in chapter 2, an Arabic character may have one to four shapes depending on its position in a word; for example, see Table 3.1. The four possible shapes are isolated, initial, middle and end. Thus, it is valuable to analyse the effectiveness of Arabic OCR systems in recognising isolated, initial, middle and end characters. To analyse the accuracy of this class, the ground truth Arabic characters are categorised into four classes: isolated, initial, middle, and end, as shown in Table 3.1. The accuracy of each class by using Equation (3.1).

Isolated	Initial	Middle	End
ه	هـ	ـهـ	ـه

Table 3.1. An Arabic character with dissimilar shapes

One dot	Two dots	Three dots	No-dot
ب ج خ ذ ز ض ظ غ ف ن	ت ق ي ة	ث ش	ح د ر س ص ط ع ل م ه و ك

Table 3.2. Examples of dot characters and no-dot characters

Dot character above baseline	Dot character below baseline
ت ث خ ذ ز ش ض ظ غ ف ن ق	ب ج ي

Table 3.3. Examples of dot characters based on the baseline

2. Dot character class and no-dot character accuracy

One of the challenges of Arabic OCR is the presence of dots in Arabic script. Assessing the impact of dot and no-dot characters on the performance of Arabic OCR systems is therefore of interest. To do so, Arabic characters are categorised into four classes: one dot class, two dot character class, three dot character class and no-dot character class, as illustrated in Table 3.2. The accuracy of each class is determined by Equation (3.1).

3. Dot character accuracy based on baseline

Dots of Arabic characters can be placed either above or below the baseline. Owing to the significance of the baseline in Arabic OCR, it would be valuable to compute the accuracy of characters that are most related to the baseline. Regarding the baseline, Arabic dot characters will be divided into two groups: above baseline and below baseline, as shown in Table 3.3. Each group accuracy is expressed by Equation (3.1).

4. Zigzag-shaped character accuracy

One of the distinguishing characteristics of Arabic script is the presence of a zigzag-shaped mark (ء), called *Hamza*, with some Arabic characters. The aim of using this metric is to expose the sensitivity of Arabic OCR systems to zigzag-shaped characters. The zigzag-shaped character accuracy is computed by using Equation (3.1).

ص ض ط ظ ـعـ ـغـ ف ق م ـمـ و ه ة

Figure 3.1. Loop characters

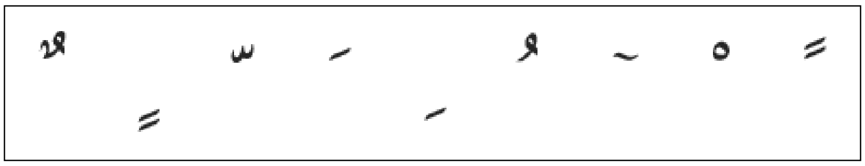


Figure 3.2. Diacritic marks

5. Loop-shaped character accuracy

When considering the characteristics of Arabic script, several letters consist of a loop shape. Such a feature may be an obstacle in Arabic OCR development. Thus, it would be desirable to assess the accuracy of characters that have this feature. In order to assess the accuracy of this class, loop characters are defined according to how they are presented in Figure 3.1. The accuracy of loop-shaped characters is also determined by Equation (3.1).

c. Diacritics accuracy

A major feature of Arabic script is the appearance of diacritical marks, as shown in Figure 3.2. Thus, it is essential to evaluate the performance of Arabic OCR systems in recognising Arabic diacritical marks. As before, diacritical mark accuracy is determined by Equation (3.1).

d. Digit accuracy

It is critical for developers to assess the recognition of numbers by Arabic OCR applications. In respect to Arabic script, both Hindu–Arabic digits and English numerals may appear in any Arabic text. Thus, the evaluation considers all numerals in computing the accuracy of digits. Digit accuracy is determined by Equation (3.1).

e. Punctuation accuracy

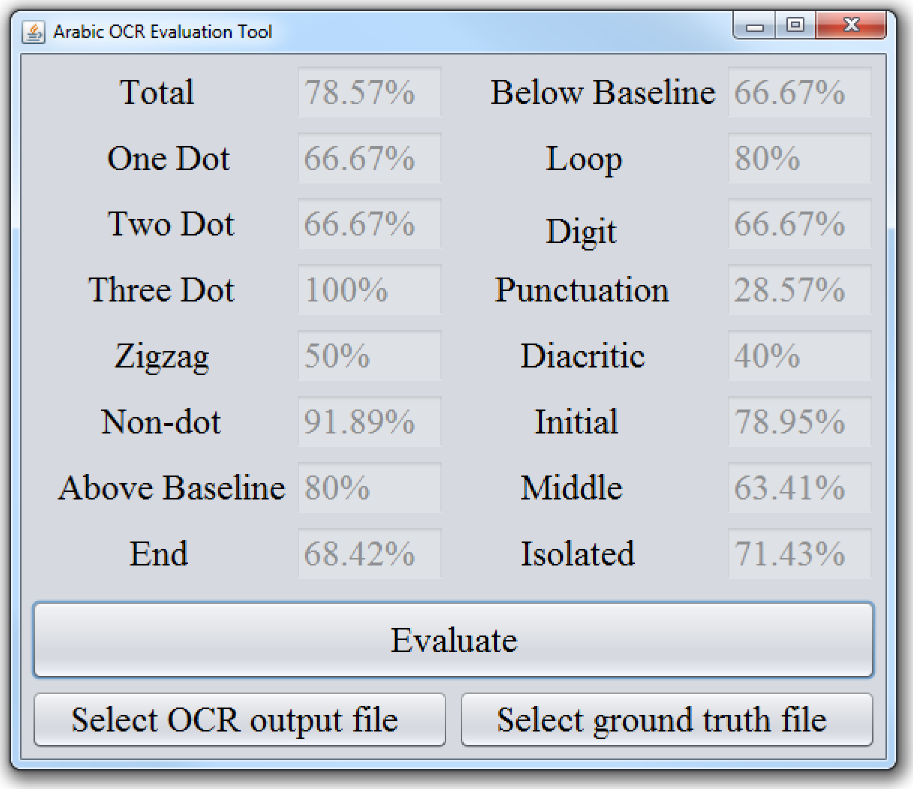
Several punctuation symbols resemble Arabic characters. For example, the full-stop punctuation (.) is quite identical to the Arabic Zero number (٠). This illustration confirms that the punctuation group has a vital effect on Arabic OCR accuracy. Punctuation accuracy is determined by Equation (3.1).

3.3 An Automated Arabic OCR Evaluation Tool

The new Arabic OCR evaluation tool that was developed for this study, which is programmed in Java, allows for comparison of an OCR-generated text file with a ground truth text file. The difference between the two text files is computed in terms of the minimum cost of converting the OCR output text to the ground truth. For this purpose, Levenshtein edit distance algorithm in (Levenshtein, 1966) has been adopted.

By using the software tool, an evaluator can quantitatively measure the performance of Arabic OCR systems according to the proposed Arabic accuracy metrics, mentioned in section 3.2. A Graphical User Interface (GUI) has been implemented to facilitate evaluators to select the OCR output text file and the ground truth text file, as demonstrated in Figure 3.3. Also, as can be seen in Figure 3.3, after clicking on the *Evaluate* button, the GUI displays statistics of all the accuracy metrics discussed. The Arabic OCR evaluation tool is provided freely as open-source software at the following GitHub address: https://github.com/NLPBangor/OCREvaluationTool.

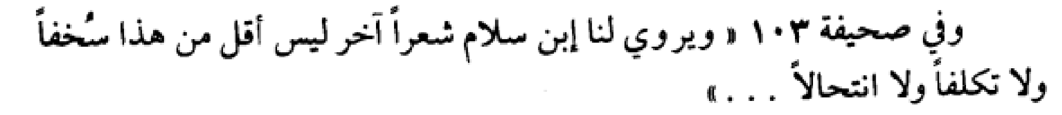
In order to illustrate how the tool works, an example is provided here. Figure 3.4 shows the text image that contains Arabic text with digits, diacritics and punctuation marks.


Arabic OCR Evaluation Tool

Total	78.57%	Below Baseline	66.67%
One Dot	66.67%	Loop	80%
Two Dot	66.67%	Digit	66.67%
Three Dot	100%	Punctuation	28.57%
Zigzag	50%	Diacritic	40%
Non-dot	91.89%	Initial	78.95%
Above Baseline	80%	Middle	63.41%
End	68.42%	Isolated	71.43%

Figure 3.3. GUI of the Arabic OCR evaluation tool

The open source OCR Tesseract engine[1], which supports Arabic, was run to convert the Arabic text image to an editable text. Figure 3.5 displays the corresponding OCR-generated text. To evaluate the accuracy of the utilised OCR application, the tool was run on the OCR output text and the ground truth text. The result of the evaluation is presented in Figure. 3.3.



وفي صحيفة ١٠٣ « ويروي لنا إبن سلام شعراً آخر ليس أقل من هذا سُخفاً
ولا تكلفاً ولا انتحالاً . . . »

Figure 3.4. A sample Arabic text image

وفي صسيقة ة١٠((وتروي لنا إس سلام شعرأ آخر ليس أقل من هذا سعخنأ
ولاتكلقأ ولا اتمرلأ ؟؟

Figure 3.5. OCR output text for image in Fig. 3.4

This example illustrates the idea that evaluation of Arabic OCR systems according to different accuracy measures is required. In other words, it can be seen from the results in Fig. 3.3 that the total character accuracy is about 78%. However, the accuracy results for individual character classes are significantly different. In particular, the accuracy of the one dot characters group and the two dots characters group are around 67%, whereas, the accuracy of the three dot character class is 100%. Moreover, the observed difference between the accuracy of the zigzag characters group and the loop characters group is significant, 50% and 80% respectively. Another important finding is that the accuracy of the above baseline characters class (80%) is very different to the accuracy of below baseline characters (67%). Also, it can clearly be seen that the punctuation class has the lowest accuracy rate.

Based on the findings above, it is possible to state that measuring Arabic OCR performance in terms of the accuracy metrics, which are implemented in the Arabic OCR evaluation tool, can help researchers to automatically assess how Arabic OCR

[1] https://github.com/tesseract-ocr

systems are overcoming the challenges of Arabic text and it contributes to eliminate human error, improve speed and precision, and reduce repetition in evaluating Arabic OCR systems.

3.4 Arabic OCR Systems

Some research efforts on printed Arabic text recognition have been reported in the literature. A multi-font cursive Arabic OCR system is reported by Khorsheed and Clocksin (2000). This system is a free-segmentation based approach in which each word in a used lexicon was transformed into a normalised polar image. The system extracts Fourier transform coefficients from a normalised polar image. Each word is represented by a template that contains a Fourier transform coefficient. The recognition is performed using a normalized Euclidean distance that measures the distance between the templates and tested word images. This system was evaluated using a sample of 1700 Arabic words and achieved an accuracy of 90%. To improve the accuracy rate, Khorsheed (2004) presents a new system based on Hidden Markov Models (HMM) in which each word is represented by a single HMM. The system has achieved an accuracy rate of 97% which is a higher accuracy rate compared to the template-based system.

A printed Arabic OCR system based on HMMs was developed by Al-Muhtaseb et al. (2008). They used overlapping and non-overlapping hierarchical windows to extract 16 features. The system was tested on eight Arabic fonts and achieved accuracy rates between 98.1% and 99.9%. In another effort, Khorsheed (2007) introduced a free-segmentation OCR system for recognising printed Arabic text. The system extracts statistical features to be fed to the HMM. The system was tested on six different Arabic fonts and the highest accuracy achieved was 92.4%. In a later work, Khorsheed (2015) proposed another free-segmentation OCR system using HMMs with run-length encoding as a feature extraction method. The system achieved a higher accuracy rate compared to the pervious developed system. Recently, Ko et al. (2018) proposed an OCR system for recognising printed Arabic text using a deep-learning approach. The system achieved an accuracy rate of 99.9%.

However, it is a challenge to compare the performance accuracy that are obtained on different datasets. Only a handful of accessible OCR systems claim that they are capable of recognising Arabic script. Therefore, our evaluation study is limited to the four most well-known Arabic OCR systems, namely, Automatic Reader 11.2 produced by the Sakhr Software Company; FineReader 12 produced by the ABBYY Company; Clever Page produced by RDI (Research & Development International); and Tesseract produced originally by Hewlett Packard (HP). In the following subsections, the four Arabic OCR engines are briefly discussed.

Automatic Reader 11.2 software

Automatic Reader is a commercial product first developed by the Sakhr Software Company in 1982 for text recognition of Arabic script. It supports Arabic language and several Arabic character-based languages, such as Arabic, Farsi and Urdu. Sakhr claims that Automatic Reader has been ranked as the best existing Arabic OCR software for high-quality text images by United States (US) government evaluators (Sakhr Software OCR, 2017). It supports multi-font type and multi-resolution images. However, font size 8 is not supported by the Automatic Reader OCR software (henceforth, referred to as Sahkr OCR).

FineReader 12 software

FineReader is produced commercially by a global company, called ABBYY, as advanced OCR software. The performance of FineReader has been enhanced by ABBYY for many years. FineReader 12 supports 190 languages including Arabic script using dictionary support (Abbyy OCR, 2017). It supports multi-font types, multi-size and multi-resolution images. Henceforth, FineReader will be referred to as ABBYY OCR.

Clever Page software

Clever Page originally began as a PhD research study by El-Mahallawy (2008). Since 2008, Clever Page has been designed and developed as an Omni font-written Arabic OCR engine by Research & Development International (RDI). Clever Page is a HMM based OCR system. El-Mahallawy (2008) designed a new feature extraction method based on autonomously normalized horizontal differentials. These features are used with HMMs to model Arabic characters. He reported a word error rate of 0.77% which is equivalent to an accuracy rate of 99.3%. This very high performance rate is due to the use of a noise-free dataset and using high quality Arabic text images. It supports multi-font types and multi-size text images. However, it is worth mentioning that the

Clever Page OCR software only works on pages with 300 dots per inch (dpi). Henceforth, Clever Page will be referred to as RDI OCR.

Tesseract software

Tesseract is an OCR engine designed at Hewlett Packard (HP) between 1984 and 1994. Since late 2005, it has been maintained by Google and released as open source OCR software. Tesseract OCR processes in different steps. In the first step, it converts an image to a binary image using the Otsu thresholding method (Otus, 1979). In the next step, it uses a page layout analysis method that has been developed by HP in order to extract the text blocks from an image (Smith, 2007). Then, it segments a text block into lines and words using definite spaces and fuzzy spaces (Smith, 2007). Tesseract extracts the outlines of the segmented words to obtain the polygonal approximation as features (Smith et al., 2009) . For the recognition task, it uses adaptive and static classifiers with the use of a dictionary lookup (Smith, 2007). It supports various languages. However, Arabic support has only been added recently (Sabbour and Shafait, 2013). Tesseract is the only Arabic OCR software that is freely available. It supports multi-font types, and multi-size and multi-resolution images.

3.5 Experimental Protocol

Very few Arabic datasets are freely available to researchers. The most widely used dataset for evaluating Arabic OCR approaches is the Arabic Printed Text Image (APTI). This public dataset was developed by Slimane et al. (2009). However, APTI is a word-level dataset where each text image contains only one Arabic word. Another Arabic dataset is ALPH-REGIM which is provided by Ben Moussa et al. (2010). This dataset contains about 5000 printed and handwritten Arabic text images. Compared to the APTI dataset, ALPH-REGIM is a paragraph-based text image dataset. However, the text images are only available in one font size and at one resolution level. A recent printed Arabic dataset is created by Doush et al. (2018). However, this dataset comprises text images of only two Arabic font types. Thus, the KAFD dataset, developed by Luqman et al. (2014), is used for evaluating the performance of the four Arabic OCR systems.

The KAFD dataset is freely available and is a page-level dataset where each text image consists of a text. It has Arabic printed text images and corresponding ground-truth text files. These text images are available at 100 dpi, 200 dpi and 300 dpi. Furthermore, it comprises text images of 40 Arabic font types, 10 pitch sizes and four styles, with resolutions of 100 dpi, 200 dpi, 300 dpi and 600 dpi.

For the current work, 10 different font types of the text image, in which the forms of the characters are of various types, are selected, namely, Andalus, Arabic Transparent, AdvertisingBold, Diwani Letter, DecoType Thuluth, Simplified Arabic, Tahoma, Traditional Arabic, DecoType Naskh and M Unicode Sara. These font types are selected based on the level of complexity of the writing style in printed Arabic script, ranging from simple fonts with no overlaps and ligatures, such as AdvertisingBold, to more complex fonts with overlaps, such as Diwani Letter. For each font type, 8, 12, 18 and 24 pitch sizes have been used in order to highlight the effectiveness of Arabic OCR systems on specific pitch sizes. Moreover, to enable the evaluation of the performance of Arabic OCR systems in terms of font style, normal and italic font styles are used. To study the influence of the page resolution level on Arabic OCR system performance, all text images in the above set have been randomly selected at 100 dpi, 200 dpi and 300 dpi. This resulted in 80 text images for each resolution. Thus, this experiment is performed on 240 text images in TIFF format.

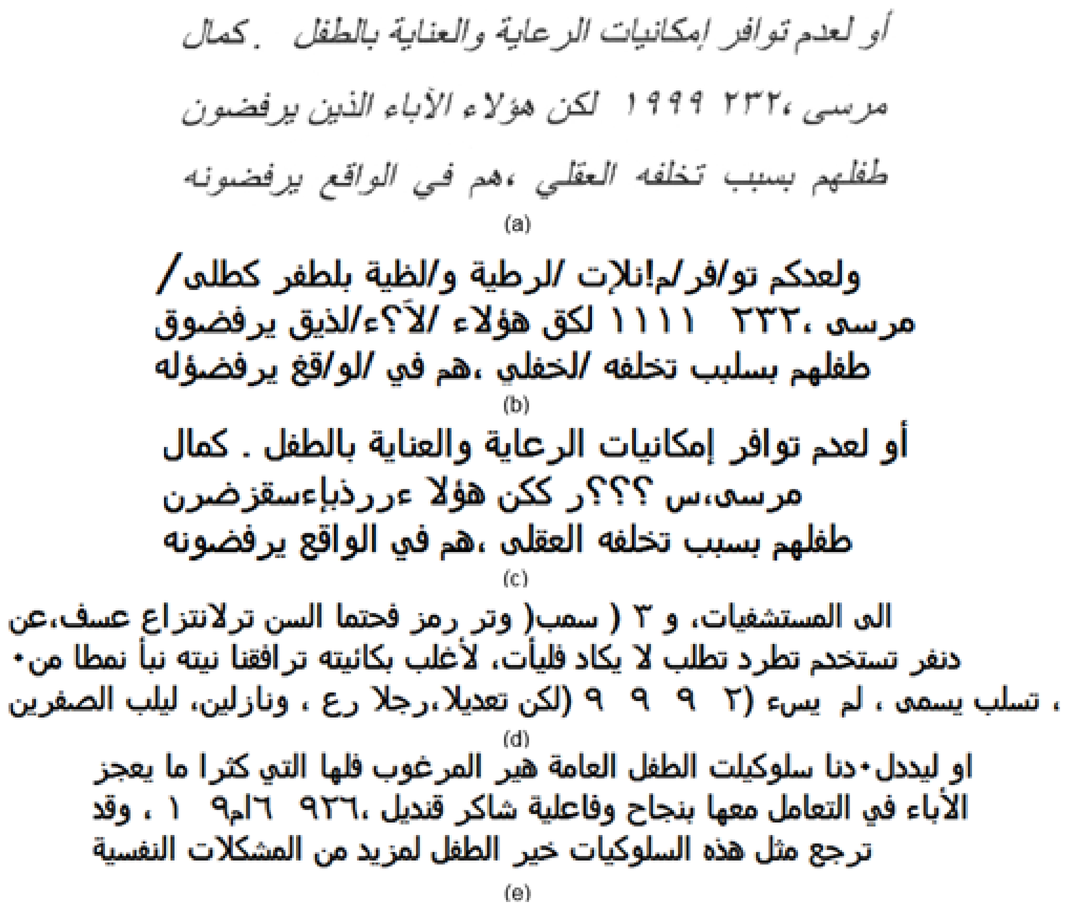
The experiment was conducted on the four Arabic OCR systems, as previously discussed, namely, Sakhr, ABBYY, RDI and Tesseract OCR systems. Sakhr and ABBYY OCR systems were kindly provided to the authors by their developers, whereas the RDI OCR system output was obtained by sending the dataset to the system’s developer to apply the system to the test images.

To statistically assess the performance of each Arabic OCR system, the performance metrics described in section 3.2 were used. In particular, the output text files of each Arabic OCR system were utilised to compute the quantities of each performance metric, corresponding to the ground-truth text files for the dataset. To eliminate human error, improve speed and precision, and reduce repetition, the automated open access tool for evaluating Arabic OCR systems, descripted in section 3.3 was utilised to obtain the statistical data. A sample of a text image from the dataset and the corresponding Arabic OCR output are visually illustrated in Figure 3.6.

3.6 Experimental Results and Discussion

The experimental results, obtained from the evaluation experiment discussed in the previous section, are presented in this section to analyse the effectiveness of the evaluated Arabic OCR systems in printed Arabic text recognition.

The overall character accuracy scores for the four evaluated Arabic OCR systems over different resolutions are presented in Figure 3.7. It is apparent that the Arabic OCR systems are affected by the resolution of the text images. In particular, the results shows a gradual increase in the overall character accuracy of all Arabic OCR systems when increasing the resolution of text images from 100 dpi to 300 dpi. In addition, a clear upward trend is apparent in the overall character accuracy of Sakhr OCR when the resolution of text images increased from 100 dpi to 200 dpi. Interestingly, high scores of character accuracy at 100 dpi, 200 dpi and 300 dpi were obtained by the ABBYY OCR system. Crucially, this system among the four Arabic OCR systems has the highest character accuracy at 100 dpi, 200 dpi and 300 dpi of 46%, 60% and 62%, respectively.



أو لعدم توافر إمكانيات الرعاية والعناية بالطفل . كمال
مرسى ،٢٣٢ ١٩٩٩ لكن هؤلاء الآباء الذين يرفضون
طفلهم بسبب تخلفه العقلي ،هم في الواقع يرفضونه

(a)

ولعدكم تو/فر/م!نلاِت /الرطية و/الظية بلطفر كطلى/
مرسى ،٢٣٢ ١١١١ لكق هؤلاء /لآ؟ء/لذيق يرفضوق
طفلهم بسلبب تخلفه /لخفلي ،هم في /لو/قغ يرفضؤله

(b)

أو لعدم توافر إمكانيات الرعاية والعناية بالطفل . كمال
مرسى،س ؟؟؟ر ككن هؤلا ءررذبإءسقزضرن
طفلهم بسبب تخلفه العقلى ،هم في الواقع يرفضونه

(c)

الى المستشفيات، و ٣ (سمب(وتر رمز فحتما السن ترلانتزاع عسف،عن
دنفر تستخدم تطرد تطلب لا يكاد فليأت، لأغلب بكائيته ترافقنا نيته نبأ نمطا من•
، تسلب يسمى ، لم يسء (٢ ٩ ٩ ٩ (لكن تعديلا،رجلا رع ، ونازلين، ليلب الصفرين

(d)

او ليددل•دنا سلوكيلت الطفل العامة هير المرغوب فلها التي كثرا ما يعجز
الأباء في التعامل معها بنجاح وفاعلية شاكر قنديل ،٩٢٦ ٦١م٩ ١ ، وقد
ترجع مثل هذه السلوكيات خير الطفل لمزيد من المشكلات النفسية

(e)

Figure 3.6. (a) A text image from the KAFD dataset; (b) output of the Sakhr OCR system; (c) output of the ABBYY OCR system; (d) output of the RDI OCR system; and (e) output of the Tesseract OCR system

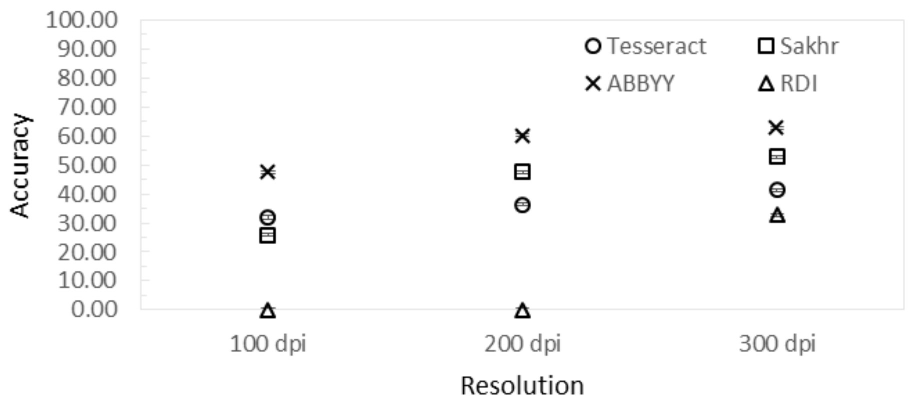


Figure 3.7. Overall accuracy vs. resolution results from the OCR system evaluation

The accuracy of Arabic OCR systems in recognising Arabic characters based on their position in a word are provided in Figure 3.8. As has been mentioned, the connectivity feature of Arabic script is an obstacle to Arabic text recognition. This is supported by the experiment data which indicate that all of the evaluated Arabic OCR systems have higher accuracy in recognising isolated characters when they are not connected with other characters in a word. On the other hand, the performance of the Arabic OCR systems decreased in recognising initial, middle and end characters, as shown in Figure 3.8. Compared to the Arabic OCR systems' accuracy in recognising isolated characters, the low recognition accuracy of initial, middle and end characters by the evaluated systems is possibly due to the segmentation algorithms implemented by these systems, as the segmentation process is not required for recognising isolated characters. However, a new methodology is needed to evaluate the segmentation stage in Arabic OCR systems to gain a better understanding of these results.

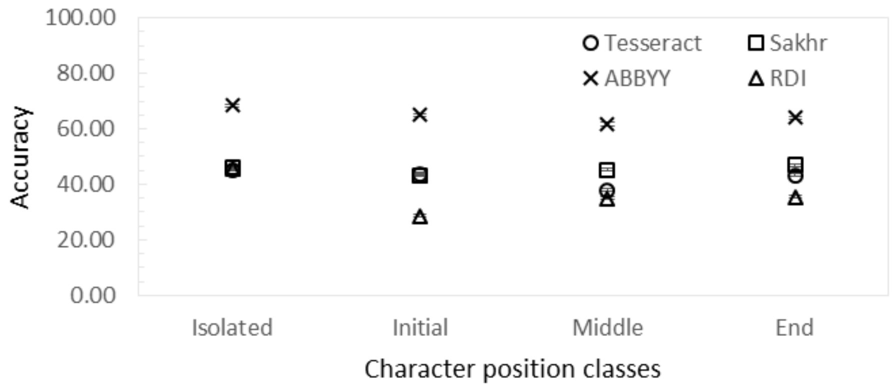


Figure 3.8. Character accuracy in terms of character position results from the OCR system evaluation

The results of the recognition accuracy performance of the Arabic OCR systems in terms of one dot, two dot, three dot and no-dot characters are illustrated in Figure 3.9. It is obvious that the recognition accuracy rates of no-dot characters are significantly better for all evaluated Arabic OCR systems, compared to one, two and three dot characters. This confirms that one of the challenges in Arabic OCR development is the presence of dots in Arabic script, as discussed previously. It has been hypothesised that a thinning algorithm, as a pre-processing technique, has an influence on the recognition accuracy of dot characters (Hosseini, 1997). In particular, some thinning algorithms may remove the dots of Arabic characters which results in misrecognising these characters. Therefore, these results are likely to be related to thinning algorithms used in the pre-processing stage of the Arabic OCR system.

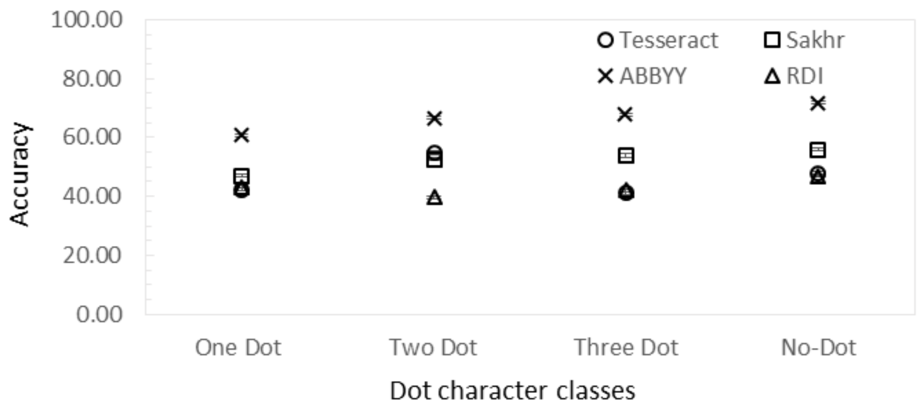


Figure 3.9. Dot character accuracy vs. no-dot character accuracy results from the OCR system evaluation

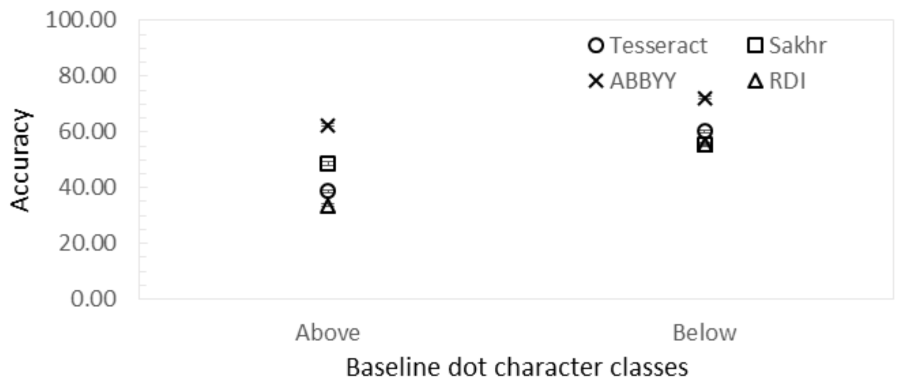


Figure 3.10. Dot character accuracy based on baseline results from the OCR system evaluation

The results of analysing the performance of the evaluated Arabic OCR systems on characters that have a dot above or below the baseline are presented in Figure 3.10. These results highlight that Arabic OCR systems perform much better in identifying characters with a dot below the baseline than characters with a dot above the baseline. The reasons for these results are not entirely clear. However, one technique used in the pre-processing stage for developing Arabic OCR systems is page decomposition which separates the lines of a text block in a text image. To be specific, some researchers emphasise that considerable attention must be paid in the page decomposition of Arabic text images to ensure that dots placed above or below a line of text are not separated (Al-Badr and Mahmoud, 1995; Sami El-Dabi et al., 1990). Thus, a possible reason for this finding may be due to the page decomposition process.

Figure 3.11 compares the recognition accuracy for zigzag-shaped characters and loop-shaped characters by the evaluated Arabic OCR systems. It is apparent from the results that the performance rates of Arabic OCR systems in accurately recognising loop-shaped characters are higher than in recognising zigzag-shaped characters. It can thus be assumed that recognising characters with a zigzag shape is more challenging than recognising characters that have a loop shape.

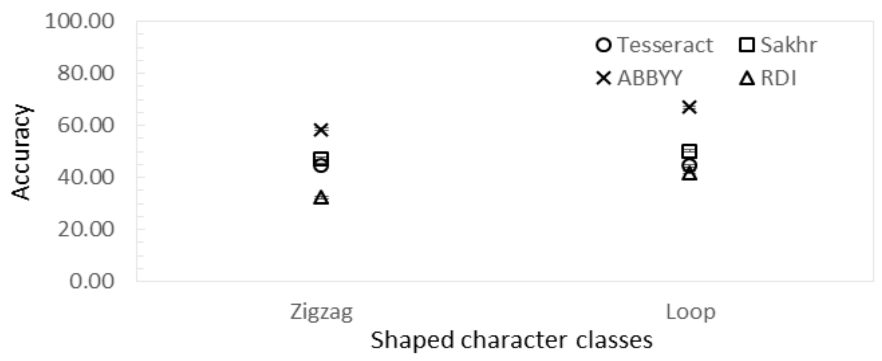


Figure 3.11. Character accuracy based on shaped character results from the OCR system evaluation

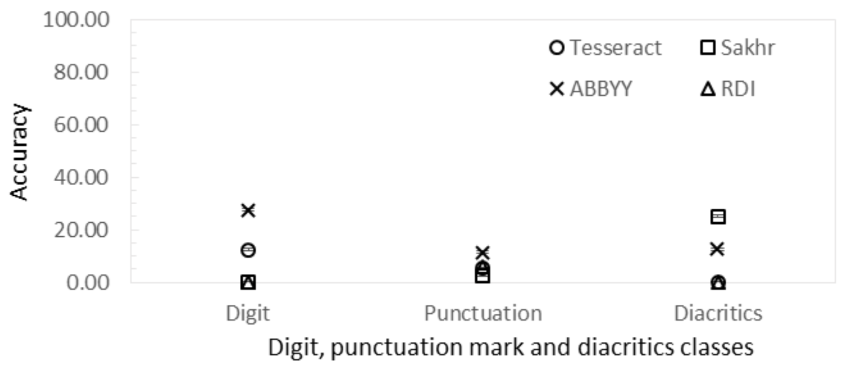


Figure 3.12. Digit, punctuation and diacritics accuracy results from the OCR system evaluation

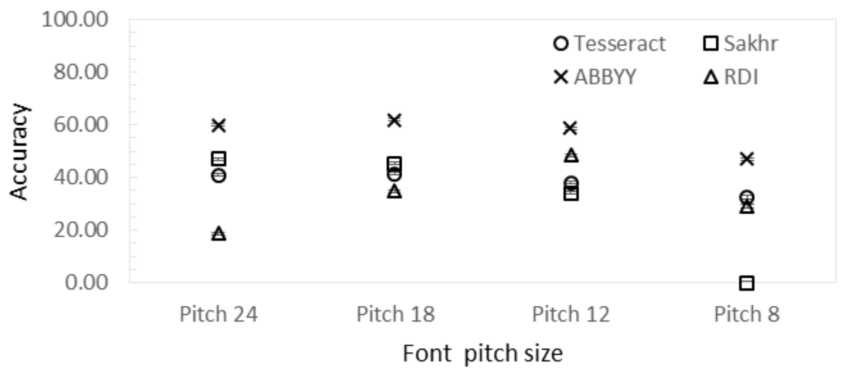


Figure 3.13. Arabic OCR accuracy vs. font pitch size results from the OCR system evaluation.

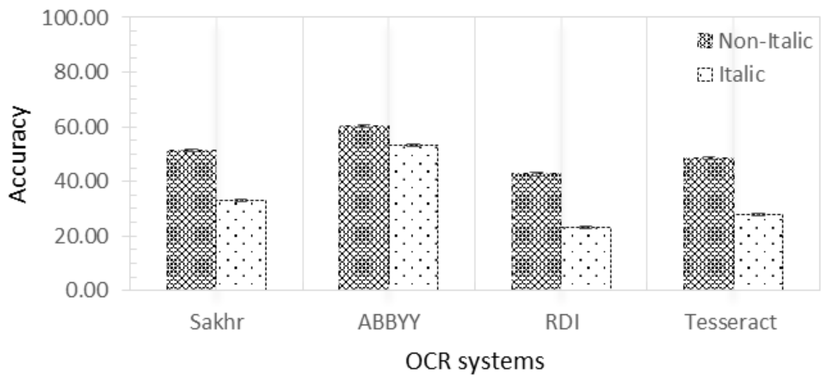


Figure 3.14. Italic and non-italic font accuracy comparison results from the OCR system evaluation

Figure 3.12 presents the recognition accuracy of the evaluated Arabic OCR systems in terms of the digits, punctuation and diacritics groups. Surprisingly, Tesseract and RDI OCR systems were not able to recognise any diacritical marks. However, one possible reason is that these OCR systems were not trained to recognise diacritics. In addition, Figure 3.12 indicates that the Sakhr OCR system was unable to recognise Arabic digits (Hindu digits). As this result was not expected, we undertook further investigation to find an explanation. After manual inspection of the output of the Sakhr OCR system, we discovered that the system recognises Arabic digits as English digits. In other words, the system replaced the Arabic digits with English digits when producing the output. Improved accuracy rates for the Sakhr OCR system could be achieved if the system were to overcome this problem.

The respective accuracy of the Arabic OCR systems in relation to different font pitch sizes is plotted in Figure 3.13. It can be seen that a drop in character recognition accuracy occurs when reducing the font pitch size. It can thus be concluded that the evaluated systems struggle when the font pitch size is too small, such as a font pitch of 8. Another important finding from this result is that the RDI OCR system performs better for a font pitch of 12 than for a font pitch of 24. The reason for this finding is unclear.

The recognition performance analysis of the evaluated Arabic OCR systems on fonts with an italic style compared to those with a non-italic font style is shown in Figure 3.14. Our experimental results show that, for all evaluated systems, the recognition accuracy rates for a non-italic font style are higher than for an italic font style. The low recognition accuracy percentages for an italic font style are likely to be related to the fact that the italic font style is a cursive font.

Table 3.4 reports on the accuracy of the evaluated Arabic OCR systems in terms of font types. As expected, the low accuracy rates for all evaluated systems in recognising the Diwani Letter font are due to the complexity of this font which contains overlaps.

In summary, most of the performance accuracy rates of the evaluated Arabic OCR systems are below 75%, indicating the continuing need for improvements in the recognition of printed Arabic script. Moreover, the correlation between the performance accuracy of Arabic OCR systems and the features of Arabic script have been highlighted. In particular, the experimental analysis indicates that the characteristics of printed Arabic script, such as the connectivity, and the presence of dots and zigzag shapes, all contribute to the challenge for Arabic OCR systems. Overall, the results indicate that recognition of Arabic script remains an open research problem.

Font Type	OCR System			
	Sakhr	*ABBYY*	*RDI*	*Tesseract*
Traditional Arabic	48.5%	67.6%	51.9%	47.1%
Tahoma	40.5%	69.9%	26.4%	38.4%
Simplified Arabic	52.9%	67.7%	44.9%	46.7%
M Unicode Sara	36.1%	59.4%	25.9%	33.7%
Diwani Letter	18.1%	18.4%	18.1%	23.3%
DecoType Thuluth	36.1%	37.7%	24.2%	32.5%
DecoType Naskh	48.8%	50.2%	41.6%	40.9%
Arabic Transparent	51.5%	75.1%	46.1%	48.6%
Andalus	28.1%	37.5%	21.7%	25.3%
AdvertisingBold	57.3%	70.2%	27.2%	39.4%

Table 3.4. Arabic OCR performance accuracy according to 10 different font types

3.7 Summary and Conclusion

This chapter describes an automated tool for performance evaluation based on different objective accuracy metrics with respect to the challenges of Arabic text. This tool has been specifically developed to assist Arabic OCR researchers to automatically calculate the accuracy of different Arabic OCR systems and to assess how Arabic OCR systems are copying with the challenges of Arabic text. Furthermore, by using this automated tool, an OCR evaluator can quickly process an enormous number of testing experiments. It also provides an experimental protocol that enables the effectiveness of different Arabic OCR systems to be compared.

In addition, this chapter describes an experiment to automatically evaluate four well-known Arabic OCR systems: Sakhr, ABBYY, RDI and Tesseract. The evaluation experiment is conducted on a publicly available printed Arabic dataset comprising 240 text images with a variety of resolution levels, font types, font styles and font sizes. The experimental results show that all the evaluated Arabic OCR systems have low performance accuracy rates, below 75%, which means that the time which it would take to manually correct the OCR output would be a prohibitive.

Chapter 4: Thinning

4.1 Introduction

This work aims to develop a novel OCR for printed Arabic text. To achieve this aim, we implement a new Printed Arabic Optical Character Recognition (PATRION) system using five stages: thinning, feature extraction, character segmentation, classification and post-processing, as illustrated in Figure 4.1. In this chapter, we present the first stage for developing the PATRION system which is thinning as a pre-processing stage.

Thinning is one of the critical processes that can be applied to the input text images for facilitating the subsequent stages of the PATRION OCR development process. In particular, producing skeletons is required for the PATRION OCR development in which extracting features from the skeleton of Arabic text is essential. Thus, a new thinning algorithm for Arabic script has been implemented in order to fulfil the requirements of the development of the PATRION OCR system. Furthermore, several new performance metrics for evaluating thinning algorithms for Arabic text are presented, since there is a lack of quantitative performance measures of thinning techniques, as stated in the literature.

The work in this chapter has been published in the following paper:

- Alghamdi, M.A. and Teahan, W.J., 2017. A new thinning algorithm for Arabic script. *International Journal of Computer Science and Information Security*, *15*(1), p.204-211.

This chapter first describes an effective new thinning algorithm for printed Arabic script. Then, evaluation of thinning algorithms for Arabic will be discussed with introducing several new objective performance metrics for thinning algorithms. An evaluation experiment is conducted on the proposed algorithm with providing a comparison with a number of most common thinning algorithms. In the last section, a summary and conclusions will be provided.

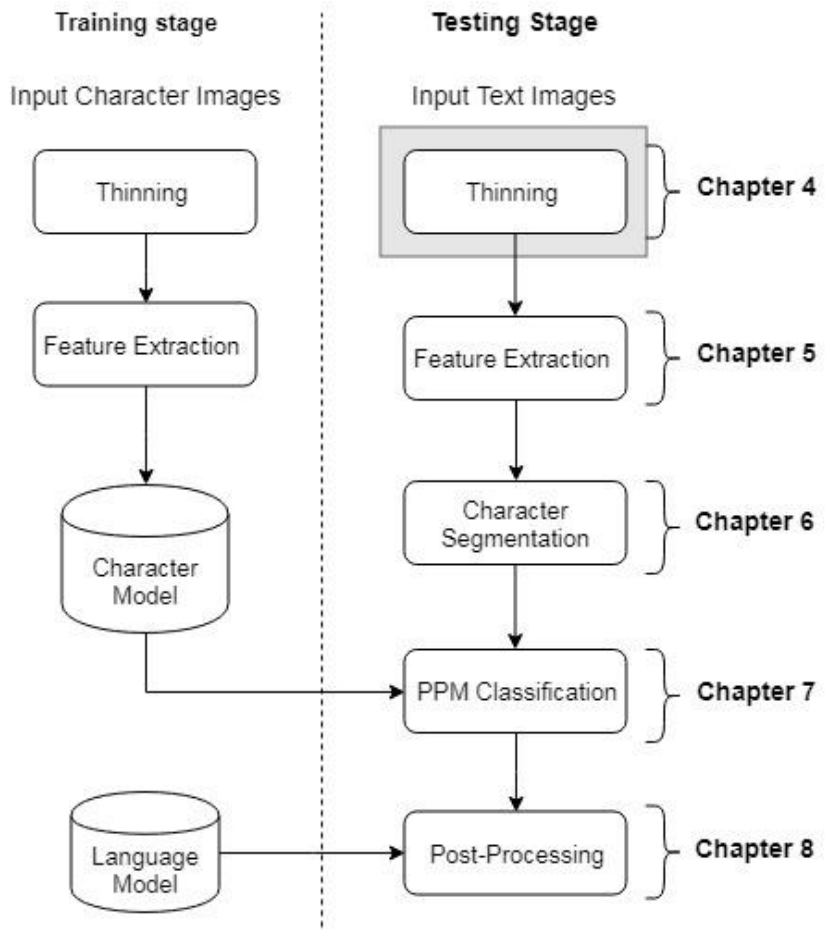


Figure 4.1: Block diagram of the PATRION system

Note: The shaded area shows the scope of the current chapter (the thinning stage).

4.2 An Effective Thinning Algorithm for Arabic Text

An effective thinning algorithm produced by Kocyigit (2012) has been used in this work. However, this algorithm was only investigated for English characters. The method aims to produce a skeleton with one width pixel by detecting the neighborhood connectivity of any given pixel. Each pixel is directly connected with eight neighboring pixels; the pixel is considered as "encased" if only all the black neighbor pixels are interconnected with each other. In other words, if each black neighbour pixel of a considered pixel is connected vertically or diagonally to at least one other black pixel, the given pixel will achieve a state of encasement. For example, if the center pixel (P) is being processed pixel, it is considered as "encased" in Figure 4.2: (a), (b) and (c). In contrast, (b), (c) and (d), the pixel (P) is not considered as "encased".

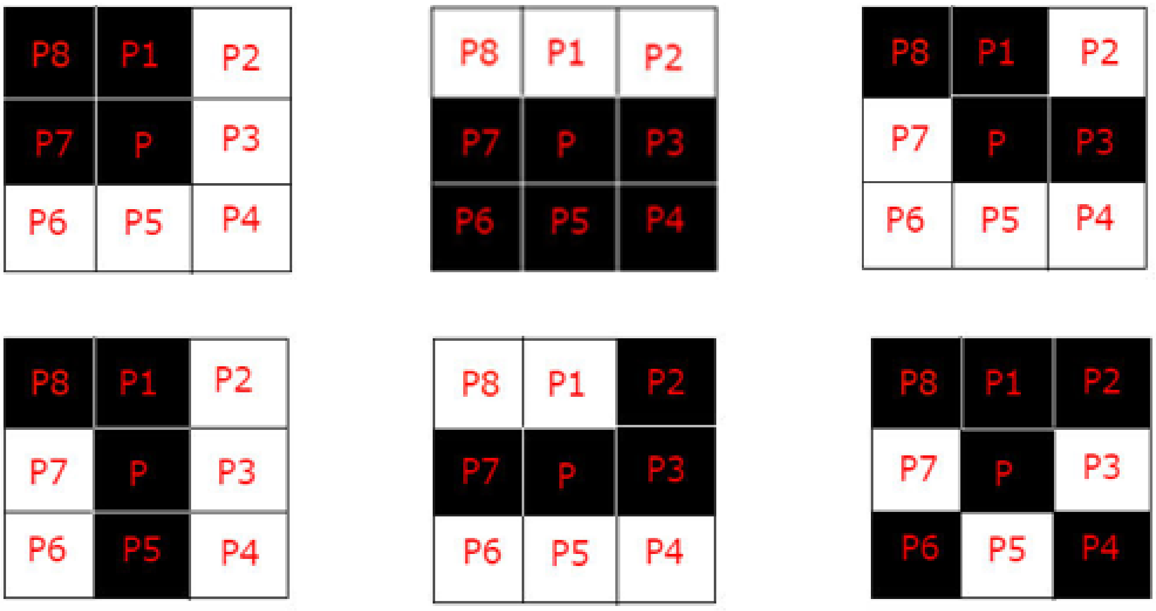


Figure 4.2: Examples of pixels (a), (b), (c) with an encasement and (d), (e), (f) without an encasement

Kocyigit algorithm utilizes a 3x3, 2D array of pixels which is obtained by thresholding the image. The descriptions of the rows and columns are equivalent to the pixel's coordinates and are set to either value "1" (black) or "0" (white) depending on the character. Deletion of a pixel will change the value of the pixel from 1 to 0.

Whenever any pixel is encased, the algorithm cleans up the defined pixel. The algorithm observes sequential and iterative principles in that whenever any pixel is removed, the entire algorithm restarts the iterative process, continues until a state of consistency is achieved where no further pixel is deleted.

A flowchart of Kocyigit algorithm is provided in Figure 4.3. Initially, the algorithm finds a black pixel. Then, delete the black pixel if it has an encasement.

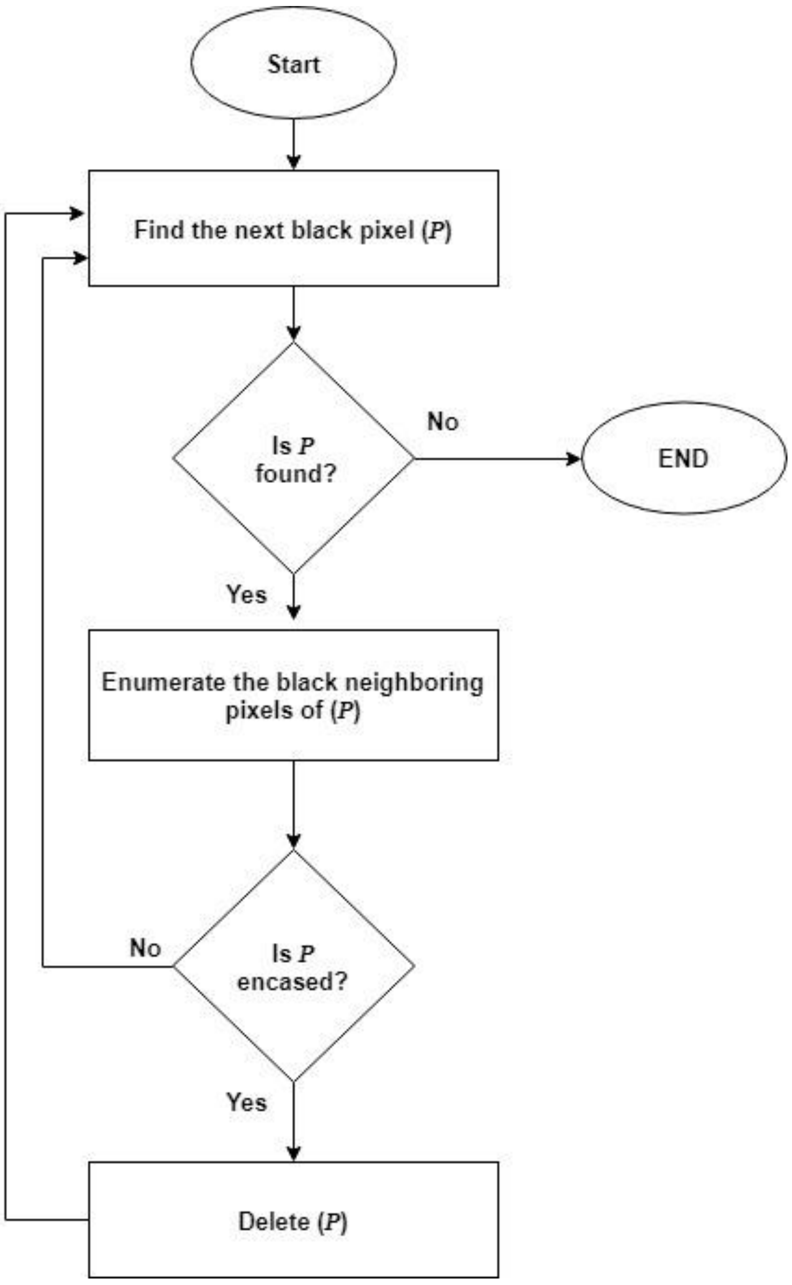

Start
Find the next black pixel (*P*)
Is *P* found?
No
END
Yes
Enumerate the black neighboring pixels of (*P*)
No
Is *P* encased?
Yes
Delete (*P*)

Figure 4.3: Flowchart of Kocyigit algorithm

4.3 Evaluating Thinning Algorithms for Arabic Text

In respect to performance evaluation of thinning algorithms, most of studies evaluate their proposed thinning algorithm in terms of execution time and compression ratio, such as in Naccache and Shinghal, (1984), Saudagar and Mohammed, (2016) and Zhang and Wang, (1988). Furthermore, some researchers evaluate thinning algorithms by only illustrating the output images of the algorithms, regardless of application requirements, such as Ali and Jumari, (2003), Ali, (2012), Abu-Ain et al. (2013) and Abu-Ain et al. (2014). However, from an OCR perspective, these metrics are not sufficient to provide us with insight into which algorithm performs better for OCR.

Zhou et al. (1995) provide other general metrics for assessing thinning algorithms, such as the thinness metric, which assesses the level to which a patter image is thinned, and the connectivity metric that measures the connectivity of the thinned pattern images.

As mentioned before Arabic script uses a cursive writing style and therefore, the connectivity measure is considered as a critical metric in order to evaluate the connectivity of the output skeleton of an Arabic text image. For instance, the study in Al-Shatnawi et al. (2014) utilizes the connectivity measure to assess the performance of different thinning algorithms for Arabic. The idea of this metric is to consider the number of connected components in the original image and in the output image of the thinning algorithm. Namely, if the number of components is equal in both images, then this is a sign that the thinning algorithm is preserving text connectivity. However, some thinning algorithms may remove some dots of Arabic characters and they may spilt the body of a character into several parts. Consequently, an issue will arise when adopting the connectivity metric to evaluate the connectivity of thinned Arabic text. In particular, when thinning algorithms remove the characters dots or make a disconnection through the letters forms, then a performance metric which is relying on the number of connected components, will not be reliable.

To illustrate this problem, Figure 4.4 shows that the number of component of the original image (a) is four and the number of components in the thinned image (b) is four. Thus, according to the connectivity metric, the algorithm will be assessed as preserving the text connectivity, where in fact it is not. This problem occurs owing to

Figure 4.4: Example of the problem of the connectivity metric: (a) an Arabic letter with four components and (b) a skeleton for (a) with four components

the removing of one dot of the letter and making a gap of the letter's shape. Therefore, it is difficult to obtain statistical evaluation results of connectivity preservation by relying on this measurement.

To overcome this problem, we propose new performance metrics for evaluating thinning algorithms for Arabic text in terms of connectivity and dots persevering. Moreover, other further objective performance metrics for evaluating thinning algorithms will be discussed below.

4.3.1 Connectivity preservation metric

Typically, an image for a pattern is constructed from a set of vertices (nodes) and edges (links) some of which might be connected or disconnected (Gao et al., 2010), see Figure 4.5 for illustration. For thinning algorithms to preserve text connectivity, they must not divide a pattern in an image into incorrect pieces. A sophisticated thinning algorithm therefore must not delete edges of an images in order to maintain the connectivity characteristic of Arabic script.

Basically, there are two possible cases that will affect preserving the connectivity when producing skeletons for Arabic script. The first case is deletion of edges. For example, considering Figure 4.6 (a), it is clear that the word in the original image has fourteen (14) vertices and seven (7) edges. In contrast, in the thinned image, which has been manually produced in order to illustrate the first case that affects preserving the connectivity, in Figure 4.6 (b) there are fourteen (14) vertices and six (6) edges. The lower number of edges is because of the deletion of one edge. The second case is insertion of a gap which results further insertion of edges and vertices. This can be seen in the case shown in Figure 4.7 where the thinned image, which was manually produced to illustrate the second case that affects preserving the connectivity, has sixteen (16) vertices and eight (8) edges, compared the original image in Figure 4.5 (a) which has fourteen (14) vertices and seven (7) edges.

Graph Edit Distance (GED) is a distortion measure on graphs that determines the differences between two pattern images (Neuhaus and Bunke, 2007). The differences between the two images are obtained with a number of edit operations that are required to convert one image into another image. The authors here will utilize the concept of GED to assess the effectiveness of a thinning algorithm in preserving the connectivity of Arabic text.

The edit operations are: insertion of edges, insertion of vertices and deletion of edges. In order to measure GED, the minimum number of edit operations (edit distance) needed to convert the thinned image into the original image will be considered. Then, accuracy of thinning algorithms in preserving Arabic connectivity is determined by:

$$\frac{g-e}{g} \tag{4.1}$$

where g is the number of edges and vertices in the original image, and e is the edit distance. The cost of each edit operation will be defined as 1. For example, if a thinning algorithm inserted a gap, it will result in adding two vertices and delete one edge. Thus, the edit distance cost will be three, as three operations are required to transform the thinned image into the original – two deletions of vertices and one deletion of an edge, each having a cost of 1.

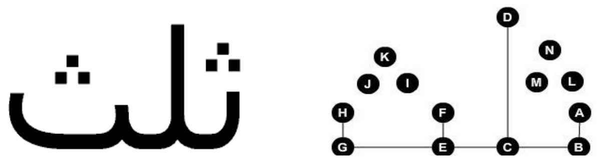


Figure 4.5. Using a graph to represent a text image: (a) text image and (b) graph for (a)

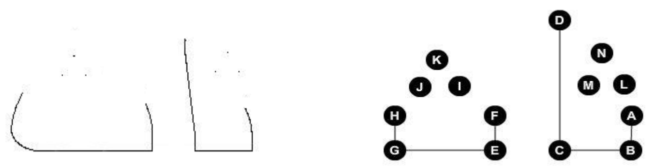


Figure 4.6. One thinning example for text image in Figure 4.5(a): (a) its skeleton and (b) a graph for (a).

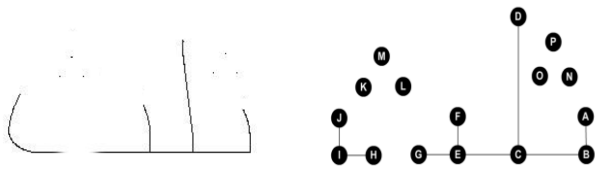


Figure 4.7. Another thinning example for text in Figure 4.5 (a): (a) its skeleton and (b) a graph for (a).

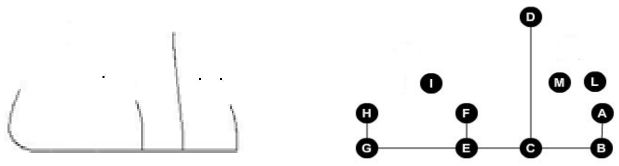


Figure 4.8. Example of dot perservation for thinning. (a) a skeleton and (b) a graph for (a)

4.3.2 Dot preservation metric

In order for thinning algorithms to preserve dots of Arabic script, they must not remove isolated vertices which present the dots of Arabic text (see Figure 4.8 which was manually produced for illustration). Thus, to statistically measure the effectiveness of thinning algorithms in preserving Arabic dots, the following equation is utilized:

$$\frac{v-e}{v} \tag{4.2}$$

where v is the number of isolated vertices in the original image, and e is the edit distance. The cost for each edit operation will be defined to be 1.

4.3.3 Topology preservation metric

The topology preservation metric is the performance measurement utilized to assess the degree of thinning algorithm in preserving the visual information of the original image (Chatbri and Kameyama, 2014; Jang and Chin, 1992; Huang et al., 2003; Goyal and Dutta, 2016a; Goyal and Dutta, 2016b). As reported by the researchers in Al-Shatnawi et al. (2014) and Al-shatnawi and Omar, (2014), producing spurious tails is considered a common problem of thinning algorithm for text images. Spurious tails can change the shape of a pattern, thereby affecting the accuracy of OCR output.

Accordingly, it is essential to evaluate the effectiveness of thinning algorithms in terms of producing spurious tails.

The accuracy of thinning algorithms in preserving topology is determined by Chatbri and Kameyama, (2014) :

$$1-\left[\frac{1}{2}-\frac{o}{c}\right] \tag{4.3}$$

where *o* is the number of object pixels in thinned image and *c* is the number of counter pixels in the thinned image. Note that in order to state that a thinning algorithm maintains visual information of the original image, the algorithm needs to have a preserving topology value close to 1.

4.3.4 Thinning rate (unit pixel width)

It is critical to evaluate the thinning algorithms in term of thinness of the skeleton, since one of the main principles of thinning algorithms is to ensure the peeling is as thin as possible. The thinness of the skeleton can be determined by computing the thinning rate as the following (Tarábek, 2008):

$$1-\frac{tc}{tr} \tag{4.4}$$

where *tc* and *tr* refer to the number of triangles in the thinned image and in the original image, respectively. A thinning rate of 1 indicates that the skeleton is thinned to one-pixel wide, whereas a 0 value indicates that the skeleton is not thinned.

4.4 Experimental Results and Discussion

We have used performance metrics described in the previous section for evaluating the Kocyigit thinning algorithm. For comparison purposes, we also produce results for two other commonly used thinning algorithms for Arabic script, namely the Zhan-Suen (Zhang and Suen, 1984) thinning algorithm and the Hilditch algorithm (Hilditch, 1969). Note that all the three thinning algorithms were implemented in the Java programing language. The performance evaluation will provide us with insight of which algorithm performs better for Arabic script. In particular, a good thinning algorithm for Arabic should have a high performance accuracy in preserving connectivity and dots with a value of preserving topology and thinning rate both close to 1.

Three datasets were used in this experiment. Specifically, the first dataset (Luqman et al., 2014) is the printed Arabic word dataset, which consists of 50 word images, differing in fonts, sizes and styles. Also, two further sets of images were created for this evaluation experiment: images of Arabic characters and images of Arabic digits images.

In this experiment, all images from the three datasets were sent to each thinning algorithm to obtain thinned images. Then, the output images of each thinning algorithm were utilized to compute the quantities of each performance metric, corresponding to the original images for the datasets. Note that the performance metrics are implemented using the Matlab programming language. Samples of the evaluation experiment from the three datasets are visually illustrated in Table I. Table II presents the experimental data on the visual samples shown in Table 4.1.

As discussed previously, connectivity preservation is one of the most essential features expected of thinning algorithms. From the data presented in Table 4.1, it is apparent that the connectivity characteristic of Arabic script is almost maintained by the Kocyigit algorithm. In particular, the Zhan-Suen algorithm and the Hilditch algorithm failed in preserving the connectivity of sample 1 and sample 5, whereas the connectivity is preserved by the new algorithm in both samples.

There were major differences in the preservation of the dots of Arabic script between the three thinning algorithms. However, the Kocyigit algorithm preserves the dots of Arabic script on all samples images that contain dots, as illustrated in Table 4.1.

Furthermore, it is instructive to note that the Zhan-Suen algorithm and the Hilditch algorithm were not able to preserve the loop shape of the sample 5 image, as shown in Table 4.1 and Table 4.2, whereas it has been preserved by the Kocyigit algorithm.

The average scores for each metric for the three thinning algorithms are presented in Table 4.3. The first column compares the three thinning algorithms in terms of the connectivity preservation which shows that the Kocyigit algorithm obtained the highest accuracy (94.6%) of connectivity preservation among the other two algorithms.

The second column of Table 4.3 compares the thinning algorithms in terms of preservation of the dots of Arabic. The results show that there are significant differences between the performance accuracy for dot preservation by the Zhan-Suen algorithm, the Hilditch algorithm and the Kocyigit algorithm, 78.2 %, 85.2 % and 99.4 % respectively. Thus, it can be stated that the Kocyigit algorithm is the most effective at preserving the dots.

The third column of Table 4.3 lists the values for topology preservation. As mentioned previously, a value of topology preservation close to 1 indicates that the algorithm is successful in preserving the shape of the original images. The results in the third column of Table 4.3 show that the value of topology preservation is significantly better for the Kocyigit algorithm (0.9599), compared to the Zhan-Suen algorithm (0.9398) and the Hilditch algorithm (0.9411).

The results of the thinning rate analysis are presented in the fourth column in Table 4.3. From this column, it clear that the thinning rate value achieved by the Kocyigit algorithm (0.9887) is superior to the other two algorithms. This indicates that the skeletons produced by the Kocyigit algorithm is thinner than skeletons produced by the Zhan-Suen and the Hilditch algorithms.

In summary, the evaluation experiment results show that the Kocyigit algorithm is more effective at dealing with the challenges of thinning for Arabic script, namely, connectivity and dots preservation. Moreover, the experimental analysis indicates that the skeletons of Arabic text produced by the Kocyigit algorithm are better than those produced by the two other algorithms in terms of topology preservation and thinning rate. Also, the evaluation experiment of thinning algorithms by utilizing the proposed performance metrics provides a more in-depth analysis of which algorithm will perform better for Arabic OCR systems.

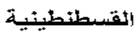
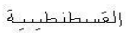
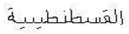
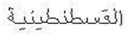
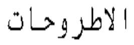
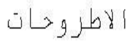
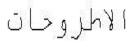
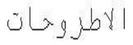
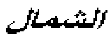
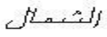
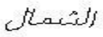
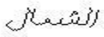
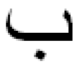
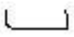
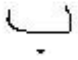
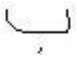
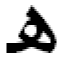
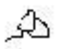
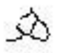
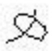
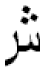
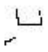
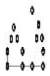
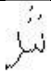
Sample No.	Original image	Zhan-Suen	Hilditch	Kocyigit
Sample 1				
Sample 2				
Sample 3				
Sample 4				
Sample 5				
Sample 6				
Sample 7				

Table 4.1: Visual samples used in the evaluation

Image	***Connectivity Preservation***			***Dots Preservation***			***Topology Preservation***			***Thinning rate***		
	Zhan-Suen	***Hilditch***	***Kocyigit***	***Zhan-Suen***	***Hilditch***	***Kocyigit***	***Zhan-Suen***	***Hilditch***	***Kocyigit***	***Zhan-Suen***	***Hilditch***	***Kocyigit***
Sample 1	0.909	0.909	1	0.9	0.9	1	0.9955	0.9288	0.9511	0.9473	0.9259	0.9766
Sample 2	1	0.958	1	1	1	1	0.9751	0.9729	0.9875	0.8711	0.8813	0.9796
Sample 3	0.941	0.941	1	1	1	1	0.9630	0.9983	0.9244	0.9695	0.9390	1
Sample 4	1	1	1	0	1	1	0.9247	0.9513	0.9513	0.9716	0.8865	0.9929
Sample 5	0.833	0.833	1	N/A	NIA	N/A	0.8614	0.8614	0.9045	1	0.9752	0.9886
Sample 6	0.5	0.5	0.833	0	0.333	1	0.9695	0.9434	0.9782	0.9389	0.9061	0.9859
Sample 7	0.8	0.8	1	N/A	N/A	N/A	0.9426	0.9590	0.9918	1	0.9606	0.9859

Table 4.2: comparing the three thinning algorithms on the samples

Algorithms	Connectivity Preservation	Dot Preservation	Topology Preservation	Thinning Rate
Zhan-Suen	0.844	0.782	0.9398	0.9552
Hilditch	0.837	0.852	0.9411	0.9266
Kocyigit	0.946	0.994	0.9599	0.9887

Table 4.3: summary of results comparing the thinning algorithms

4.5 Summary and Conclusion

An effective thinning algorithm for Arabic script is described. Furthermore, the chapter describes several objective performance metrics for assessing statistically the effectiveness of thinning algorithms including, two new metrics for assessing topology preservation and dots preservation. By utilizing these metrics, a more robust evaluation of the effectiveness of different thinning algorithms can be carried out. Consequently, these metric will simplify the choice of thinning algorithms for Arabic OCR developers.

An evaluation experiment was conducted to evaluate the performance of the Kocyigit algorithm against other two well established thinning algorithms. In all performance tests, the Kocyigit algorithm obtains the best results.

In the next chapter, the output of the thinning algorithm will be used for the next stage of the PATRION OCR system.

Chapter 5: Feature Extraction

5.1 Introduction

In the previous chapter, the skeletons for Arabic text images have been produced by using the new thinning algorithm. In the following stage of the developed PATRION system, which is the feature extraction stage (as shown in Figure 5.1), the features of the skeletons will be extracted. Feature extraction is the process by which certain attributes of interest within a text image are obtained to be utilized for further processing of recognizing Arabic script. Feature extraction is the crucial phase in the development of Arabic OCR, as a well-defined feature set can facilitate recognition and increase accuracy performance (Lawgali, 2015).

In order to recognize Arabic script, our new approach develops a feature extraction method that transforms the word into a set of features that are utilized to recognise the Arabic word. Since the input for the PATRION OCR system is a text image of a page of printed Arabic, several pre-processing steps are required to be applied prior to or within feature extraction. These steps are *baseline determination* and *word segmentation*.

The remainder of this chapter describes our approach to the feature extraction stage. It formally states the rationale behind using this approach. Then, it explains in detail the feature extraction method that has been used in this work with the required pre-processing steps, baseline determination and word segmentation.

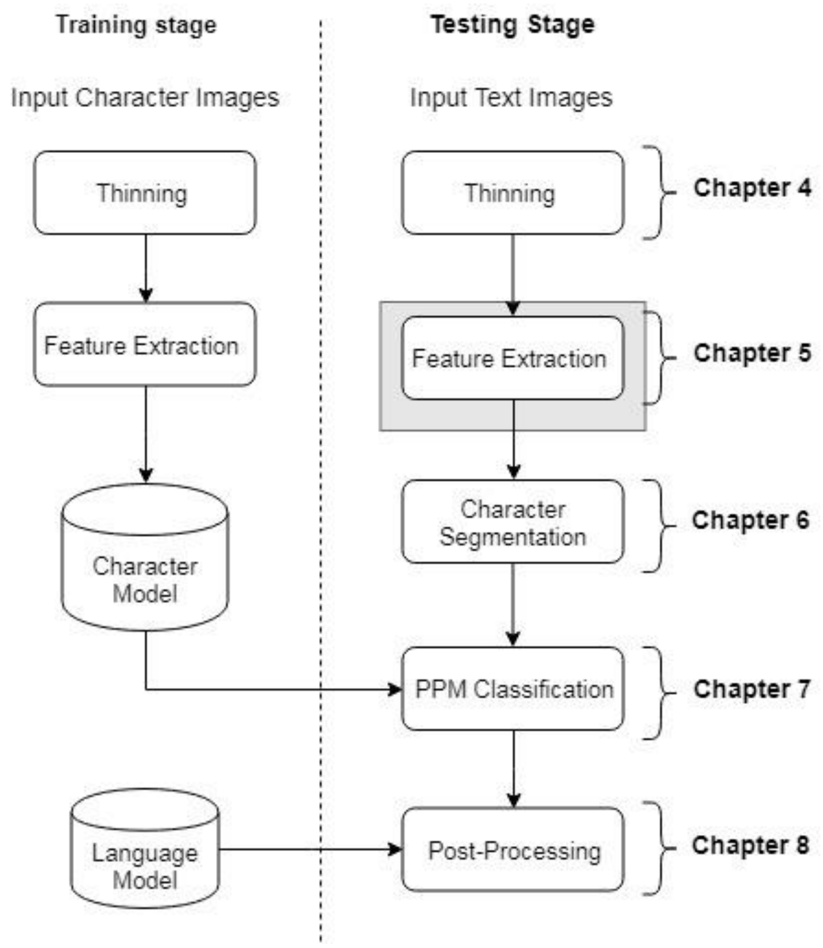


Figure 5.1: Block diagram of the PATRION OCR system
Note: The shaded area shows the scope of the current chapter 'Feature Extraction'.

5.2 Why This Approach?

The human reading task is a cognitive process nearly equivalent to OCR that involves detecting characters and classifying them from among enormous possibilities (Rashid, 2014). Humans can efficiently and accurately recognize text in different fonts and sizes. The accuracy of human recognition of text can be attributed to several cognitive processes, involving reading and writing strategies.

Generally, it is claimed that humans rely on their existing knowledge in order to accomplish a reading task (Pi et al., 2008). For instance, people can recognise identical words or characters based on the surrounding context. A good example for native Arabic readers is the text illustrated in Figure 5.2 which has a number of identical characters (non-dotted characters) that have different pronunciations but can be identified based on the context. Another example from English is that

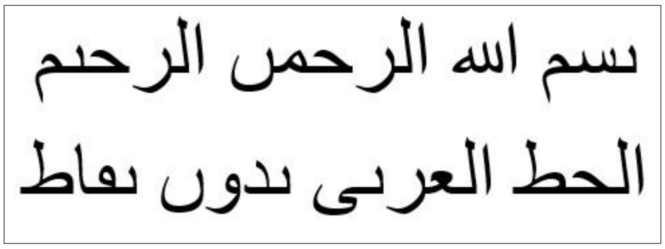

ٮسم الله الرحمں الرحىم
الحط العرٮى ٮدوں ٮڡاط

Figure 5.2: An Arabic text example that has identical characters but different pronunciation

people can readily recognize the correct form for the letter ‘1’ in the following three strings based on the surrounding context: “l234”, “lntelligence” and “left”. Therefore, humans’ existing knowledge is essential for the reading task. Researchers in the field of cognitive psychology have presented several theories to explain the principal cognitive processes in human recognition. For example, several studies such as Bobik and Sayre, (1963) have shown that human recognition of a character is directly related to understanding the general idea of the character, the basic elements that define it, irrespective of color and size. Some cognitive scientists argue that the visual system is sensitive to the way the characters are physically produced; thus the rules that are followed in order to write a character affects the perception of it (Kandel et al., 2000; Tse and Cavanagh, 2000). According to this viewpoint, humans extract writing information of the characters from an image by mentally tracing over them in their minds as if they were writing them with a pen.

Also, the most used strategy for developing handwriting skills in primary schools is the tracing handwriting strategy (Vinter and Chartrel, 2010). This is a practice that comprises tracing characters or words, which have been printed using dots, by a pen, as shown in Figure 5.3. The worksheet in Figure 5.3 is an example of handwriting exercise that have been used in teaching children on handwriting by demonstrating the action of tracing movements which the learners should follow in order to produce a character or a word in Arabic and English.

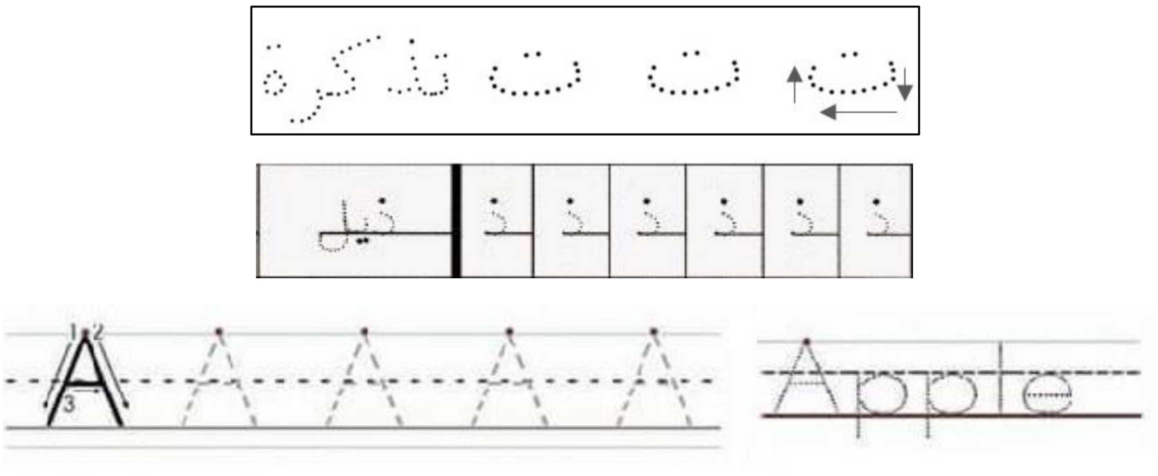


Figure 5.3: An example of a worksheet that adopts the tracing handwriting strategy, from (Garcia, 2005).

The proposed feature extraction method, to some extent, follows the above insights from cognitive science in order to design a robust Arabic OCR system.

5.3 New Feature Extraction Method

The feature extraction phase processes the skeleton text images produced by the thinning algorithm that was described in the previous chapter. As mentioned in literature review in section 2.2 Arabic scripts have some dotted characters which could contain one, two or three dots. Moreover, these dots could be the only feature that could be used to differentiate between dotted characters. Recall that Arabic script was not originally dotted: referring to the literature review in Chapter 2, some issues have been raised when recognizing dots in Arabic script. For example, dots in Arabic text may characterised as noise. In order to help recognition, pre-processing techniques, such as a noise removal algorithm may remove the dots of Arabic text. Also, applying thinning algorithm on Arabic text images may affect the preserving of the dots in Arabic text. Due to these issues, researchers in Arabic OCR have extracted the dots information before applying pre-processing methods. However, in some Arabic fonts, two dots are joined together and printed as one stroke which is similar to the diacritical mark called Fat-hah. As a result, it is difficult to isolate connected dots and to distinguish between the dots and the Fat-hah diacritical mark. Thus, the feature extraction process will only deal with the main part of Arabic script. That is, it will be assumed that all Arabic words in the text images are non-dotted, as dots will be recovered in the post-processing stage. Moreover, ignoring recognising dots in this stage will result in reducing the number of character models needed in the classification stage as will be discussed in section 7.2.1.

Referring to the literature review, most studies in previously developing analytical Arabic OCR systems perform the character segmentation phase prior to the feature extraction phase. However, in this work, the feature extraction stage does not require performing character segmentation in advance.

The proposed method is based on feature extraction methods using direction codes, such as the Freeman chain code (Freeman, 1961) which provides information about the structure of an object based on its boundary. It follows the object's boundary from a fixed starting point and continues moving in a clockwise or anticlockwise direction until it reaches the starting point. For more details of the Freeman chain code, refer to Nixon and Aguado, (2008).

A study by Kocyigit (2012) proposes a feature extraction method based on direction codes for tracing English characters' skeletons in order to extract the direction movement features used to follow the character's skeleton shape. This method has been adopted here for extracting features from Arabic script with some modifications (described below) that control the tracing movements in order to extract feature from cursive printed text, since the method was only investigated for English script.

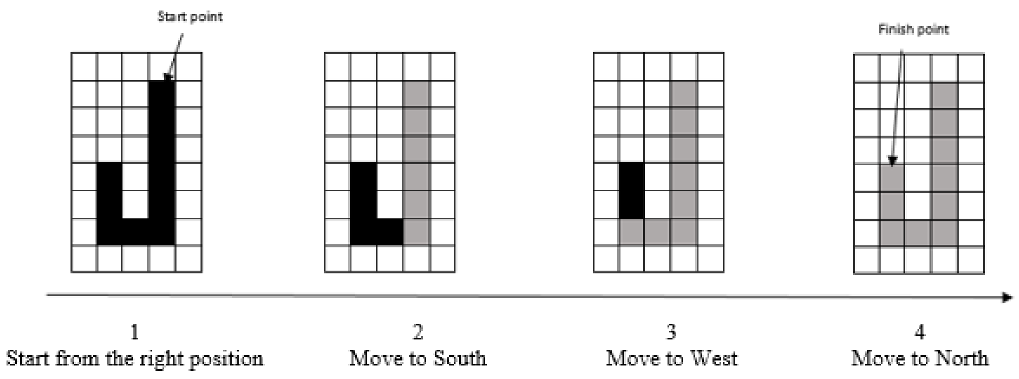


Figure 5.4. An example of how an Arabic letter "*Laam*" is traced
Note: the black squares are the unprocessed pixels of the character image and the gray squares are the pixels that have already been processed.

In this work, the proposed algorithm extracts the direction movements as features from Arabic words by tracing their skeletons (thinned images). This resembles the way a human might write a word or a character with a pen using the handwriting development strategy, which is a tracing strategy as discussed previously. When individuals trace a word or a character, they place the pen at a certain point and move in different directions in order to produce the word or the character. This can be illustrated by assuming the start point of the Arabic letter "*Laam*" (ل) is the right top corner of the letter. In order to write the Arabic letter, we begin from the start point and move forward in a south direction, then turn left and move forward to the west. After that, we move forward in the north direction; see Figure 5.4 for illustration.

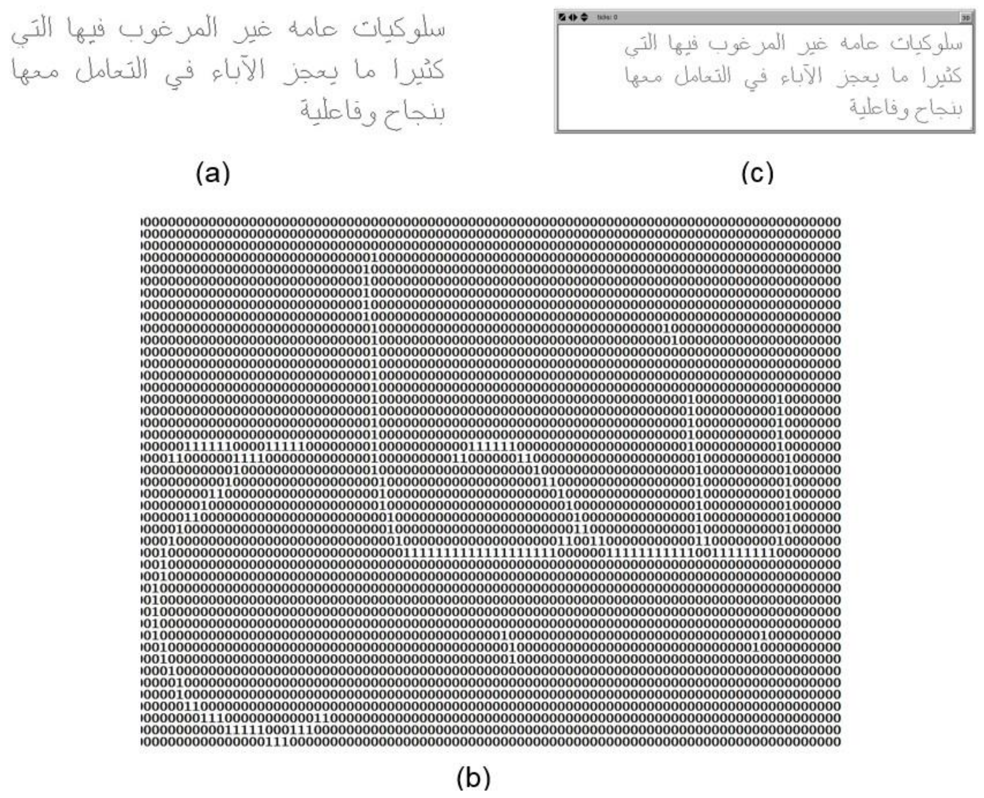

سلوكيات عامه غير المرغوب فيها التي
كثيرا ما يعجز الآباء في التعامل معها
بنجاح وفاعلية
(a)
(c)
(b)

Figure 5.5. An example of an Arabic text image conveyed into NetLogo
Note: (a) the original thinned text image, (b) the 2D array for a part of (a) and (c) the text image imported into NetLogo.

This can be reproduced by using an agent-based model implemented in NetLogo that uses turtle movements controlled from a first-person perspective. To this aim, several pre-processing steps are applied to the text images in order to prepare them for feature extraction. Firstly, the thinned text image to be recognised is imported into the system as a 2D array of 1 (black patches) and 0 (white patches) values to be read in NetLogo, as shown in Figure 5.5. Then, the feature extraction method is broken into three sub-stages: baseline determination; word segmentation and extraction of features, as shown in Figure 5.6.

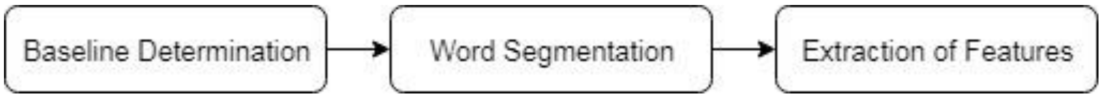
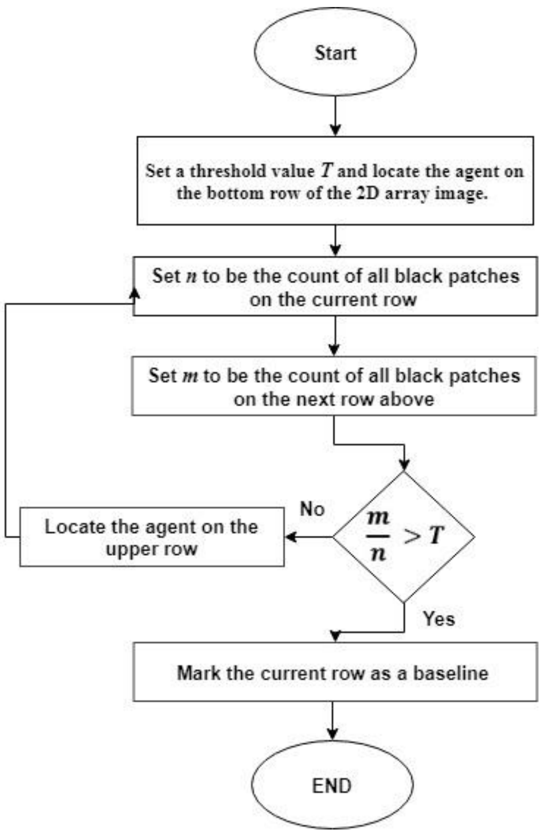


Figure 5.6. Block diagram of feature extraction method

5.3.3 Baseline determination

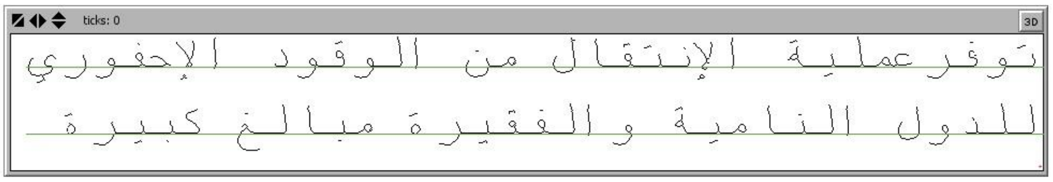
The proposed method relies on the baselines of Arabic text. Thus, the baseline of Arabic text needs to be detected. The baseline of Arabic script is a horizontal line through which Arabic characters are joined. It usually has the maximal amount of black pixels, especially for printed Arabic text (although not for handwritten text images in which the writer is moving the pen above and below the baseline). In the literature, there a number of baseline determination algorithms (see Chapter 2, section 2.3.1). For this work, the method proposed by Kocyigit (2012) has been used due to its effectiveness in the feature extraction stage (as evidenced by the accuracy performance evaluation detailed below). The baselines of text images are extracted using a thresholding algorithm developed in the NetLogo implementation. The steps of the baseline determination algorithm are as follows in the flowchart provided in Figure 5.7. The threshold value T used in the algorithm is changeable (from 1 to 10) depending on the text density of the lines. The algorithm calculates the ratio of the counts of black patches on adjacent rows and determines that a new line of text starts when this ratio exceeds a selected threshold value. To always ensure non-zero counts, a black border is placed around the 2D array of the text image.

In this work, it is assumed that the input Arabic text images have 0˚ skew, as skew correction is beyond the scope of this research. Figure 5.8 shows an example of Arabic text images after determining the baselines, which are green colored lines shown through the text, by the baseline determination algorithm.



Start
Set a threshold value T and locate the agent on the bottom row of the 2D array image.
Set n to be the count of all black patches on the current row
Set m to be the count of all black patches on the next row above
$\frac{m}{n} > T$
No
Locate the agent on the upper row
Yes
Mark the current row as a baseline
END

Figure 5.7: Flowchart of the feature baseline determination algorithm



توفر عملية الإنتقال من الوقود الإحفوري
للدول النامية والفقيرة مبالغ كبيرة

Figure 5.8. Screenshot from the NetLogo model that was developed for baseline determination

5.3.4 Word segmentation

Determining where each word is in a text image is required in order to segment a word into characters. In this work, the white space (white patches) between each word is used for word segmentation. In particular, the tracing agent records the finish point for each word (connected component) after completing the movement over the word (see step 9 in Figure 5.9). When the tracing agent locates the finish point and comes to the start point of the next word, the white space between the start point and finish point is marked as sub-word space (segmenting each text line into sub-words, since an Arabic word may have more than one connected components (sub-words) as mentioned in chapter 2).

5.3.5 Extraction of features

This section explains how the features of Arabic text images are extracted. Figure 5.9 states the algorithm's flowchart for determining how the tracing agent moves over the Arabic text. After determining the baselines in the text image, the tracing agent in the NetLogo environment is placed at the right bottom of the lowest baseline of the Arabic text images. It then visits each baseline from bottom to top of the image moving across the baseline from right to left (see steps 1, 2 and 3 in Figure 5.9). We choose the starting point at the right bottom on account of the fact that Arabic text is written from right to left and a pen is placed on the right bottom of a baseline when writing Arabic text.

While the tracing agent is moving over the selected row, the agent marks the first black patch on the row as the start point of a word (refer to step 4). Then, the tracing agent moves onto the black neighbouring patches using four directions: south; north; west; and east, according to the direction of the patch (see steps 5 and 6). Due to the fact that Arabic script has some curved characters, the algorithm will treat the curved characters as diagonally connected pixels. The agent checks if the pixels of the curved parts of the characters create a corner and changes the direction. If not, the agent accepts these pixels as the same direction which the agent is already moving in.

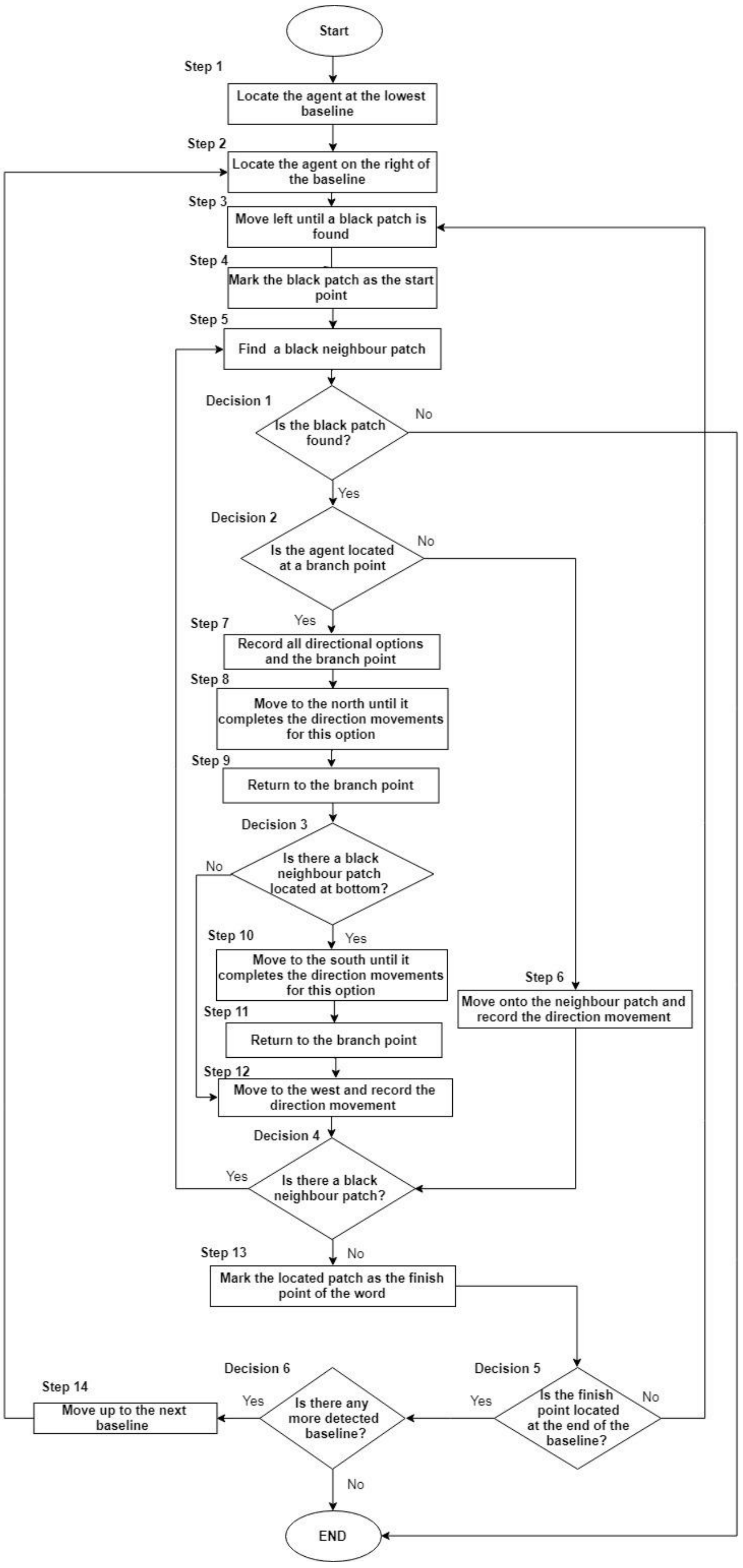

Start
Step 1
Locate the agent at the lowest baseline
Step 2
Locate the agent on the right of the baseline
Step 3
Move left until a black patch is found
Step 4
Mark the black patch as the start point
Step 5
Find a black neighbour patch
Decision 1
Is the black patch found?
No
Yes
Decision 2
Is the agent located at a branch point
No
Yes
Step 7
Record all directional options and the branch point
Step 8
Move to the north until it completes the direction movements for this option
Step 9
Return to the branch point
Decision 3
Is there a black neighbour patch located at bottom?
No
Yes
Step 10
Move to the south until it completes the direction movements for this option
Step 6
Move onto the neighbour patch and record the direction movement
Step 11
Return to the branch point
Step 12
Move to the west and record the direction movement
Decision 4
Yes
Is there a black neighbour patch?
No
Step 13
Mark the located patch as the finish point of the word
Decision 6
Decision 5
Step 14
Move up to the next baseline
Yes
Is there any more detected baseline?
Yes
Is the finish point located at the end of the baseline?
No
No
END

Figure 5.9: Flowchart of feature extraction algorithm

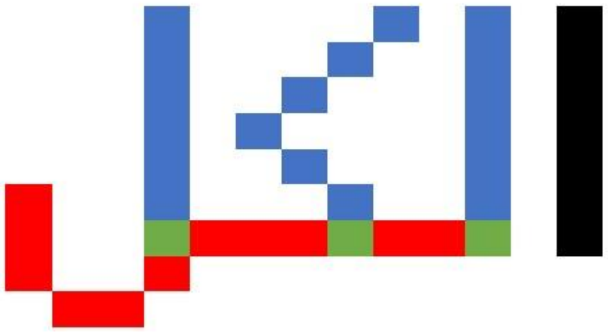
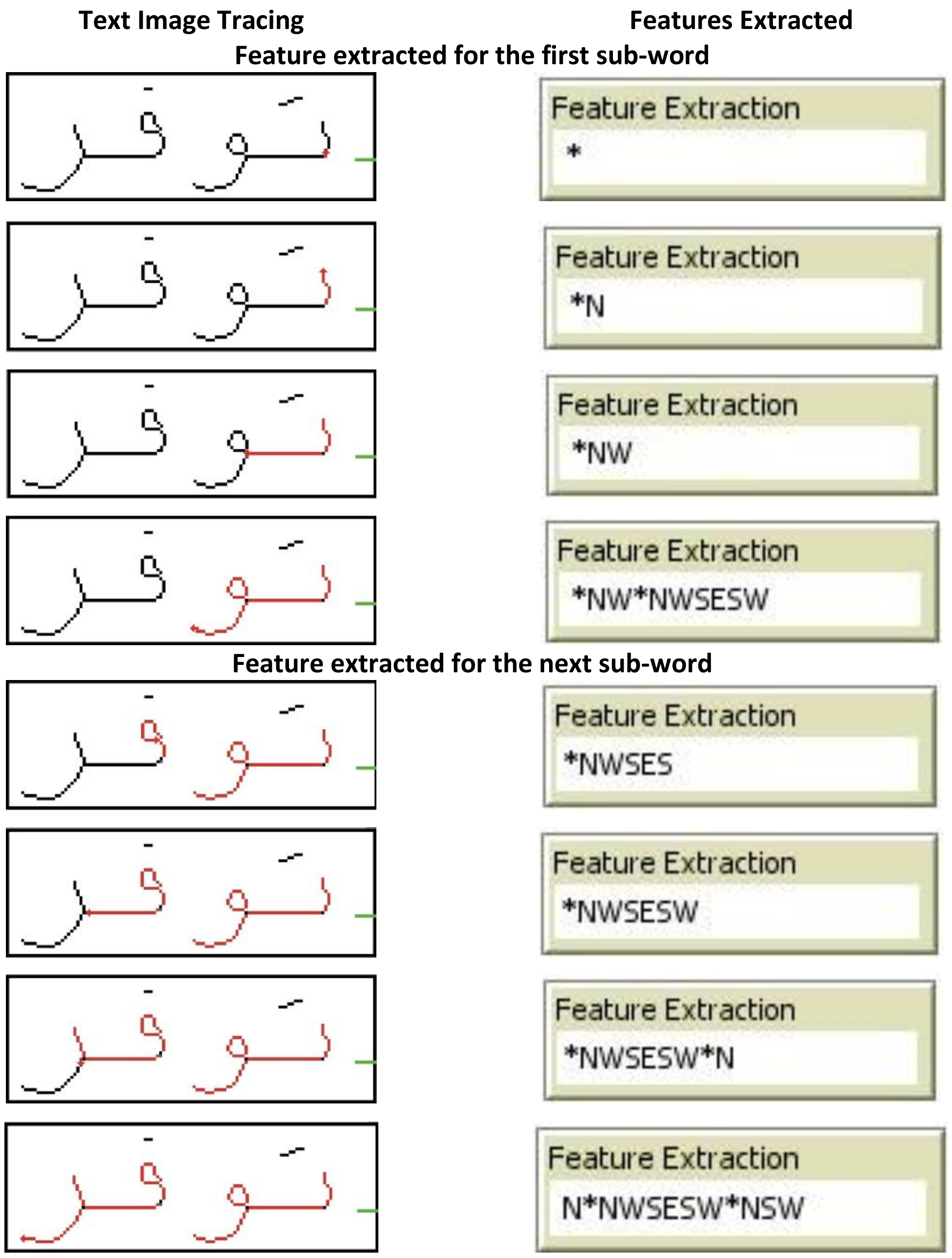


Figure 5.10: Example of three branch points that occur in the Arabic word ("الكل")

Note: When the agent finds the branch points (the first and the second green pixels), the agent moves first to the north direction (the blue pixels), then to west direction (the red pixels). While the agent finds the branch point (the third green pixel), the agent moves first to the north direction (the blue pixels), then to the south direction (the red pixels).

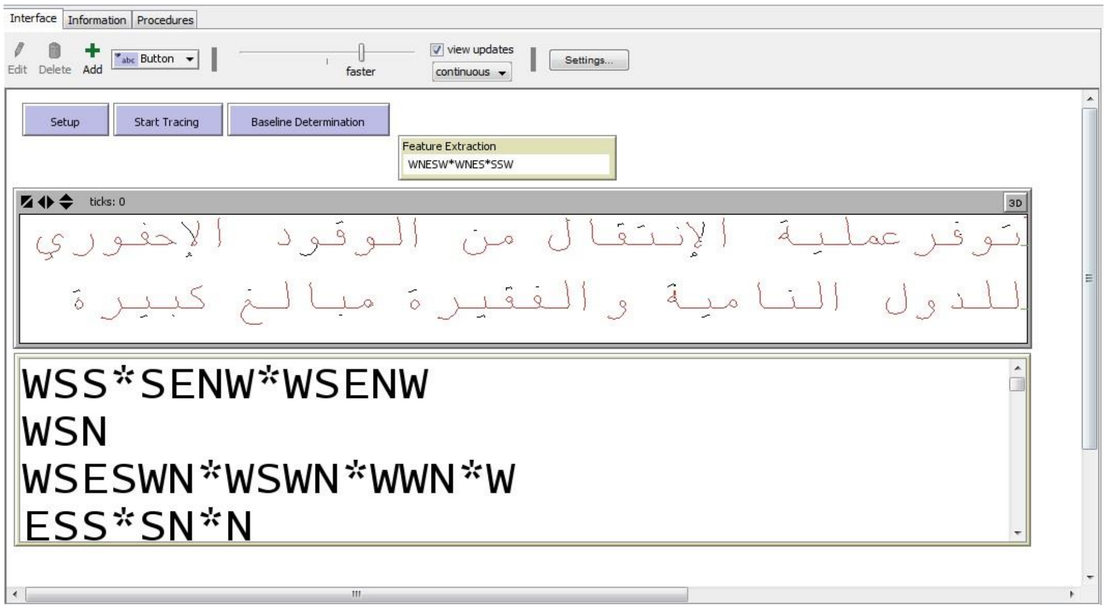
During the tracing process, the tracing agent may face a branch point which presents more than one direction for the next step of the agent. If the agent faces a branch point, it records all options and the branch point. Then it follows all the options one by one in order with priority given to move first to the north, then to the south, and finally to the west (refer to decision 2 and 3 and steps 7 to 12 in Figure 5.9). For example if there are two directions on the next step of the agent where one of them is to the north and the other is to the west, the agent first follows the north direction. When it has completed the movement for this option, it returns to the branch point and follows the west direction. We set these steps due to the fact that a branch point in an Arabic word has two or three directions – north, south and west; see Figure 5.10 for illustration. Therefore, it was decided to control the order the agent moves over the Arabic word, as this is a valuable step for the next stage of the PATRION OCR which will be discussed in the next chapter.

Figure 5.11 displays the tracing process and the features extracted from an Arabic word. The initials "N", "S", "E", "W" in Figure 11 refer to the direction movements which are north, south, east and west, respectively and the symbol "*" refers to the branch point. The red lines in the figure illustrate how the tracing process proceeds. In Figure 5.12, a screenshot from the NetLogo model is shown while feature extraction is being performed on an Arabic text image. It is obvious that the feature string which is extracted by the feature extraction method is irreversible. Thus, it is not possible to recover the original text image by using the produced feature strings. However, this does not affect the performance of the OCR system which will be discussed in section 7.2.1.


Text Image Tracing
Features Extracted
Feature extracted for the first sub-word
Feature extracted for the next sub-word

```
Feature Extraction
*

Feature Extraction
*N

Feature Extraction
*NW

Feature Extraction
*NW*NWSESW

Feature Extraction
*NWSES

Feature Extraction
*NWSESW

Feature Extraction
*NWSESW*N

Feature Extraction
N*NWSESW*NSW
```

Figure 5.11: Tracing example for an Arabic word with features extracted in NetLogo
Note: The tracing is performed from right to left. The red lines illustrate the progress as the tracing process proceeds.



Feature Extraction: WNESW*WNES*SSW

توفر عملية الإنتقال من الوقود الإحفوري
للدول النامية والفقيرة مبالغ كبيرة

```
WSS*SENW*WSENW
WSN
WSESWN*WSWN*WWN*W
ESS*SN*N
```

Figure 5.12: Screenshot of the NetLogo model extracting features from an Arabic text image
Note: the extracting features are shown in the output box below.

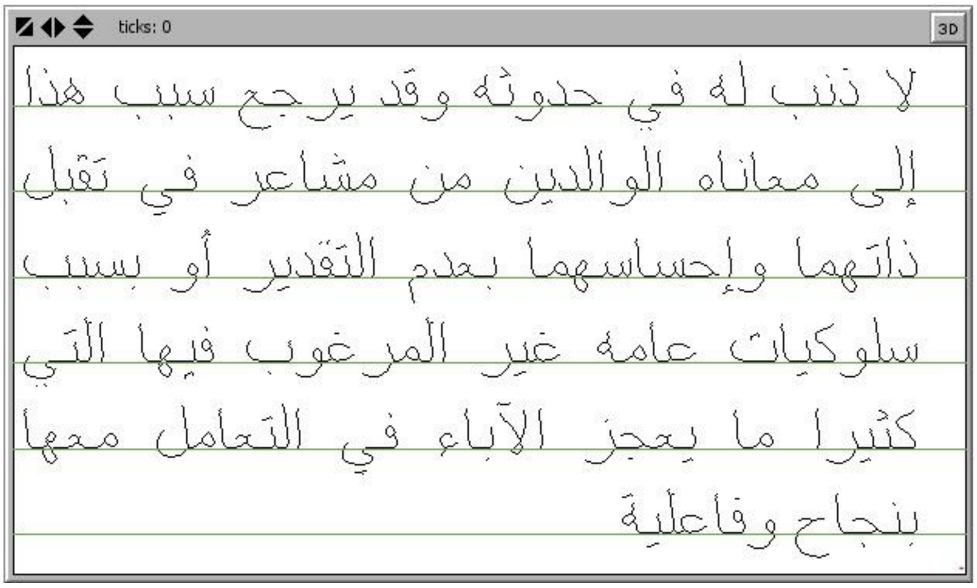
5.4 Experimental Evaluation

In this section, three experimental evaluations are conducted in order to assess the baseline algorithm, word segmentation method and the feature extraction methods. The KAFD dataset, which was mentioned in Chapter 3, has been used in these experimental evaluations.

5.4.1 Baseline determination method evaluation experiment

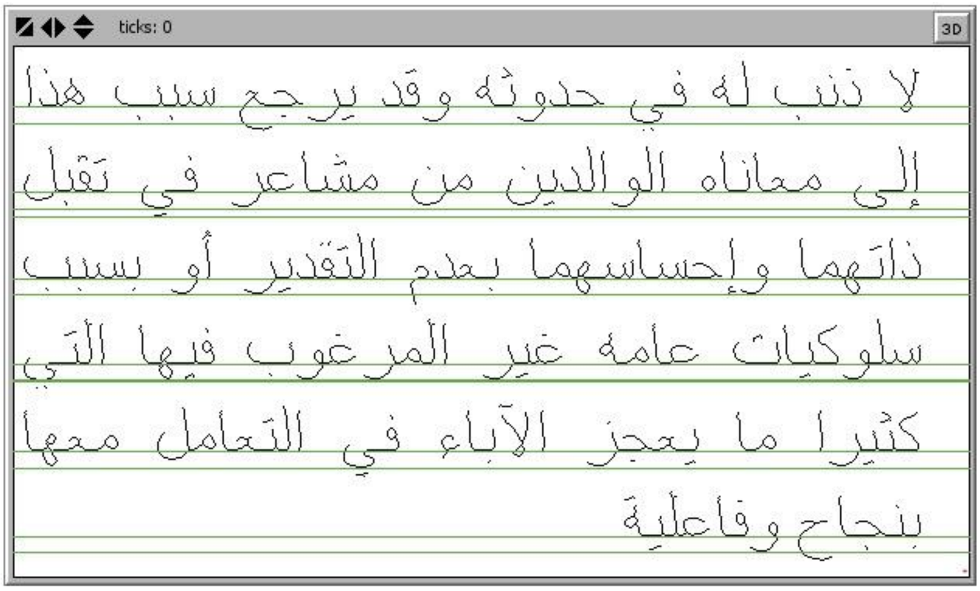
The baseline determination algorithm is evaluated on Arabic printed text images, which were selected from KAFD dataset, at 300 dpi and with 24 pitch size multiple font types, namely AdvertisingBold, Simplified Arabic and Traditional Arabic. The text images contain 105 baselines. The accuracy of the algorithm relies on the threshold value T. For the text image samples in this evaluation, we manually adjusted the T value for accurate baseline determination. We found that T values between 5 and 7 were the most accurate threshold values in the test samples. Figure 5.13 shows how the algorithm has determined the baselines when the threshold value was adjusted to 7 whereas, Figure 5.14 shows the output of the baseline algorithm when selecting a threshold value of 3. The algorithm obtained 100%

accuracy when selecting 7 as a threshold value. Table 5.1 shows the evaluation results which were obtained through the evaluation experiment on the sample text images.



لا ذنب له في حدوثه وقد يرجع سبب هذا
إلى معاناه الوالدين من مشاعر في تقبل
ذاتهما وإحساسهما بعدم التقدير أو بسبب
سلوكيات عامة غير المرغوب فيها التي
كثيرا ما يعجز الآباء في التعامل معها
بنجاح وفاعلية

Figure 5.13: Screenshot from NetLogo model showing example of correct baseline determination using 7 as a threshold value



لا ذنب له في حدوثه وقد يرجع سبب هذا
إلى معاناه الوالدين من مشاعر في تقبل
ذاتهما وإحساسهما بعدم التقدير أو بسبب
سلوكيات عامة غير المرغوب فيها التي
كثيرا ما يعجز الآباء في التعامل معها
بنجاح وفاعلية

Figure 5.14: Screenshot from NetLogo model showing example of incorrect baseline determination using 3 as a threshold value

Font Type	Total Number of Baselines	Number of Baselines Correctly Determined	Accuracy
AdvertisingBold	39	39	100%
Simplified Arabic	34	34	100%
Traditional Arabic	32	32	100%
Total	105	105	100%

Table 5.1: Results of baseline determination algorithm on sample test text images on different Arabic fonts using threshold values between 5 and 7 based on text images

5.4.2 Word segmentation method evaluation experiment

In this work, the word segmentation method attempts to segment Arabic text in the images into sub-words. Thus, to evaluate the word segmentation method, the ground truth of the dataset has been modified by adding a space between each sub-word in the test sample as only word spaces (spaces between words) are available in the ground truth of the dataset. The word segmentation method is evaluated by the following two measures:

$$Recall = 100 \times a/(a+c) \qquad (5.1)$$

$$Precision = 100 \times a/(a+b) \qquad (5.2)$$

where a is correct insertion of space, b is spurious space insertion and c is missed space. A perfect word segmentation will have recall and precision of 100%. Using text images that contain 100 Arabic words, the method produced a recall of 100% and a precision of 96.99%. It seems that the algorithm suffers from inserting spurious spaces (over segmentation). This issue occurred as a result of the thinning algorithm producing disconnections (not maintaining the connectivity) through Arabic words.

5.4.3 Feature extraction method evaluation experiment

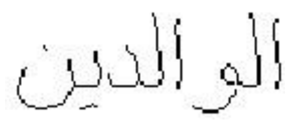
To evaluate the feature extraction method, a ground truth of directional features of the images has been generated in order to be able to make a comparison of the correct output string and the generated extracted features output string. The evaluation in this experiment is based on the edit distance between the extracted features string and the ground truth string. The edit distance here is the minimum transformation sequence that converts the extracted features string into the ground truth string. The Accuracy *A* is defined as:

$$A = \frac{f-e}{f} \times 100 \tag{5.3}$$

where e is the edit distance, and f is the number of features in the ground truth string. The cost for each edit operation is defined as 1. For example, Figure 5.15 illustrates the edit distance between the feature string (b), which was extracted from the thinned image (a) and the ground truth string (c) is 3 because three operations are required to convert the feature string (b) to the ground truth string (c) which are adding one symbol "*" (branch point) and adding two direction strings, "W", and "S", that are written in red colour.

The average accuracy of the feature extraction technique was found to be 98.07%. It is critical to note that most of the errors produced by the method are related to the thinning algorithm. That is, when the thinning algorithm failed in persevering connectivity (disconnection), the feature extraction makes an error in extracting the directional features; see Figure 5.16 for illustration. Another problem that arose is due the loss of visual information of the original text images that have been produced by the thinning algorithm. In fact this issue occurs mostly in AdvertisingBold font with loop characters, as shown in Figure 5.17.

(a)



(b)

NWSN*WN* – NWN*WN* – N – ESWNWN*– N

(C)

NWSN*WN* – NWN*WN* – N – WSESWN*WN*– N

Figure 5.15: An example of edit distance between two feature strings
Note: (a) A thinned Arabic word, (b) the extracted feature string from (a) and (c) the ground truth string for (a) with the edit operations highlighted in red colour

Table 5.2 summarizes the errors that occurred in the feature extraction method according to different error types which have been produced by the thinning algorithm. The data in Table 5.2 shows that most of the errors in the test samples occurred in text images that involved the AdvertisingBold font. The high number of errors that occurred in text images of AdvertisingBold font is likely due to the inefficiency of the thinning algorithm (discussed in chapter 4) in thinning bold style text.

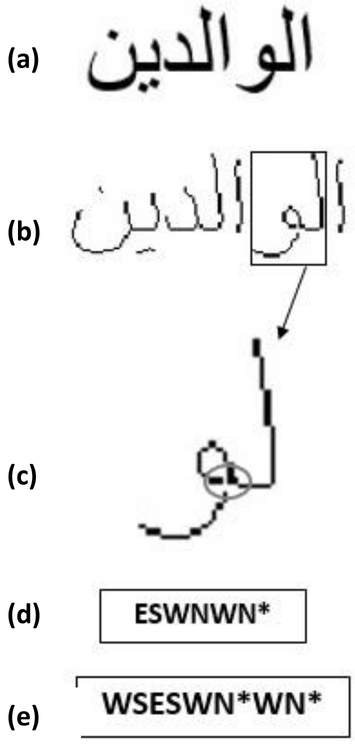


Figure 5.16. Example of the effect of disconnection errors (produced by thinning algorithm) on the feature extraction algorithm

Note: (a) An original Arabic word image, (b) the thinned image of (a), (c) the sub-word in which the disconnection occurred (highlighted in a circle), (d) the extracted feature string from (c) by the feature extraction algorithm and (e) the ground truth for (c).

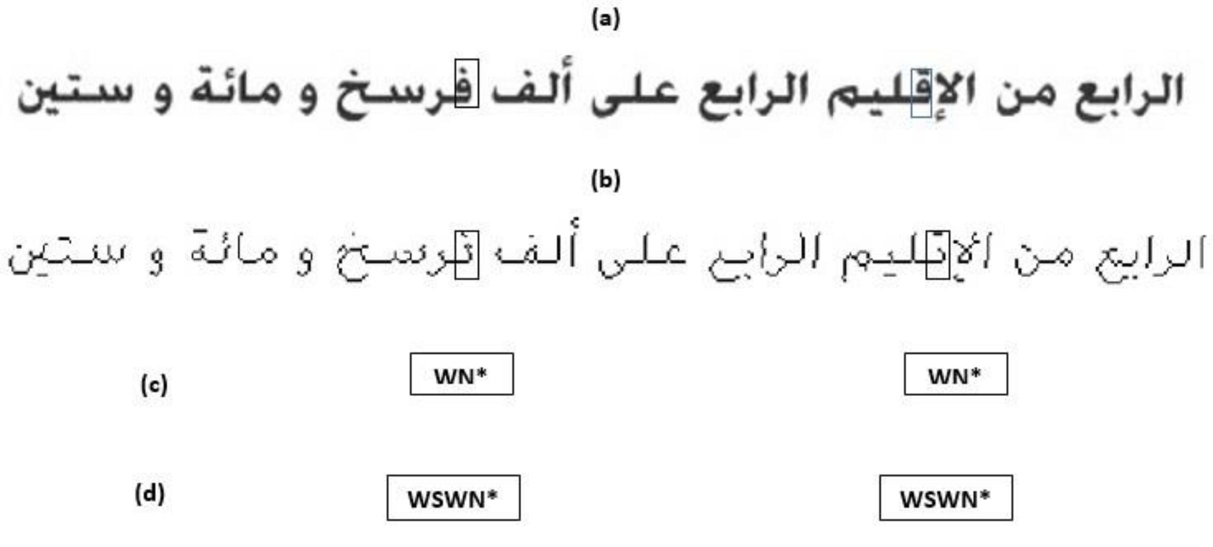


Figure 5.17. Example of the effect of loss of visual information (caused by thinning algorithm) on the feature extraction algorithm

Note: (a) An original Arabic text image (the patterns in which the error occurred are highlighted in squares), (b) the thinned image of (a), (c) the extracted feature string from the highlighted characters in (b), and (d) the ground truth for the highlighted characters in (b).

Error Category / Font	Disconnection	Loss of visual Information	Total Errors **(%)**
AdvertisingBold	5	10	15 **(53.5%)**
Simplified Arabic	2	4	6 **(21.4%)**
Traditional Arabic	4	3	7 **(25%)**
Total Errors (%)	11 **(39.2%)**	17 **(60.7%)**	28

Table 5.2: Error analysis according to different fonts and error types

5.5 Summary and Discussion

In this chapter, a new technique, based on a cognitive insight, for extracting features from non-dotted Arabic text images has been proposed and developed using an agent-based NetLogo model. Some pre-processing steps are applied to prepare the text images for feature extraction, such as the baseline determination and the word segmentation.

The feature extraction method transforms an Arabic word into a string of direction movements, which are representative of how the agent in the model traces Arabic words, combined with the branch points. The average performance accuracy of the feature extraction algorithm on a sample of 100 images is 98.07%.

The following two chapters use these extracted features in order to design a system for recognising Arabic text.

Chapter 6: Character Segmentation

6.1 Introduction

Chapter 5 presents a method for extracting direction code features from Arabic text images. The next stage in this work for the developed PATRION system is the character segmentation which differs from a typical Arabic OCR system in that here the character segmentation proceeds subsequently to the feature extraction stage, as shown in Figure 6.1. Character segmentation is the process of segmenting a word into its characters. Referring to the literature in chapter 2, it is stated that character segmentation is a critical step as most recognition errors are produced as a result of character segmentation (Al-Badr and Mahmoud, 1995; Tamen and Drias, 2010).

In the literature, there are a number of image-level algorithms for segmenting Arabic words into characters; for a comprehensive survey in character segmentation for Arabic script, refer to Zeki et al. (2011). Although many studies

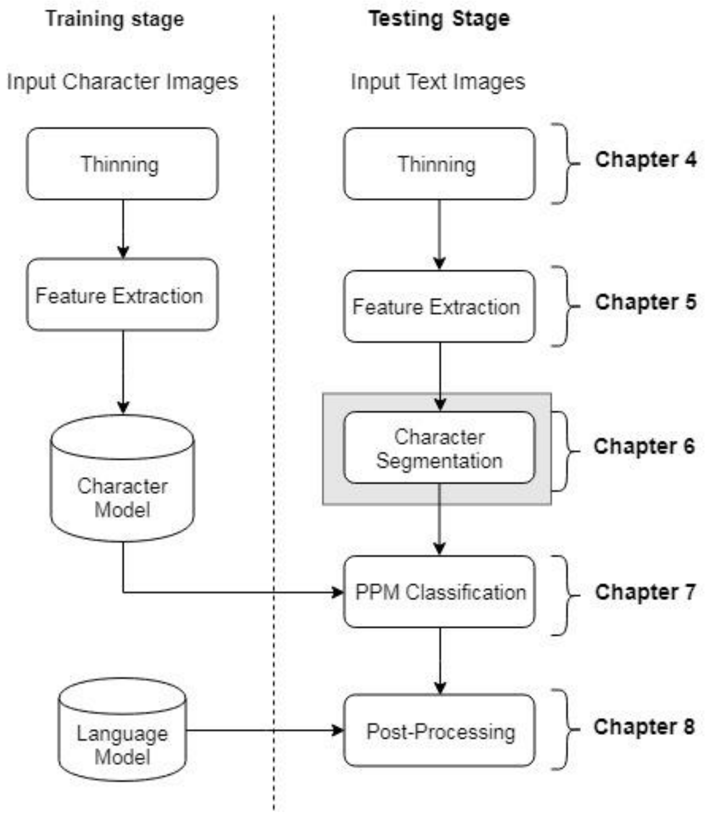


Figure 6.1. Block diagram of the PATRION OCR system
Note: The shaded area shows the scope of the current chapter 'Character Segmentation'.

have provided different character segmentation algorithms, there are no accurate character segmentation algorithms (Gouda and Rashwan, 2004; Zeki et al., 2011). A number of the algorithms face various segmentation problems, such as over-segmentation and under-segmentation. A major drawback of applying such character segmentation techniques for Arabic text is that they fail in segmenting overlapping characters. Overlapping characters in Arabic script cause difficulties for a number of existing character segmentation algorithms (see Figure 6.2).

Some studies overcome the challenges of character segmentation by developing Arabic OCR systems based on a free-segmentation or holistic approach which recognize words or sub-words as a whole unit with no segmentation. However, such systems can only recognize Arabic words that appear in the lexicon because these systems are lexicon based systems. This provides the motivation for developing a novel character segmentation method for the Arabic OCR system. Doing this enables the delivery of a robust Arabic OCR system (as evidenced by the results provided in Chapter 8).

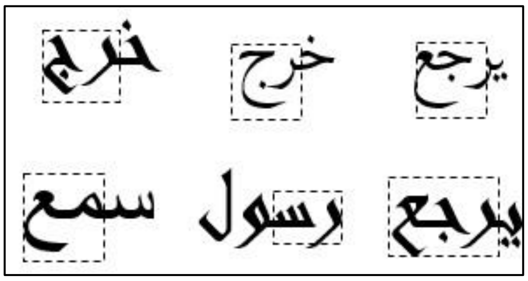


Figure 6.2: Examples of overlapping characters highlighted in Arabic words in different fonts

This chapter firstly discuss the features of Arabic script that are useful when designing character segmentation methods. Then, section 6.3 presents a new method for segmenting Arabic words into characters. In section 6.4, an evaluation experiment is conducted on the proposed character segmentation method.

6.2 Arabic script features used for designing the character segmentation technique

The challenge of character segmentation task is undoubtedly due to the characteristics of Arabic script, as discussed in chapter 2 and discussed again above. As a result, the special characteristics of written Arabic script should not be ignored in order to design a robust character segmentation method for Arabic script.

One feature of Arabic script that can be exploited for designing a character segmentation technique is the baseline. As already discussed in chapter 2, the baseline is the horizontal line through which Arabic characters are joined together. That is, when an Arabic word is written, the baseline is used to join the characters together in order to produce the word. There is a part of the baseline which is between each of the connected characters called the *junction line*. The junction line is used by different image processing algorithms to segment Arabic words into characters. However, this feature may not work well in image-level algorithms as the length of the junction line varies in different fonts being often short in many printed Arabic fonts. Also, character segmentation approaches based on the junction line suffer in segmenting overlapping characters in Arabic text (Gouda and Rashwan, 2004).

Another feature of Arabic script that can be utilized for character segmentation is the branch point which occurs at the beginning of a connected Arabic character in a word and after a junction line. The branch point as described in chapter 5 is the point where there is more than one direction to follow in the process of executing the tracing algorithm.

According to the two above characteristics which are the junction line and the branch point, there is usually a character segmentation point in a word if there is a branch point after a junction line. Figure 6.3 illustrates the two selected features of Arabic script for detecting the character segmentation point. Recall that the Arabic language has some characters which divide one word into one or more pieces of words called sub-words. These characters are not connected with the succeeding characters and they can be only linked from their right sides. Therefore, when they occur in the middle of an Arabic word, the word is divided into sub-words by adding a space. Figure 6.3 (a) shows an Arabic word comprising three sub-words; two are isolated characters and the third is a connected sub-word of five characters. Figure 6.3 (b) and (c) shows that the junction line and the branch point are appear only in a connected sub-word, not in sub-words with isolated characters.

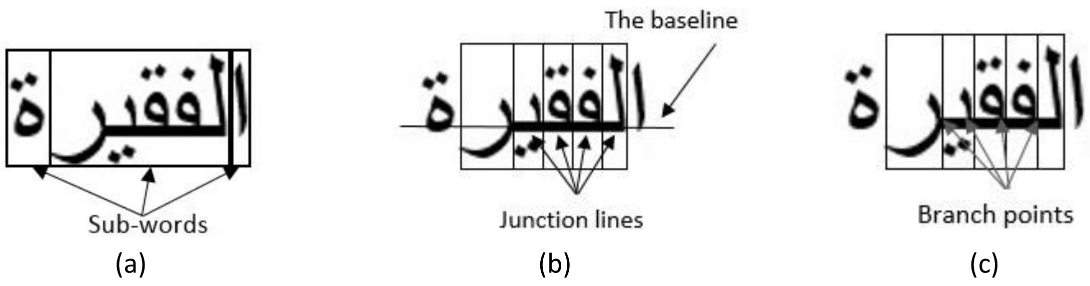


Figure 6.3: Junction line and branch point features of an Arabic word

6.3 New character segmentation approach

The new approach for character segmentation is applied in order to overcome the aforementioned problems. The extracted features string described in the previous chapter is used for performing the character segmentation task. Recall that the extracted features from Arabic text is comprised of four directions represented as symbols in a text string: south ('S'); north ('N'); west ('W'); east ('E') and a branch point symbol ("*").

The feature extraction algorithm in chapter 5 extracts the west direction as the last direction feature after detecting a branch point, since this confirms that the tracing agent does not move to a succeeding character until it extracts the whole features of the preceding character in a word. That is, it does not move to the next character in the connected word until it finishes tracing the character being processed.

The aim of the character segmentation technique in this work is to identify the segmentation point in the extracted features string in order to break the connected characters in a connected word or a connected sub-word.

After investigation of the direction features extracted from printed Arabic script, we noticed that the west 'W' direction feature symbol commonly represents the junction line (and does not represent a part of a character) when it is followed by a branch point (the star '*' in the extracted features string). Therefore, there should be a character segmentation point between the 'W' and '*' symbols.

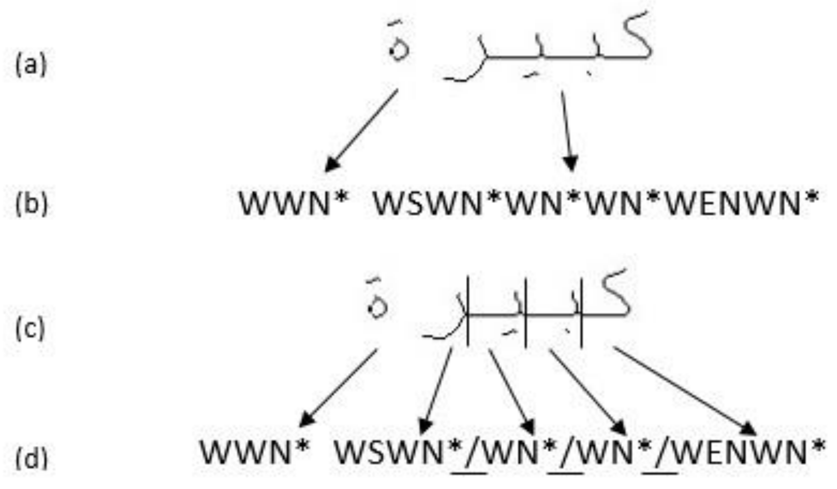


Figure 6.4. A sample of applying character segmentation on an Arabic word
Note: (a) An Arabic word with two sub-words, (b) the feature extraction string produced from (a), (c) segmentation points in the word and (d) the feature extraction string (b) after applying the character segmentation technique.

Simply, in order to segment an Arabic word into characters, the new approach inserts a slash sign '/' (to indicate a segment break) between the 'W' and '*' symbols into the output of the feature extraction algorithm that was discussed in chapter 5. Figure 6.4 shows a sample of extracted feature string (b) that was produced, using the feature extraction method in chapter 5, from the Arabic word shown in (a). The Arabic word has two sub-words: one consists of four connected characters which should be represented as four symbols in the text string when segmented; and one sub-word of an isolated character which should not be segmented. Figure 6.4 (c) shows the positions (segmentation points) where the sub-word should be segmented. Figure 6.4 (c) shows the feature extraction string (b) after applying the character segmentation technique.

6.4 Experimental Results and Discussion

Comparing the performance of character segmentation methods is a challenging task due to the different experimental protocols, different testing datasets and different performance measures. Besides most studies in Arabic text recognition have only provided the overall performance accuracy of their systems. In this work, an experiment was designed in order to evaluate the performance of the novel character segmentation method.

The character segmentation method is evaluated on the dataset of 100 Arabic word images which were used in chapter 5. The text in the testing dataset comprises different features of Arabic script that may contribute to the challenges of Arabic character segmentation. Table 6.1 shows basic statistics of the test images sample in terms of Arabic script characteristics.

The character segmentation method is evaluated by the following two performance measures:

$$Recall = 100 \times s/(s+m) \tag{6.1}$$

$$Precision = 100 \times s/(s+b) \tag{6.2}$$

where s is correct insertion of the slash symbol '/', b is spurious slash sign insertion and m is missed slash sign. A perfect character segmentation will have recall and precision of 100%. The method produced a recall of 84.2% and a precision of 77.2%. The prime causes of the segmentation errors are due to specific features of Arabic script and the errors which have been delivered from the previous stages. Table 6.2 shows the segmentation error rate percentage of the method in terms of the features of Arabic script.

During the evaluation experiment, it was observed that there are a number of re-occurring situations where usually the character segmentation method did not perform well or produced segmentation errors. For instance, an over segmentation problem occurs on segmenting the Arabic character (Seen "س"). The method over segmented the (Seen) letter into three segments because it consists of two junction lines and two branches.

Features	**Count of items**
Word	100
Sub-word	129
Sub-word with isolated character	28
Overlapping character	89
Loop character	180
End-form character	101

Table 6.1: Testing dataset statistics

Features	Error rate
Overlapping character segmentation	4.5%
End-form character segmentation	14.8%
Loop character segmentation	6.9%

Table 6.2. Segmentation Error Rate Percentage

Furthermore, as stated in Table 6.2, segmenting end-form characters has the highest segmentation error rate which is 14.8%. This high error rate is related to three Arabic characters if they are found at the end of words or sub-words (the end form shape). These characters are Alef “ـا”, Haa “ـه” and Yaa “ـى”. The method did not segment these characters (under-segmentation) as there is no branch point after the junction line. Such characters will be dealt with at the post-processing stage in Chapter 8 rather than be corrected at this stage.

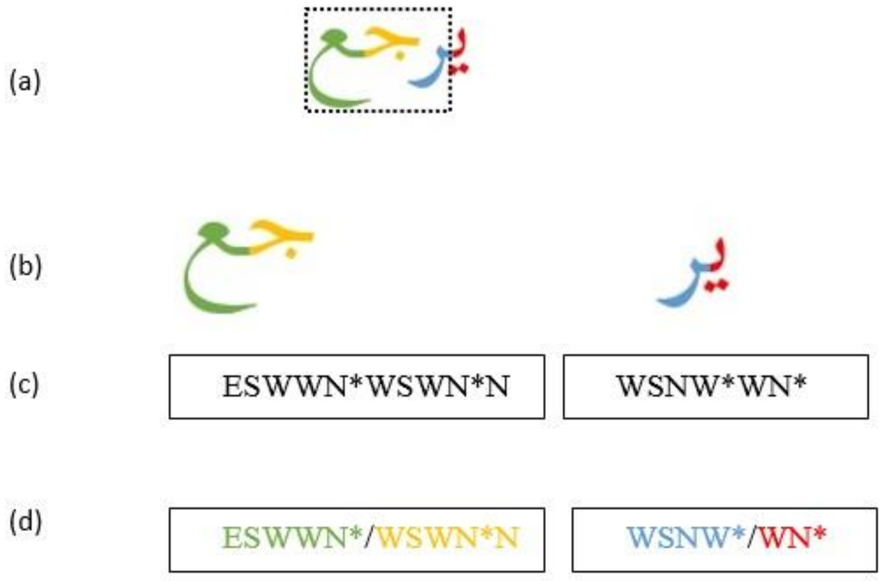


Figure 6.5. Sample of segmenting overlapping characters

Note: (a) An Arabic word containing overlapping characters (Traditional Arabic Font); (b) the sub-words of the word shown in (a); (c) the feature extraction string produced from each sub-word shown in (b); (d) the segmented feature extraction string after applying the character segmentation technique which are coloured corresponding to the way characters are coloured in (a).

The error rate percentage of the character segmentation method in relation to loop characters is about 7%, as shown in Table 6.2. The segmentation errors in segmenting loop characters are due to errors which have been produced in the two previous stages. In particular, when the thinning algorithm (discussed in Chapter

4) has produced a disconnection through a loop character’s form, a branch point after a junction line existed in the main form of the character. The feature extraction algorithm (discussed in Chapter 5) thus has extracted these features which are the west ‘W’ direction feature and the branch point ‘*’. As a result, the character segmentation method over segmented the character into more than one segment.

On the other hand, the segmentation method succeeds in dealing with the problems that arise when segmenting overlapping characters. For example, Figure 6.5 shows how overlapping characters in an Arabic word have been segmented successfully by the character segmentation method. The character segmentation produced only an error rate of 4.5% for segmenting overlapping characters. In fact, this error rate is related only with the Arabic ligature (لا) which is a combination of two overlapping characters: Lam (ل) and Alef (ا). The algorithm did not segment such ligatures, as there were no identified segmentation points. We overcome this issue by dealing with such ligatures as one character, so segmenting the ligature into characters is not required.

6.5 Summary and Discussion

In this chapter, a new character segmentation technique for segmenting connected Arabic words into characters has been provided. The character segmentation method uses the extracted features string, which have been obtained as a result of the feature extraction process described in Chapter 5, for performing the segmentation process.

The character segmentation method segments a connected word into characters by identifying the segmentation point in the extracted feature string. The segmentation point is determined based on two features of Arabic script: the junction line and the branch point. The method has been tested on a sample of 100 Arabic words images and has produced a recall of 84.2% and a precision of 77.2%. The character segmentation method leads to under-segmentation in some end-form characters. However, the method has successfully segmented overlapping characters.

In the following chapter, the segmented feature strings will be used for recognising Arabic text.

Chapter 7: Classification Using PPM

7.1 Chapter Introduction

Chapter 4 presents a method for extracting feature strings from printed Arabic text images. Then, chapter 5 describes a method that uses the features in order to segment Arabic words into characters. In this chapter, the segmented feature strings will be used for recognising Arabic text. In other words, the following stage of the developed PATRION system, which is the classification stage (as shown in Figure 7.1), is responsible for classifying the feature strings into the best character class in order to complete the recognition task.

In the literature, a number of classifiers have been used for Arabic OCR. However, no prior study exists on the use of compression based models as a classifier in Arabic text recognition. Therefore, in this work a compression based method has been applied for the first time as a classifier for the text recognition. Moreover, since the aim of the classification stage (implemented in the PATRION OCR system) in this work is to distinguish (classify) the feature strings (extracted in chapter 5). PPM, which is discussed in the review chapter, was chosen to be applied as a classifier.

The following sections of this chapter describe the use of the PPM compression-based method for text recognition (classification). Also, constructing PPM classifiers and experimental results for PPM model classification are reported. In particular, section 7.2 discusses the implementation of the PPM classifier. We then discuss experimental results using the PPM classifier in section 7.3. A summary and discussion of this chapter is presented in section 7.4.

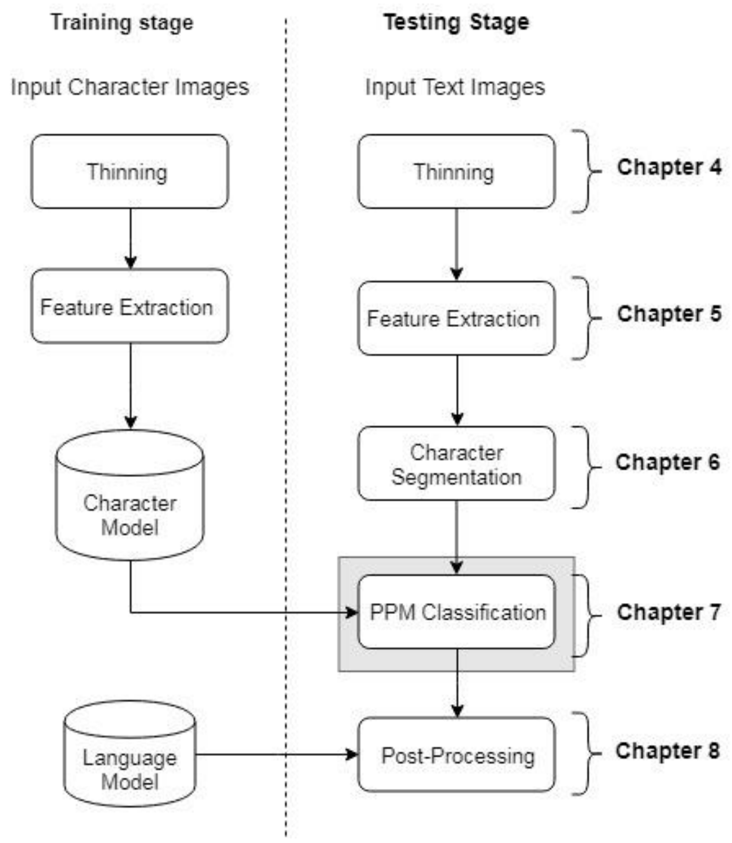


Figure 7.1: Block diagram of the PATRION OCR system
Note: The shaded area shows the scope of the current chapter ‘Classification using PPM’.

7.2 Implementation of PPM Classifier

Our approach is to recognise Arabic characters by the use of the PPM method that classifies the feature strings obtained from the previous chapter. The PPM classification approach in the PATRION OCR system consists of two stages: a training stage and a testing/recognition stage. During the training stage, character models are constructed using a pre-segmented character image dataset. Order 5 PPMD character models, which are implemented by the Tawa Toolkit, described in (Teahan, 2018), have been used for constructing the character models. During the testing stage, the recognition of both Arabic text and pre-segmented Arabic character images were analysed by the text classification method which is also provided by the toolkit. In the following sections, the two stages will be discussed in detail.

7.2.1 Stage I: Training stage

For constructing a PPM classifier, a training dataset consisting of pre-segmented printed Arabic characters is required as the main idea is to train PPM for isolated Arabic characters. The literature suggests a comprehensive isolated printed Arabic characters dataset named PAC-NFB which is provided by Alkholy (2016). The dataset is freely available and comprises character images of 10 different font types, 10 pitch sized and four styles (normal, italic, bold, bold italic), with 300 dot per inch (dpi). The dataset contains 4080 character images for each font with different variations. For this study, four various font types have been selected from the PAC-NFB dataset; namely, Arabic Transparent, AdvertisingBold, Simplified Arabic and Traditional Arabic. For each font, 12 and 24 pitch sizes have been selected. Additionally, normal and italic font styles for each font pitch size have been used.

Since this work only considers the main part of the Arabic text (non-dotted text), Arabic characters that share an identical visual shape have been joined in one group. This leads to a reduction of the number of character models as each PPM model is formed from the grouped characters in each group.

Table 7.1 shows the joint group character classes which consist of the characters in different forms (isolated, start, middle and end) that have identical shapes with the only characteristic that can be used to differentiate between them being the number, location and appearance of dots. If we consider all characters in the Arabic alphabet and LAM ALEF and HAMZA in the four different forms which are isolated, start, middle and end forms, we end up with 103 characters (as discussed in chapter 2) that need to be used for training (AbdelRaouf, 2012). However, by grouping the identical characters which differ in the number, location and appearance of dots into one class (such as characters like ب, ت, and ث which have been grouped into one class), we will have only 20 classes. Therefore, only 20 classes of characters have to be considered for training. This means that the total number of training models will be 20 instead of 103.

Table 7.1: Combining Arabic characters in character classes, grouped according to their main type of shape

Class number	Character name in English	Class Character	Characters grouped together into the same class
1	ALEF	ا	ا ـا
2	BEH	ب	ب بـ ـبـ ـب ت تـ ـتـ ـت ث ثـ ـثـ ـث ـب لـ ـلـ نـ ـنـ يـ ـيـ
3	HAH	ح	ح حـ ـحـ ـح خ خـ ـخـ ـخ ج ـج جـ ـجـ
4	DAL	د	د ـد ذ ـذ
5	REH	ر	ر ـر ز ـز
6	SEEN	س	س سـ ـسـ ـس ش ـش ـشـ شـ
7	SAD	ص	ص صـ ـصـ ـص ض ضـ ـضـ ـض
8	TAH	ط	ط ـط طـ ـطـ
9	AIN	ع	ع عـ ـعـ ـع
10	FEH	ف	ف فـ ـفـ ـف قـ ـقـ
11	QAF	ق	ق ـق
12	KAF	ك	ك كـ ـكـ ـك
13	LAM	ل	ل ـل
14	MEEM	م	م مـ ـمـ ـم
15	NOON	ن	ن ـن
16	HEH	ه	ه هـ ـهـ ـه
17	WAW	و	و ـو
18	YEH	ي	ي ـي
19	LAM ALEF	لا	لا ـلا
20	HAMZA	ء	ء

Considering the table above, for each specific font, 60 character images of each character class have been selected from the PAC-NFB with the intention to cover all font styles and sizes that are considered in this study. 70% of the selected images were used as a training set for constructing the PPM model and 30% were used for testing in stage II which will be discussed in section 7.4. Tables 7.2 and 7.3 show an example of training data used for stage I using the character classes of the Traditional Arabic font.

The PATRION OCR system in this work uses characters (isolated characters) as models. That is, each character model is constructed by passing the training set of a specific class from a specific font through the thinning stage and feature extraction stage, explained in Chapters 4 and 5 respectively, then the output of the feature extraction which is the feature strings are used to train the corresponding PPM model. For instance, the character model of the Traditional Arabic font for class 1 is trained on all characters that appear in class 1 from different font styles and pitch sizes. This results in having 20 character models for each font. For illustration, Table 7.4 shows three examples of the feature strings for four different character classes. In these examples, only ten character images have been used for each of the four classes. The table shows that there are only three distinct feature strings for each character class.

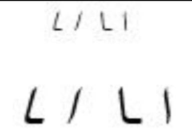
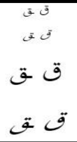
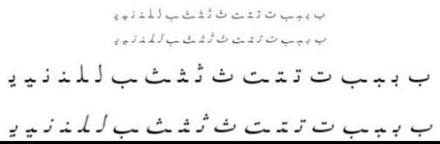
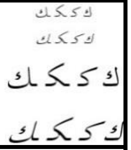
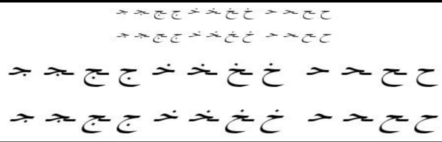
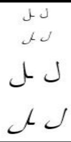
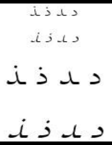
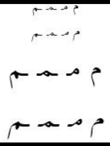
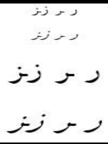
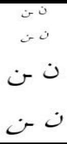
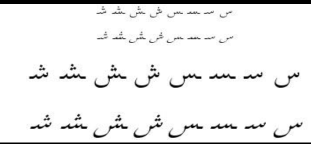
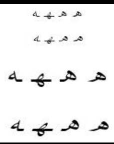
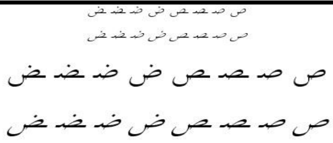
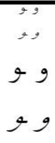
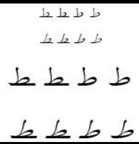
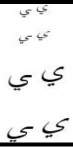
Class number	Class name	Character image	Class number	Class name	Character image
1	ALEF		11	QAF	
2	BEH		12	KAF	
3	HAH		13	LAM	
4	DAL		14	MEEM	
5	REH		15	NOON	
6	SEEN		16	HEH	
7	SAD		17	WAW	
8	TAH		18	YEH	

Table 7.2: Training data for Stage I

Note: A single font (Traditional Arabic font) is shown.

Continued onto next page

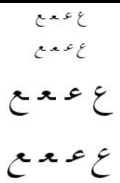
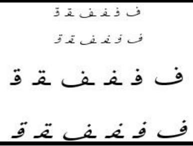
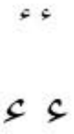
9	AIN		19	LAM ALEF	
10	FEH		20	HAMZA	

Table 7.3: Continued. Training data for stage I

Note: A single font (Traditional Arabic font) is shown.

Class name	Feature strings
HAMZA	SSENWWEN*
	SSENW*WSN*
	SSENW*WEN*
LAM ALEF	ESS*SN*N
	ESEWNWN*N
	ESEWNWN
BEH	WN*
	NWN*
	WEN*
LAM	NWSWN*
	NWSN
	NWN*

Table 7.4: Examples of feature strings for character classes in Tables 7.2 and 7.3

Note: The feature strings were extracted by running the algorithm discussed in chapter 5 on a sample of the Traditional Arabic font dataset.

The feature string produced by the feature extraction method is irreversible, as stated previously. Moreover, the feature algorithm uses only four directions: north; east; west; and south. Consequently, an issue might occur when there is a case of an edge that goes roughly north-west, south-west, north-east and north-west or even goes to secondary intermediate directions such as NNE, ENE, ESE, SSE, SSW, WSW, WNW and NNW. In particular, the mapping from skeleton images to feature strings can be at risk of labelling similar images as being different. In order to show that these problems are not major issues for the proposed OCR system, we compare the feature strings that have been extracted from twenty character images from all the twenty character classes, mentioned above, using edit distance which was discussed in section 5.4.3. Note that only Arabic characters in an isolated form have been considered in this comparison. The character images have been selected from the PAC-NFB dataset of Simplified Arabic font. The high numbers of edit distance mean that the feature string is very different. Table 7.5 shows the confusion matrix using edit distance. Table 7.5 shows that only some characters that are visually similar to each other having less edit distance, such as the similarity between the two letters ل and ن and between the letters ك, د and ب.

Actual class	**Shape**	**Predicted Class**																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
		ا	ب	ح	د	ر	س	ص	ط	ع	ف	ق	ك	ل	م	ن	ه	و	ي	لا	ء
1	ا	**0**	3	7	3	2	10	9	7	6	5	8	3	3	4	3	4	6	13	5	8
2	ب	3	**0**	5	0	3	7	6	4	5	2	5	0	2	3	2	3	4	11	4	6
3	ح	7	5	**0**	5	5	7	9	6	4	4	5	5	5	4	5	3	4	9	6	7
4	د	3	0	5	**0**	3	7	6	4	5	2	5	0	2	2	2	2	4	11	4	5
5	ر	2	3	5	3	**0**	8	7	6	5	3	6	3	1	3	1	3	4	11	4	7
6	س	10	7	7	7	8	**0**	6	6	7	5	4	7	7	6	7	7	5	8	7	6
7	ص	9	6	9	6	7	6	**0**	6	7	7	7	6	6	7	6	7	6	10	6	8
8	ط	7	4	6	4	6	6	6	**0**	7	3	6	4	5	4	5	5	4	9	4	6
9	ع	6	5	4	5	5	7	7	7	**0**	5	5	5	5	5	5	3	4	10	6	5
10	ف	5	2	4	2	3	5	7	3	5	**0**	3	2	2	2	2	2	3	10	4	5
11	ق	8	5	5	5	6	4	7	6	5	3	**0**	5	5	5	5	4	2	9	7	6
12	ك	3	0	5	0	3	7	6	4	5	2	5	**0**	2	2	2	2	4	11	4	5
13	ل	3	2	5	2	1	7	6	5	5	2	5	2	**0**	3	0	4	5	10	3	6
14	م	4	3	3	2	3	7	7	5	3	2	4	2	4	**0**	3	2	3	10	5	4
15	ن	3	2	5	2	1	7	6	5	5	2	5	2	0	3	**0**	4	5	10	3	6
16	ه	4	3	3	2	3	7	7	5	3	2	4	2	4	2	4	**0**	3	11	5	5
17	و	6	4	4	4	4	5	6	4	4	3	2	4	5	3	5	3	**0**	8	6	6
18	ي	13	11	9	11	11	8	10	9	10	10	9	11	10	10	10	11	8	**0**	9	7
19	لا	5	4	6	4	4	7	6	4	6	4	7	4	3	5	3	5	6	9	**0**	6
20	ء	8	6	7	5	7	6	8	6	5	5	6	5	6	4	6	5	6	7	6	**0**

Table 7.5: Confusion matrix for Arabic character of Simplified font-type using edit-distance.

7.2.2 Stage II: Testing/recognition stage

During the recognition stage, thinning, feature extraction and segmentation algorithms are applied to the input text images to obtain a character's feature string (see Figure 7.1 at the beginning of the chapter). Then, the character's feature string being recognized is compressed using the PPM pre-trained models, and the character class is chosen from the model used for training that compresses the feature string best, as described in chapter 2, section 2.3.4.1. In particular, the character's feature string being recognized will obtain the lowest codelength when it is compressed using the model that has been trained on the feature strings of this character.

Table 7.6 shows how this works for a sample of feature strings taken from five characters which are HEH, WAW, YEH, LAM ALEF and HAMZA. Twenty character feature strings were obtain from each character. A tested feature string of each character were compressed using order 5 PPM models trained on the remaining feature strings for the five selected character. The best performed model for each character's feature strings are highlighted in bold. In all cases, the best performed model is associated with the correct character of each feature string.

Tested feature strings	Training feature strings				
	HEH	WAW	YEH	LAM ALEF	HAMZA
HEH	**12.052**	22.562	53.764	31.466	33.510
WAW	14.959	**16.333**	52.302	31.218	40.782
YEH	22.362	30.117	**15.317**	38.224	35.042
LAM ALEF	12.766	21.676	55.084	**12.064**	43.801
HAMZA	22.896	32.315	41.766	41.351	**11.142**

Table 7.6: Recognising the feature strings for five Arabic characters

Note: compression codelengths (in bits) for the character's feature strings

7.3 Experimental Evaluation Results

In this section, two testing experiments have been conducted in order to recognize Arabic text images and evaluate the performance of the PPM classifier. In the first experiment, the classifier was evaluated on a pre-segmented character images dataset. In the second experiment, the classifier was evaluated on a paragraph-based text images dataset. In the following sub-sections, the testing procedure will be explained for both experiments.

7.3.1 Pre-segmented character images evaluation experiment

From the character dataset used in the training stage, 30% of the selected images were selected as a testing set. For this experiment, the testing dataset follows the same procedure used for the training stage. In particular, the two processing steps, thinning and feature extraction, were applied to each image. This resulted in feature strings that were then passed to the PPM classifier. Note that the segmentation method was not applied on this set, since the dataset consists of pre-segmented characters.

A number of performance metrics have been used to evaluate the accuracy of classification methods. In order to evaluate the PPM classifier, three of the most common evaluation metrics have been used in this experiment: recall, precision and F-measure (Witten et al., 2005). These metrics can be calculated using a confusion matrix, which is used for visualizing the performance of the classification techniques, as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{7.1}$$

$$Recall = \frac{TP}{TP+FN} \tag{7.2}$$

$$Precision = \frac{TP}{TP+FP} \tag{7.3}$$

$$F - measure = \frac{2 \times Precision \times Recall}{Precision+Recall} \tag{7.4}$$

where TP (True Positives) is the number of cases where the prediction correctly classified and TN (True Negatives) is the number of cases where the prediction correctly classified as negative. FP (False Positives) is the number of cases where the prediction incorrectly classified as positive and FN (False Negatives) is the number of cases where the prediction is incorrectly classified as negative.

Confusion matrices that resulted from applying PPM to classify the pre-segmented datasets for Traditional Arabic font, Arabic Transparent font, Simplified Arabic font and AdvertisingBold font are shown in Table 7.7, Table 7.9, Table 7.11, and Table 7.13 respectively. The values in bold font represent the number of correct classifications made by the PPM classifier.

It can be seen from Table 7.7 that a major confusion made by the PPM classifier is between classes 2, 13 and 15, which are the BEH, LAAM and NOON character classes respectively. This misclassification seems a reasonable error to make, since most characters in class number 13 and class number 15 of the Traditional font dataset have identical shapes which results in them sharing the same feature strings. In addition, some characters from class number 13 have been misclassified as characters from class number 2 by PPM. These misclassifications have occurred as the LAAM character in the middle form, which is a part of class number 13, and the BEH characters in middle form have an identical feature. Table 7.8 shows the accuracy, precision, recall and F-measure values that are achieved by the PPM classifier when classifying the Traditional Arabic font dataset. The bottom row displays an overall average for accuracy, precision, recall and F-measure. PPM classified the Traditional Arabic font and AdvertisingBold font with an average accuracy of 0.99, an average precision, recall and F-measure of 0.93, as shown in Tables 7.8 and 7.14.

Remarkably, PPM performs better when classifying characters from the Simplified Arabic font dataset in class 2 and 13, as shown in Table 7.11. The increase of the number of the corrected classifications made by the PPM for classifying Simplified Arabic characters in class 13 is due to the LAAM characters in the middle form having a dissimilar shape to the BEH character in the Simplified Arabic font. Thus, the PPM classifier produced an average accuracy rate of 0.99 and average rates for all precision, recall and F-measure of 0.95, as shown in Table 7.12.

Actual class	**Predicted Class**																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	**17**	0	0	0	0	0	0	0	1	0	0	2	0	0	0	0	0	0	0
3	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	**18**	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0
5	0	0	0	0	**17**	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0
6	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	**19**	0	0	0	0	0	0	0	1	0	0	0
10	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	2	0	0	**18**	0	0	0	0	0	0	0	0
13	1	4	0	2	0	0	0	0	0	0	0	0	**12**	0	1	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0
15	0	0	0	0	2	0	0	0	0	0	0	0	3	0	**15**	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0
17	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	**18**	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**

Table 7.7: Confusion matrix for Traditional font-type classification using the 20 classes

Class Number	Accuracy	Precision	Recall	F-measure
1	0.99	0.95	1.00	0.97
2	0.98	0.80	0.85	0.82
3	1.00	1.00	1.00	1.00
4	0.98	0.90	0.90	0.90
5	0.98	0.80	0.85	0.82
6	1.00	1.00	1.00	1.00
7	1.00	1.00	1.00	1.00
8	1.00	1.00	1.00	1.00
9	0.99	0.90	0.95	0.92
10	0.99	0.95	1.00	0.97
11	1.00	1.00	1.00	1.00
12	0.98	0.90	0.90	0.90
13	0.95	0.60	0.60	0.60
14	1.00	1.00	1.00	1.00
15	0.98	0.93	0.75	0.83
16	1.00	1.00	1.00	1.00
17	0.99	0.94	0.90	0.92
18	1.00	1.00	1.00	1.00
19	1.00	1.00	1.00	1.00
20	1.00	1.00	1.00	1.00
Average	**0.99**	**0.93**	**0.93**	**0.93**

Table 7.8: PPM classification results for the Traditional Arabic font type

Actual Predicted	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	**17**	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
3	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	1	0	**19**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	**18**	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
6	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0
9	0	2	0	0	0	0	0	0	**17**	0	0	0	0	0	0	0	1	0	0	0
10	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0
12	0	2	0	0	0	0	0	0	0	0	0	**17**	1	0	0	0	0	0	0	0
13	0	3	0	0	0	0	0	0	0	0	0	1	**14**	0	2	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	1	0	0	**19**	0	0	0	0	0	0
15	0	0	0	0	1	0	0	0	0	0	0	0	2	0	**17**	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	**19**	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**

Table 7.9: Confusion matrix for the Transparent font-type classification using the 20 classes

Class Number	Accuracy	Precision	Recall	F-measure
1	0.99	0.95	1.00	0.97
2	0.97	0.68	0.85	0.75
3	1.00	1.00	1.00	1.00
4	0.99	1.00	0.95	0.97
5	0.99	0.94	0.90	0.92
6	1.00	1.00	1.00	1.00
7	1.00	1.00	1.00	1.00
8	1.00	1.00	1.00	1.00
9	0.99	1.00	0.85	0.91
10	1.00	1.00	1.00	1.00
11	0.99	0.95	1.00	0.97
12	0.98	0.94	0.85	0.89
13	0.96	0.70	0.70	0.70
14	0.99	0.95	0.95	0.95
15	0.98	0.94	0.85	0.89
16	0.99	1.00	0.95	0.97
17	0.99	0.95	1.00	0.97
18	1.00	1.00	1.00	1.00
19	1.00	1.00	1.00	1.00
20	1.00	1.00	1.00	1.00
Average	**0.99**	**0.95**	**0.94**	**0.94**

Table 7.10: PPM classification results for the Transparent Arabic font type

Actual predicted	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	**18**	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
3	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	1	0	**17**	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0
5	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	**19**	0	0	0	0	0	0	0	0	1	0	0	0	0	0
7	0	0	0	0	0	0	**19**	1	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0	**18**	1	0	0	0	0	0	0	1	0	0	0
10	0	0	0	0	0	0	0	0	1	**19**	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	**18**	2	0	0	0	0	0	0	0
13	0	3	0	0	0	0	0	0	0	0	0	0	**16**	0	3	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	1	0	**19**	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0	0
17	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	**19**	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**

Table 7.11: Confusion matrix for the Simplified Arabic font-type classification using the 20 classes

Class Number	Accuracy	Precision	Recall	F-measure
1	1.00	1.00	1.00	1.00
2	0.98	0.78	0.90	0.83
3	1.00	1.00	1.00	1.00
4	0.98	0.94	0.85	0.89
5	0.99	0.95	1.00	0.97
6	0.99	1.00	0.95	0.97
7	0.99	1.00	0.95	0.97
8	0.99	0.95	1.00	0.97
9	0.99	0.94	0.90	0.92
10	0.99	1.00	0.95	0.97
11	1.00	1.00	1.00	1.00
12	0.98	0.90	0.90	0.90
13	0.97	0.80	0.80	0.80
14	1.00	1.00	1.00	1.00
15	0.98	0.82	0.95	0.88
16	1.00	1.00	1.00	1.00
17	0.99	0.95	0.95	0.95
18	1.00	1.00	1.00	1.00
19	1.00	1.00	1.00	1.00
20	1.00	1.00	1.00	1.00
Average	**0.99**	**0.95**	**0.95**	**0.95**

Table 7.12: PPM classification results for the Simplified Arabic font type

Actual class	**Predicted Class**																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	**16**	0	1	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0
3	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	1	0	**19**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	**19**	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
6	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	1	0	0	0	0	0
7	0	0	0	0	0	0	**20**	1	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0	**20**	0	0	0	0	0	0	0	0	0	0	0
10	0	2	0	0	0	0	0	0	0	**18**	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	**19**	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	**18**	1	0	1	0	0	0	0	0
13	0	6	0	0	1	0	0	0	0	0	0	0	**10**	0	3	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	2	0	0	**18**	0	0	0	0	0	0
15	0	0	0	0	2	0	0	0	0	0	0	0	0	0	**18**	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	**19**	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	**19**	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**20**

Table 7.13: Confusion matrix for the AdvertisingBold font type classification using the 20 classes

Class Number	Accuracy	Precision	Recall	F-measure
1	1.00	1.00	1.00	1.00
2	0.96	0.61	0.80	0.69
3	1.00	1.00	1.00	1.00
4	0.99	0.95	0.95	0.95
5	0.98	0.86	0.95	0.90
6	1.00	1.00	1.00	1.00
7	1.00	1.00	1.00	1.00
8	0.99	0.95	1.00	0.97
9	1.00	1.00	1.00	1.00
10	0.99	1.00	0.90	0.94
11	0.99	0.90	0.95	0.92
12	0.99	1.00	0.90	0.94
13	0.96	0.66	0.50	0.57
14	0.99	0.94	0.90	0.92
15	0.97	0.75	0.90	0.81
16	0.99	1.00	0.95	0.97
17	1.00	1.00	1.00	1.00
18	0.99	1.00	0.95	0.97
19	1.00	1.00	1.00	1.00
20	1.00	1.00	1.00	1.00
Average	**0.99**	**0.93**	**0.93**	**0.93**

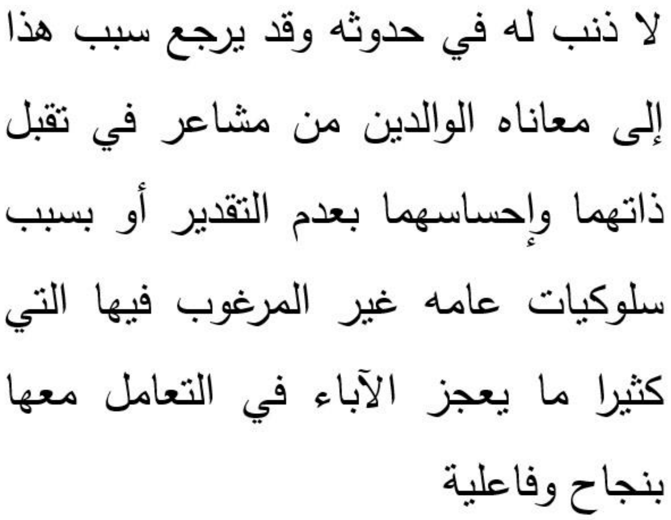
Table 7.14: PPM classification results for the AdvertisingBold font type

7.3.2 Paragraph-based text images evaluation experiment

The objective of this experiment described in this section is to assess recognition performance (combining thinning, feature extraction, segmentation and classification stages) of the PATRION OCR system using the PPM as a classifier on real text images. Thus, the edit distance evaluation metric has been used in this experiment (see chapter 3, section 3.2). The dataset which was used for this evaluation is the same dataset used for the evaluation study in chapter 3. However, Arabic Transparent, AdvertisingBold, Simplified Arabic and Traditional Arabic are the font types which have been investigated in this recognition study with text images at 300dpi. Recall that the text images have been selected from a publicly available dataset (KAFD dataset). The test set contains 16 text images from the four font types (Arabic Transparent, AdvertisingBold, Simplified Arabic and Traditional Arabic) at 300dpi. For each font, 12 and 24 pitch sizes have been selected. Additionally, normal and italic font styles for each font pitch size have been used. Note that as recognition of numbers and punctuation marks is out of the scope of this study, numbers or punctuation marks have been ignored in the classification stage.

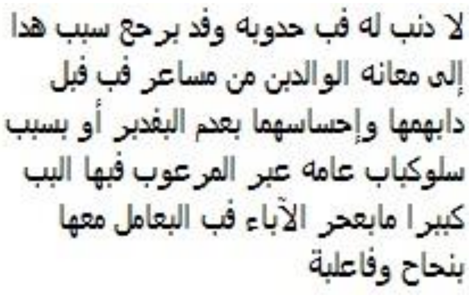
The test text images were passed through the thinning, feature extraction and segmentation stages. Then, the output of the segmentation stage which is the feature string for each character has been passed to the PPM classifier to perform the recognition task using the twenty character models.

Figure 7.7 (a) shows a sample of the text image test set. The corresponding ground-truth which is available with the dataset (see Figure 7.2 (b)) has been modified in order that they can be matched with the output of the classifier. In particular, all characters have been replaced by their corresponding class characters as shown in Table 7.1. For example, characters like ج and خ in the ground-truth have been replaced by their class character which is the character (ح). Running this image through the PATRION Arabic OCR system produced the text shown in Figure 7.3.



لا ذنب له في حدوثه وقد يرجع سبب هذا
إلى معاناه الوالدين من مشاعر في تقبل
ذاتهما وإحساسهما بعدم التقدير أو بسبب
سلوكيات عامه غير المرغوب فيها التي
كثيرا ما يعجز الآباء في التعامل معها
بنجاح وفاعلية

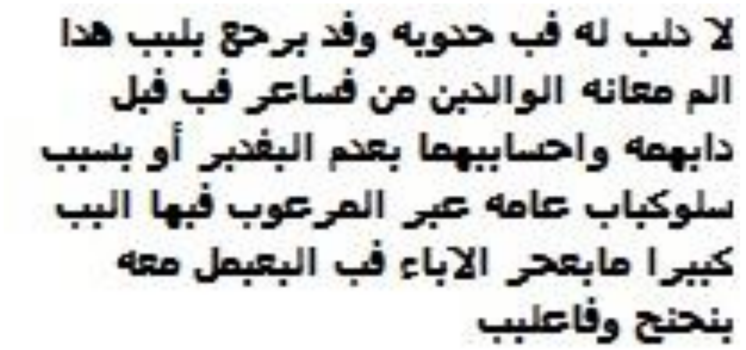
(a)



لا دنب له فب حدوبه وفد برحع سبب هدا
إلى معانه الوالدبن من مساعر فب فبل
دابهمها وإحساسهما بعدم البفدبر أو بسبب
سلوكباب عامه عبر المرعوب فبها البب
كببرا مابعحر الآباء فب البعامل معها
بنحاح وفاعلبة

(b)

Figure 7.2: Text image example used in the recognition/testing stage
Note: (a) An Arabic text image from the KFAD dataset and (b) the modified corresponding ground truth



لا دلب له فب حنويه وفد برحع بلبب هدا
الم معانه الوالدبن من فساعر فب فبل
دابهمه واحسابيهما بعدم البفدبر أو بسبب
سلوكباب عامه عبر المرعوب فبها البب
كببرا مابعحر الاباء فب البعبمل معه
بنحنح وفاعلبب

Figure 7.3: PATRION OCR output text produced for image in Figure 7.2

Table 7.15 shows the recognition accuracy for each font. As can be seen from the table, the PATRION OCR system achieves an average accuracy rate of 77.3%. The recognition results indicate that the system has a higher accuracy in recognising Simplified Arabic font with 81.5% accuracy. In terms of font style, our experiment finds that the PATRION OCR system obtains an accuracy rate of 78.5% and 76.1% for recognizing a normal font style and an italic font style respectively, as Table 7.16 shows. In addition, the respective recognition accuracy of the proposed system in relation to different font pitches sizes is presented in Table 7.17.

The possible explanation for the lower accuracy in this experiment is as follows. Recall that the connectivity of Arabic text is an obstacle of Arabic text recognition. Also, as has been mentioned in Chapter 5 that character segmentation in Arabic text recognition is a critical step as most of the recognition errors are produced by errors in character segmentation. This is supported by the experiment results, obtained from both evaluation experiments, which indicate that the accuracy performance rates of the PPM classifier are higher in classifying pre-segmented Arabic characters than in classifying characters from paragraph-based text images.

Font type	**Accuracy**
Arabic Transparent	74.7%
Traditional Arabic	77.5%
Simplified Arabic	81.5%
AdvertisingBold	75.5%
Average	**77.3%**

Table 7.15 Recognition accuracy results on different Arabic fonts

Font style	**Accuracy**
Normal	78.5%
Italic	76.1%

Table 7.16: Italic and normal font recognition accuracy results

Font pitch size	**Accuracy**
24	79.7%
12	74.9%

Table 7.17: Recognition accuracy results based on font pitch size

7.4 Discussion and Summary

In this chapter, a compression-based model has been applied for the first time to Arabic text recognition. The implementation of a compression based model has been explained. Two classification experiments were conducted in order to classify and recognise isolated Arabic character images and Arabic text images using the PPM models.

The experimental evaluation study in this chapter has analysesd the performance of the PPM classifier and underlined the major misclassifications that were produced, such as the misclassification between the LAAM character, the BEH character and the NOON character. These major misclassification errors that were made by the PPM classifier will be examined when improving the overall accuracy of the PATRION OCR system in the next chapter.

Chapter 8: Post-processing

8.1 Chapter Introduction

In the previous chapter, the new PATRION system, which was developed for this research, recognised non-dotted Arabic text images. The PATRION system mis-recognised some Arabic characters. In this chapter a post-processing stage, which is the final stage in the PATRION OCR system (see Figure 8.1), has been implemented for improving the OCR output by correcting the mis-recognition errors. Recognizing dots in Arabic text has been ignored in all the previous stages due to its challenge. However, this chapter aims to overcome the challenge of recognizing dotted characters in Arabic text by using a post-processing method to recover unrecognised dots.

In this chapter, a post-processing technique based on the PPM compression scheme, which is discussed in the review chapter, has been applied to two specific problems: adding dots to Arabic text and correcting the OCR output. Section 8.2 shows how the PPM language model can be used as a post-processing stage for the Arabic OCR system. Then, section 8.3 discusses experimental results using the PPM based correction method. Section 8.4 compares the performance of the PATRION system with others reported in chapter 3. A summary and discussion of this chapter is presented in section 8.5.

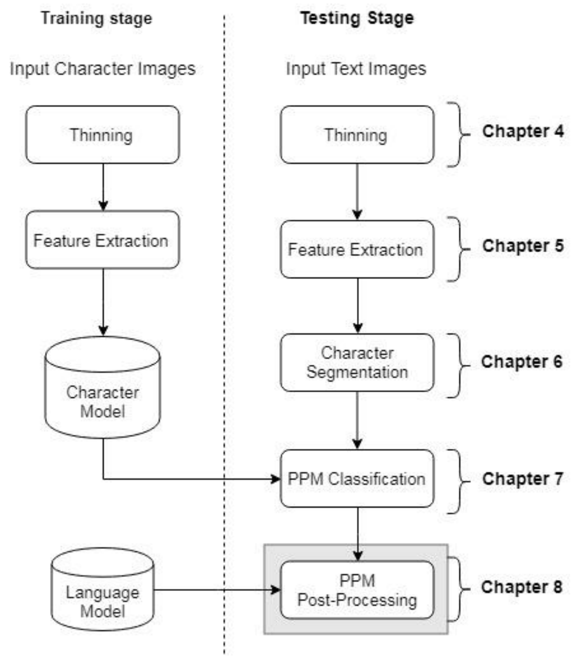


Figure 8.1: Block diagram of the PATRION OCR system
Note: The shaded area shows the scope of the current chapter ‘Post-processing’.

8.2 Applying a PPM Language Model for Arabic Text Correction

The PPM correction method uses a language model to determine the likelihood of a possible correct character using the Viterbi algorithm (Viterbi, 1967). For constructing a language model, a training corpus of Arabic text is required. Due to the limited availability and accessibility of Arabic corpora, the Bangor Arabic Compression Corpus (BACC) published by Alhawiti (2014), which contains about 31-millon Arabic words collected from several sources, has been selected. The PPM model was applied to the problem of OCR text correction using the approach used by Teahan et al. (1998) for correcting English OCR texts and by Alhawiti (2014) for correcting Arabic OCR texts. However, the main limitation of Alhawiti’s Arabic correction study is that the PPM method was only able to correct one-to one Arabic character errors. To overcome this problem, the Buckwalter text processing technique, as used by Alkhazi & Teahan (2017), has been first applied to convert Arabic characters to equivalent Latin ASCII characters. Then, an order 5 PPMD model implemented by the Tawa Toolkit described by Teahan (2018) trained on the BACC Corpus has been used for constructing the language model.

Teahan et al. (1998) define ‘confusion’ as a transformation that is used to correct characters in the text. They use $observed \rightarrow correct$ to denote a confusion transformation from the observed text to the correct text. For the PATRION system, the mis-recognitions produced in the previous chapter have been used as the confusions. Table 8.1 shows a set of confusions that have been mostly generated by the PATRION OCR. Also, Table 8.1 contains a set of confusions that is used for adding dots to Arabic characters (correcting dots characters). For example to correct the dot character ج, ح→ ج denotes the character ح corrected to the character ج. On the other hand, Arabic script has some letters that contains Hamza which is placed above or below the letter, such as the ALEF character. Hamza that is placed above or below the ALEF has been ignored in the previous stages as it is not a part of the main body of the character. Thus, we also include a confusion for correcting Hamza in the ALEF character.

Confusion			
Arabic Character	Equivalent Buckwalter	Arabic Character	Equivalent Buckwalter
ـب→ ل	i→T	س ـب→ ل	iT→I
ـل→ ن	T→U	ـب→ ي	i→A
ن→ ل	U→T	ـف→ م	W→Y
ـب→ ا	i→u	ا→ أ	u→e
ـب→ ت	i→p	ـب→ ي	i→A
ح→ ج	d→s	د→ ذ	g→h
ر→ ز	j→k	ـص→ ض	x→c
ط→ ظ	v →b	ـف→ ق	W→E
ى→ ي	P→A	ن ـب→ س	Ui→I
ـب→ ث	i→a	ع→ غ	n→m
ح→ خ	d→f	ـم→ ى	Y→P
س→ ش	I→z	ا→ إ	u→t
ر→ و	j→O	ه→ ة	I→o

Table 8.5: Set of confusions used for correcting the PATRION OCR output

8.3 Experimental Evaluation of PPM Correction Method

In this section, two experiments have been conducted for an investigation into the effectiveness of using the PPM based correction method for correcting the Arabic OCR output. In the first experiment, the PPM language model was applied to the PATRION OCR system in order to correct and add dots to the output text produced by the system. In the second experiment, we applied the PPM correction method to a commercial Arabic OCR system, in an attempt to improve OCR output to see if the correction method also applies to other Arabic OCR systems.

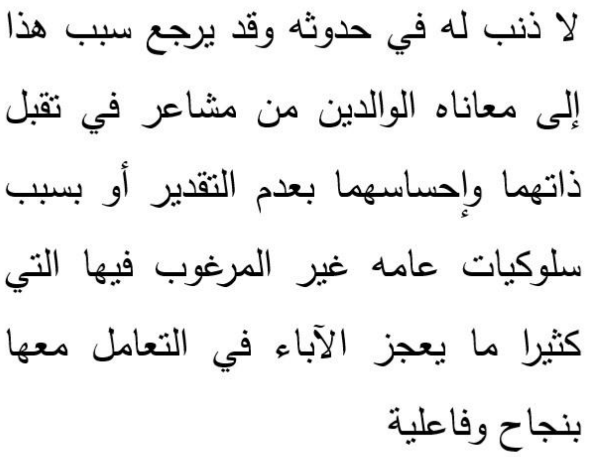
The image dataset used in both experiments is the KAFD dataset used in the previous chapter which contains 16 text images from the four font types (Arabic Transparent, AdvertisingBold, Simplified Arabic and Traditional Arabic) at 300dpi. The PPM based correction method has been evaluated by comparing the processed text with the original text (ground-truth). The difference between the two texts was obtained by using the edit distance implemented in the automated evaluation tool described in chapter 3. In the following sub-sections, the evaluation procedure will be explained for both experiments.

8.3.1 PATRION OCR output correction experiment

In this experiment, the PPM correction method was applied to the PATRION OCR system at the post-processing stage. A sample of the output text produced by the PATRION OCR system for an Arabic text image is shown in Figure 8.2. As can be seen from the output text, some characters are required to be dotted. For instance, د in the word هدا and ح in the word برحع (line 1 in Figure 8.2 (b)) are not dotted. These should be dotted as ذ, and ج respectively. Also, the OCR system produces some recognition errors, such as الم and احسابيهما which should be written as إلى and إحساسهما.

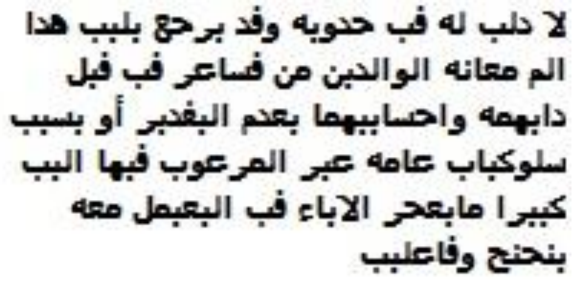
Sample texts after applying the correction method for the OCR output texts of different font types of text images are shown in Figures 8.2, 8.3, 8.4 and 8.5. Considering the corrected output texts, the text has been correctly dotted by the PPM correction method. For example, the dot corrections of فد to قد , هدا to هذا (line 1 in Figure 8.2), فرب to قرب (line 1 in Figure 8.3), فى to في (line 3 in Figure 8.3), بعسره to بعشرة (line 1 in Figure 8.4), and بسكل to بشكل (line 1 in Figure 8.5) stem from single character confusions, whereas the correction of بلبب to سبب (line 1 in Figure 8.2), البلفن to السفن (line 3 in Figure 8.3) derives from double character confusions which is ل ب → س. Also, the PPM correction method successfully corrected multiple errors that occur in some words. For example, the corrections برحع to يرجع (line 1 in Figure 8.2), حرب to جرت (line 3 in Figure 8.3) and الرعابه to الرعاية (line 3 in Figure 8.5) are multiple character corrections in one word. Interestingly, Hamza errors have been successfully corrected by the PPM method, such as and ا → إ in the word إلى (line 2 in Figure 8.2) and ا → أ in the word أجزاء (line 2 in Figure 8.4).

In the previous chapter, it is stated that the overall accuracy of the PATRION OCR system so far is 77.3%. However, by applying the post-processing method on the 16 text images used for testing mentioned in section 8.3, the recognition accuracy improves to 86.9%.



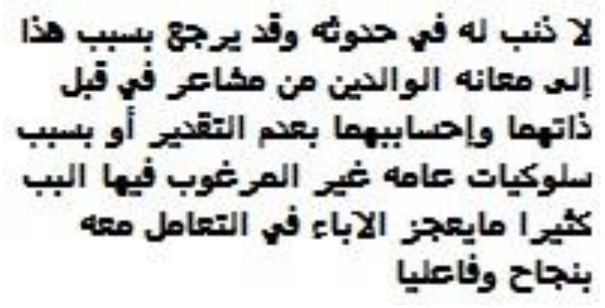
لا ذنب له في حدوثه وقد يرجع سبب هذا
إلى معاناه الوالدين من مشاعر في تقبل
ذاتهما وإحساسهما بعدم التقدير أو بسبب
سلوكيات عامه غير المرغوب فيها التي
كثيرا ما يعجز الآباء في التعامل معها
بنجاح وفاعلية

(a)



لا دلب له فب حدويه وفد برحع بلبب هدا
الم معانه الوالدبن من فساعر فب فبل
دابهمه واحسابيهما بعدم البفدبر أو بسبب
سلوكباب عامه عبر المرعوب فبها البب
كبيرا مابعحر الاباء فب البعبمل معه
بنحنح وفاعلبب

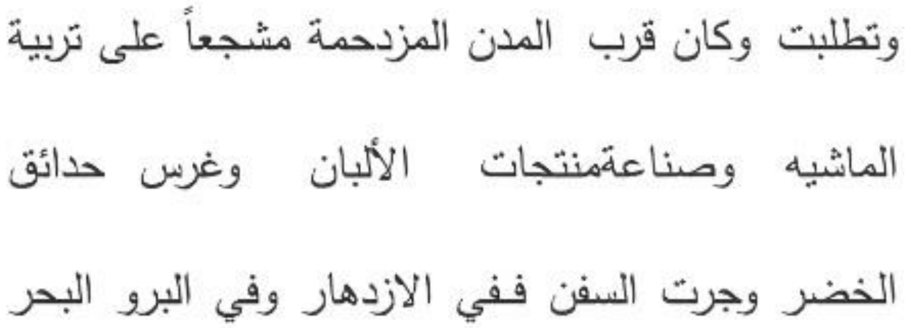
(b)



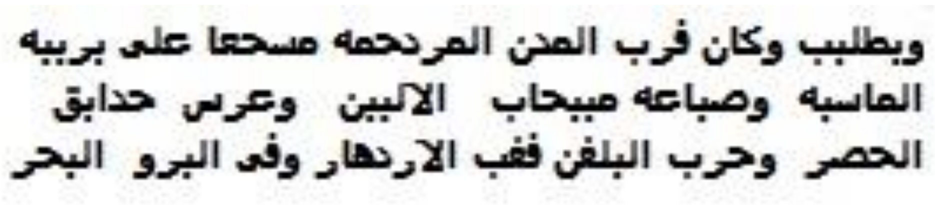
لا ذنب له في حدوثه وقد يرجع بسبب هذا
إلى معانه الوالدين من مشاعر في قبل
ذاتهما وإحسابيهما بعدم التقدير أو بسبب
سلوكيات عامه غير المرغوب فيها البب
كثيرا مايعجز الاباء في التعامل معه
بنجاح وفاعليا

(c)

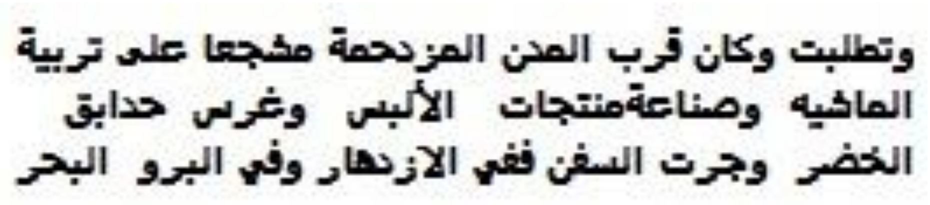
Figure 8.2: PPM correction of the PATRION OCR output text
Note: (a) An Arabic text image of the Arabic Transparent font from the KFAD dataset, (b) The output text from the proposed OCR system that was produced for the image in (a) and (c) the corrected OCR output text after applying the PPM correction method.


وتطلبت وكان قرب المدن المزدحمة مشجعاً على تربية
الماشيه وصناعةمنتجات الألبان وغرس حدائق
الخضر وجرت السفن فـفي الازدهار وفي البرو البحر

(a)

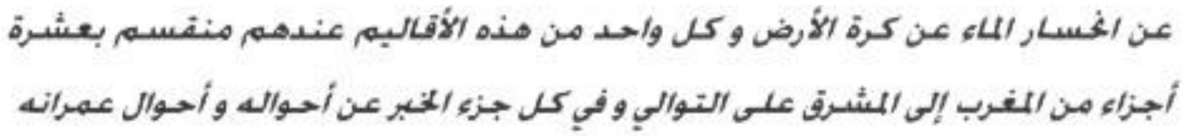

ويطلبب وكان فرب المدن المردحمه مسحعا على بربيه
الماسبه وصباعه مبيحاب الالبين وعرس حدابق
الحصر وحرب البلفن ففب الارنهار وفى البرو البحر

(b)


وتطلبت وكان قرب المدن المزدحمة مشجعا على تربية
الماشيه وصناعةمنتجات الألبس وغرس حدابق
الخضر وجرت السفن ففي الازنهار وفي البرو البحر

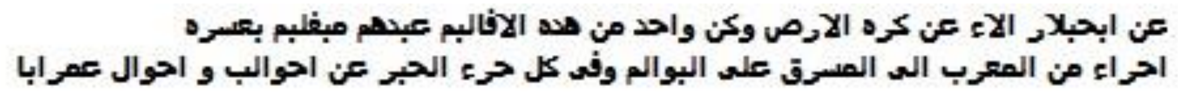
(c)

Figure 8.3: PPM correction of the PATRION OCR output text
Note: (a) An Arabic text image of the Simplified Arabic font from the KFAD dataset, (b) The output text from the proposed OCR system that was produced for the image in (a) and (c) the corrected OCR output text after applying the PPM correction method.



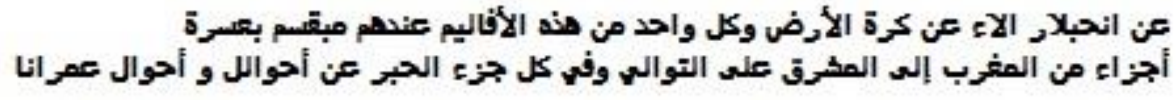
عن انحسار الماء عن كرة الأرض و كل واحد من هذه الأقاليم عندهم منقسم بعشرة
أجزاء من المغرب إلى المشرق على التوالي و في كل جزء الخبر عن أحواله و أحوال عمرانه

(a)



عن ابحبلار الاء عن كره الارص وكن واحد من هده الافالبم عبدهم مبفلبم بعسره
احراء من المعرب الى الصسرق على البوالم وفى كل حرء الحبر عن احوالب و احوال عمرابا

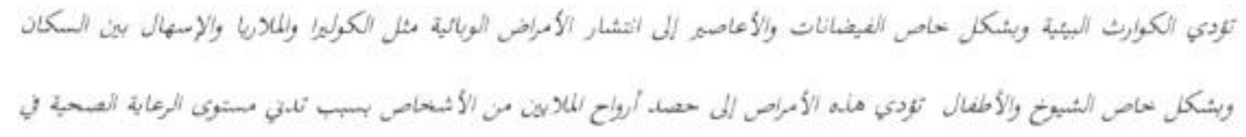
(b)



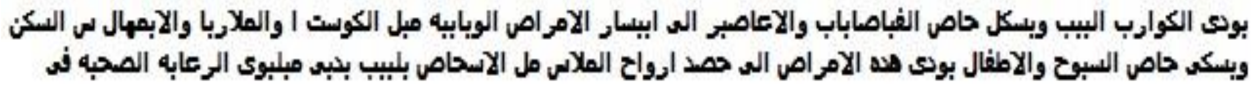
عن انحبلار الاء عن كرة الأرض وكل واحد من هذه الأقاليم عندهم مبقسم بعسرة
أجزاء من المغرب إلى المشرق على التوالي وفي كل جزء الحبر عن أحوالل و أحوال عمرانا

(c)

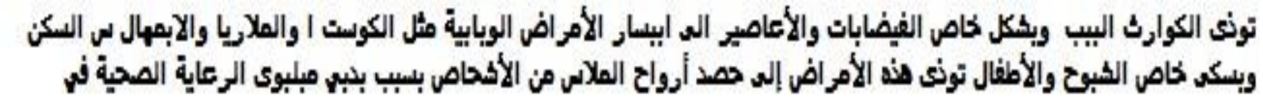
Figure 8.4: PPM correction of the PATRION OCR output text
Note: (a) An Arabic text image of the AdvertisingBold font from the KFAD dataset, (b) the output text from the proposed OCR system that was produced for the image in (a) and (c) the corrected OCR output text after applying the PPM correction method.


تودي الكوارث البيئية وبشكل خاص الفيضانات والأعاصير إلى انتشار الأمراض الوبائية مثل الكوليرا والملاريا والإسهال بين السكان
وبشكل خاص الشيوخ والأطفال تودي هذه الأمراض إلى حصد أرواح الملايين من الأشخاص بسبب تدني مستوى الرعاية الصحية في

(a)


بودى الكوارب البيب ويسكل حاص الفباصاباب والاعاصبر الى ابيسار الامراض الوبايبه مبل الكوست ا والملاريا والابمهال بى السكن
ويسكى حاص السبوح والاطفال بودى هده الامراض الى حصد ارواح الملاس مل الاسحاص بلبيب بدبى مبلبوى الرعابه الصحبه فى

(b)

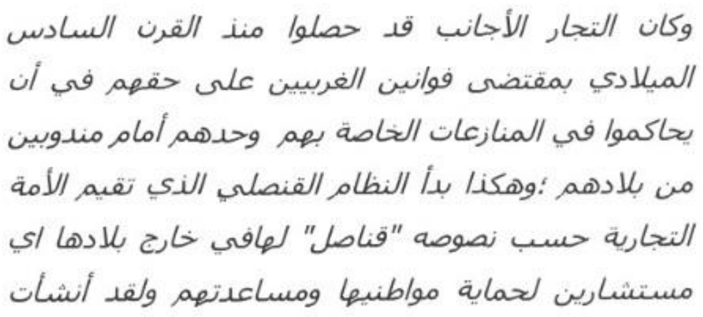

توذى الكوارث البيب ويشكل خاص الفيضابات والأعاصير الى ابيسار الأمراض الوبايية مثل الكوست ا والملاريا والابمهال بى السكن
ويسكى خاص الشبوح والأطفال توذى هذه الأمراض إلى حصد أرواح الملاس من الأشحاص بسبب بذبي مبلبوى الرعاية الصحية في

(c)

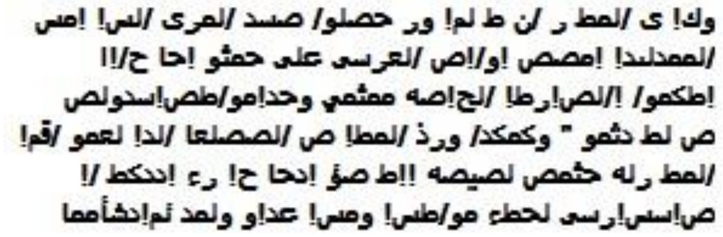
Figure 8.5: PPM correction of the PATRION OCR output text
Note: (a) An Arabic text image of the Traditional Arabic font from the KFAD dataset, (b) The output text from the proposed OCR system that was produced for the image in (a) and (c) the corrected OCR output text after applying the PPM correction method.

8.3.2 Commercial Arabic OCR output correction experiment

In an attempt to evaluate the impact of using the PPM model-based text correction method on the four Arabic OCR systems (Sakhr, ABBYY, RDI and Tesseract) studied in Chapter 3, the method was applied only to the output of the ABBYY commercial OCR system. This choice has been made as other OCR systems produced unreadable output text due to their poor performance. See figure 8.6 as an example of an output text produced by Sakhr OCR which is extremely hard at best and impossible at worst for a human to read.


وكان التجار الأجانب قد حصلوا منذ القرن السادس
الميلادي بمقتضى فوانين الغربيين على حقهم في أن
يحاكموا في المنازعات الخاصة بهم وحدهم أمام مندوبين
من بلادهم ؛وهكذا بدأ النظام القنصلي الذي تقيم الأمة
التجارية حسب نصوصه "قناصل" لهافي خارج بلادها اي
مستشارين لحماية مواطنيها ومساعدتهم ولقد أنشأت

(a)


وك!ا ى /لمط ر /ن ط لم! ور حصلوا/ صسد /لمرى /لس! إمس
/لممدلبدا إمصص إو/إص /لعرسى على حمثو إحا ح/إا
إطكمو/ إ/لص!إرطا /لح!إصه ممثمي وحدإمو/طص!إسدولص
ص لط دثمو " وكمكدا/ ورذ /لمطا ص /لصصلعا /لدا لعمو /قم!
/لمط رله حثمص لصيصه !إط صؤ إدحا ح! رء إددكط /!
ص!إسس!إرسى لحطء مو/طس! وصس! عدإو ولمد تم!إدشأمما

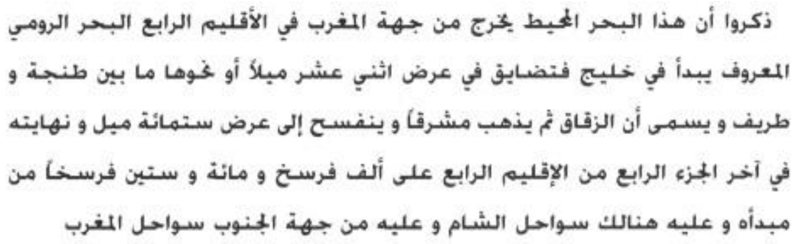
(b)

Figure 8.6: Example of unreadable Sakhr OCR output text
Note: (a) An Arabic text image from the KFAD dataset, and (b) the Sakhr OCR output text that was produced for the image in (a).

Table 8.2 lists a sample of the confusions that were generated when the ABBAYY OCR system was applied to the text images. It is worth mentioning that the most errors made by the ABBYY OCR system are related to the dot characters and Hamza characters. Examples for dot character errors are: the letter ف was incorrectly replaced by ق, the letter ب was incorrectly replaced by ي and the letter ص was replaced by the letter ض. The incorrect transformation of the letter أ to ا is an example of Hamza character error. A sample of ABBYY OCR output for a text image and the correction OCR text is shown in Figure 8.7. The correction of نكروا to ذكروا, ان to أن and المفرب to المغرب (line 1 in the text) all stem from one character confusions which are ن→ ذ, ا→ أ and ف→ غ, while the corrections of خوها to نحوها derives from multiple character confusions. The correction of المفري to المغرب (last line in the text) requires multiple corrections in one word (ف→ غ and ي→ ب). After applying the PPM correction method to the output of the ABBYY OCR system, the recognition accuracy increases from 70.2% to 77.1%.

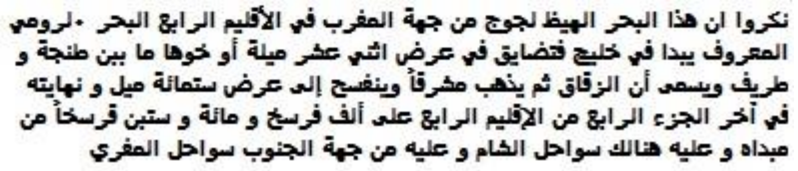
Confusion					
Arabic Character	Equivalent Buckwalter	Arabic Character	Equivalent Buckwalter	Arabic Character	Equivalent Buckwalter
إ→ا	u→t	ن ح→خ	f→Ud	أ→ا	u→e
ذ→ن	U→h	ر→ى	P→j	ض→ص	x→c
غ→ف	W→m	و→ه	I→O	ت→ث	a→p
أن→ت	p→eU	ف→ق	E→W	ر→م	Y→j
ع→غ	m→n	ظ→ط	v→b	ب→ي	A→i

Table 8.6: Sample of the confusions used that were generated from ABBYY OCR output



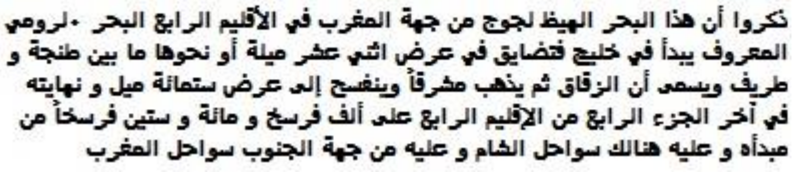
ذكروا أن هذا البحر المحيط يخرج من جهة المغرب في الأقليم الرابع البحر الرومي
المعروف يبدأ في خليج فتضايق في عرض اثني عشر ميلاً أو نحوها ما بين طنجة و
طريف و يسمى أن الزقاق ثم يذهب مشرقاً و ينفسح إلى عرض ستمائة ميل و نهايته
في آخر الجزء الرابع من الإقليم الرابع على ألف فرسخ و مائة و ستين فرسخاً من
مبدأه و عليه هنالك سواحل الشام و عليه من جهة الجنوب سواحل المغرب

(a)



نكروا ان هذا البحر الهيظ لجوج من جهة المفرب في الأقليم الرايع البحر ،لرومي
المعروف يبدا في خليج فتضايق في عرض اثني عشر ميلة أو خوها ما بين طنجة و
طريف ويسمى أن الزقاق ثم يذهب مشرقاً وينفسح إلى عرض ستمائة ميل و نهايته
في آخر الجزء الرابع من الإقليم الرابع على ألف فرسخ و مائة و ستبن فرسخاً من
مبداه و عليه هنالك سواحل الشام و عليه من جهة الجنوب سواحل المفري

(b)



ذكروا أن هذا البحر الهيظ لجوج من جهة المغرب في الأقليم الرابع البحر ،لرومي
المعروف يبدأ في خليج فتضايق في عرض اثني عشر ميلة أو نحوها ما بين طنجة و
طريف ويسمى أن الزقاق ثم يذهب مشرقاً وينفسح إلى عرض ستمائة ميل و نهايته
في آخر الجزء الرابع من الإقليم الرابع على ألف فرسخ و مائة و ستين فرسخاً من
مبدأه و عليه هنالك سواحل الشام و عليه من جهة الجنوب سواحل المغرب

(c)

Figure 8.7: PPM correction of the ABBYY OCR output text
Note: (a) An Arabic text image from the KFAD dataset, (b) the proposed OCR output text that was produced for the image in (a) and (c) the corrected OCR output text after applying the PPM correction method.

8.4 Overall Experimental Evaluation of the PATRION OCR System

In this section, we compare the PATRION OCR with results obtained using the four Arabic OCR systems studied in Chapter 3, which are Sakhr OCR, ABBYY OCR, RDI OCR and Tesseract OCR. The comparative evaluation results were obtained by running Arabic OCR systems on the same dataset mentioned in Section 8.3. For a fair comparison, the recognition accuracy of numbers and punctuations of Arabic OCR systems have been excluded from the overall accuracy, since the recognition of numbers and punctuations are out of the scope of the PATRION OCR system. The automated tool described in chapter 3 has been used in order to provide performance accuracy rates using various metrics that provide a better insight into the effectiveness of the Arabic OCR systems with respect to the challenges of Arabic script.

The accuracy of Arabic OCR systems in recognising Arabic characters based on their position in a word are provided in Figure 8.8. The results show that all of the Arabic OCR systems have higher accuracy in recognising isolated characters when they are not connected with other characters in a word. This supports the idea that the connectivity feature of Arabic script is still an obstacle to Arabic text recognition. However, high scores of character accuracy for the four character positions were obtained by the PATRION OCR system. From figure 8.8, it can be seen that a drop in character recognition accuracy occurs when recognising end characters by the PATRION OCR system. The low recognition accuracy of end characters by the PATRION OCR system is due to segmentation errors when segmenting the end form shape of the Alef "ـا", Haa "ـه" and Yaa "ـى" characters discussed in chapter 6.

The results of the recognition accuracy performance of the Arabic OCR systems in terms of one dot, two dot, three dot and no-dot characters are illustrated in figure 8.9. These results shows that the recognition accuracy rates of no-dot characters are significantly better for Sakhr, ABBYY, RDI and Tesseract OCR system, compared to one, two and three dot characters. However, among the evaluated Arabic OCR systems, only the PATRION OCR system's recognition accuracy rates of two and three dot characters are similar to the recognition accuracy rate of no-dot characters. This confirms that the PATRION OCR system overcomes the challenge of the presence of dots in Arabic script by using the PPM correction method for adding dots to Arabic texts.

The results of analysing the performance of the Arabic OCR systems on characters that have a dot above or below the baseline are presented in figure 8.10. It is obvious that all of the evaluated Arabic OCR systems and the PATRION OCR system perform much better in identifying characters with a dot below the baseline than characters with a dot above the baseline. The reasons for these results are not entirely clear. It could be due to baseline determination algorithms implemented by these systems.

Figure 8.11 compares the recognition accuracy for zigzag-shaped characters and loop-shaped characters by the Arabic OCR systems. It is apparent from the results that the performance rates of Sakhr, ABBYY, RDI and Tesseract OCR system in recognising loop-shaped characters are higher than in recognising zigzag-shaped characters, whereas the PATRION OCR system performs better in recognising zigzag-shaped characters than loop-shaped characters. However, the PATRION OCR system has the highest performance accuracy rates for both character shapes, compared to the other four Arabic OCR systems. Moreover, the results indicate that the PATRION OCR system is more robust in recognising zigzag-shaped characters which is a challenge for the four other Arabic OCR systems.

Figure 8.12 and Table 8.3 summarize the overall accuracy scores for the comparison of the Arabic OCR systems. The evaluation results when comparing the PATRION OCR system against the other four Arabic OCR systems show that the PATRION OCR system achieved the highest accuracy of 86.9%. In summary, the results shows that the PATRION OCR system is very effective when compared the other four Arabic OCR systems.

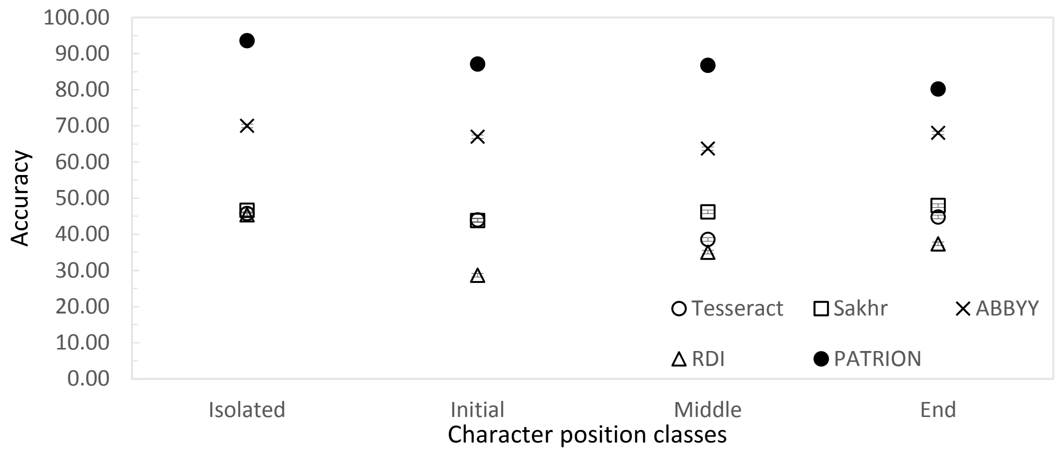


Figure 8.8: Character position class accuracy comparison on KAFD dataset

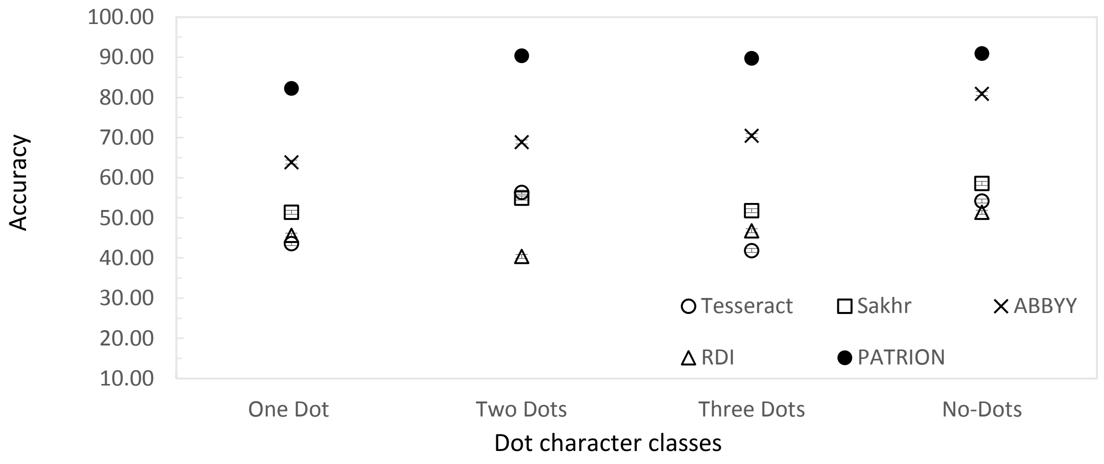


Figure 8.9: Dot character class accuracy comparison on KAFD dataset

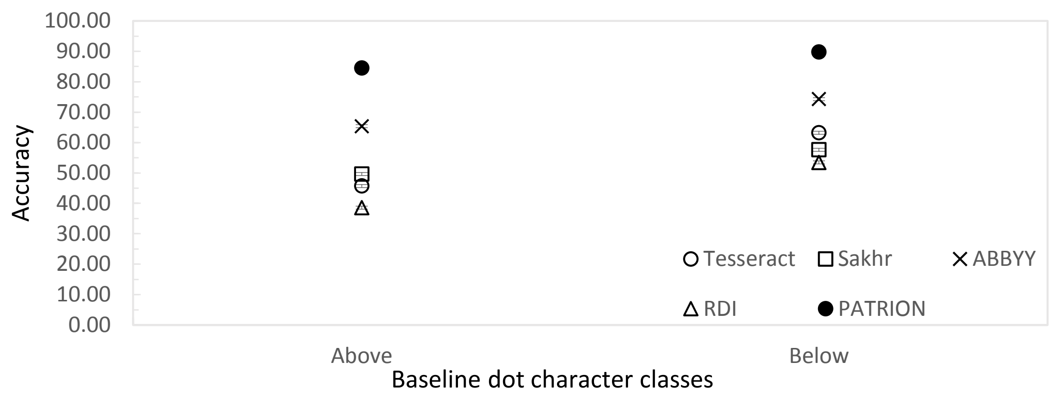


Figure 8.10: Baseline dot character class accuracy comparison on KAFD dataset

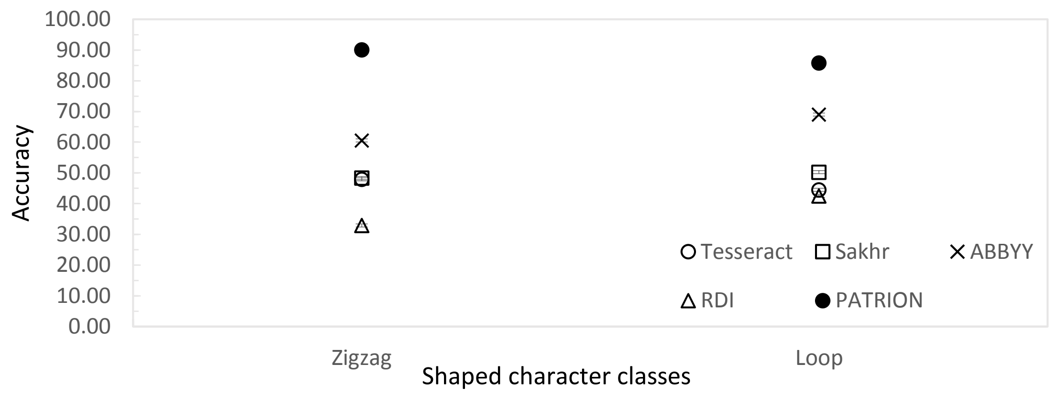


Figure 8.11: Shape character class accuracy comparison on KAFD dataset

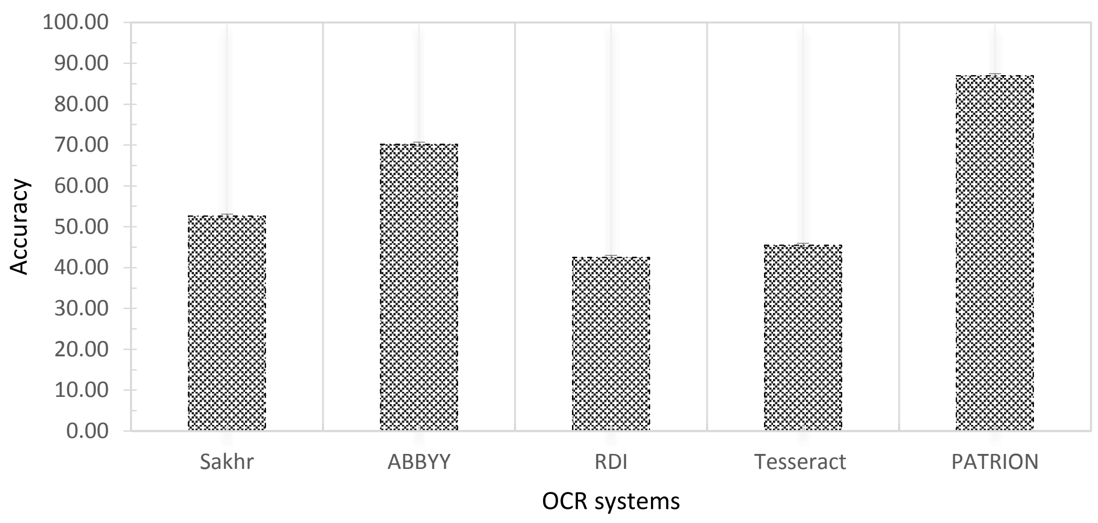


Figure 8.12: Overall Arabic OCR performance accuracy comparison on KAFD dataset

Font Type	OCR System				
	Sakhr	***ABBYY***	***RDI***	***Tesseract***	***PATRION***
Traditional Arabic	48.5%	67.7%	51.9%	47.1%	87.9%
Simplified Arabic	52.9%	67.7%	44.9%	46.7%	91.8%
Arabic Transparent	51.5%	75.2%	46.1%	48.6%	83.1%
AdvertisingBold	57.3%	70.3%	27.2%	39.4%	84.7%
Average	**52.6%**	**70.2%**	**42.5%**	**45.4%**	**86.9%**

Table 8.7: Arabic OCR performance accuracy according to four font types

8.5 Discussion and Summary

In this chapter, a post-processing method based on a PPM model was introduced for correcting Arabic OCR output. This method has been applied to two problems of Arabic OCR systems: adding dots to Arabic text and correcting the OCR output. The method has successfully corrected mis-recognition errors and dotted characters in Arabic OCR output. By applying the PPM model to the new PATRION OCR system, the edit distance accuracy improved from 77.3% to 86.9%. Also, the implementation of the PPM correction method in the PATRION OCR system has helped to overcome the challenges of the presence of dots and zigzag-shaped characters in Arabic script, since the experimental results show that the PATRION OCR system is robust in recognising dotted characters and zigzag-shaped characters.

Moreover, experimental results on a commercial Arabic OCR system show that the post-processing method was also able to increase the edit distance accuracy for that system from 70.2% to 77.1%.

Chapter 9: Conclusion and Future Work

9.1 Chapter Introduction

The need to extend digital Arabic content on the Internet and to analyse online text motivated the development of an accurate OCR system for Arabic text. However, the development of an OCR system for printed Arabic is a challenging task. These challenges are due to the specific characteristics of Arabic script. Previous research adopted different methods that were developed for other languages without considering the specific characteristics of Arabic script. Moreover, there is a lack of objective performance metrics for assessing how Arabic OCR systems are coping with the challenges of Arabic text.

In this study, we proposed several related methods that take into consideration the characteristics of Arabic text for the implementation of a robust printed Arabic OCR system. In addition, a novel automated evaluation tool was developed with new objective performance metrics.

Compression based language modelling has proven to be very effective for many NLP applications. However, no prior study has exploited the effectiveness of compression based models for the problems of Arabic OCR. In this work, we developed new approaches that make use of a compression-based model.

This chapter first provides the summary of this dissertation. Then, it reviews the aims and objectives of this work followed by reviewing the research questions. It also highlights the limitations of this work. Finally, it provides several directions for future work.

9.2 Summary

This work firstly provided a comprehensive literature review of printed Arabic text recognition. It reviewed different methods that have been used for developing Arabic OCR systems. In addition, it reviewed relevant issues of printed Arabic OCR including the challenges of printed Arabic script and performance evaluation issues. Also, it discussed the current status of printed Arabic OCR as well as the challenges and open problems in printed Arabic text recognition. It concluded that there is still a crucial need for more research, although there are various attempts to solve the problems of Arabic text recognition.

In chapter 3, we firstly developed a novel automated Arabic OCR evaluation tool. This tool has been developed based on a new set of objective performance metrics with respect to the challenges of Arabic text which are character accuracy based on character position, dot character accuracy, zigzag-shaped character accuracy, loop-shaped character accuracy, diacritics accuracy, digit accuracy and punctuation accuracy. We then proposed a standard protocol for measuring the effectiveness of Arabic optical character recognition. An automated evaluation experiment has been conducted, using the developed evaluation tool, to evaluate the performance of four well-known Arabic OCR systems: Sakhr, ABBYY, RDI and Tesseract. The experimental results showed that all the evaluated Arabic OCR systems have low performance accuracy rates, below 75%, which means that the field of character recognition for printed Arabic still requires further research to reach an efficient text recognition method for Arabic script. Chapter 3 also concluded that the newly developed automated evaluation tool is able to automatically assess how Arabic OCR systems are overcoming the challenges of Arabic text and it contributes to eliminate human error, improve speed and precision, and reduce repetition in evaluating Arabic OCR systems.

Our study mainly aimed at developing a novel OCR for printed Arabic text that substantially improves upon the state-of-the-art. To achieve this aim, a new PATRION OCR system has been implemented in this work using five stages: (1) thinning; (2) feature extraction; (3) character segmentation; (4) classification; (5) post-processing.

In chapter 4, a new thinning algorithm for Arabic text has been developed. Also, we proposed an experimental framework with new objective performance metrics for evaluating thinning algorithms for Arabic text in terms of connectivity and dots persevering. An experimental evaluation was conducted to evaluate the new thinning algorithm against two well-known thinning algorithms, Zhan-Suen and Hilditch, with respect to the new proposed performance metrics and other different adopted metrics. The experimental results showed that the new algorithm has the better performance than the other two thinning algorithms. Chapter 4 also concluded that the new objective performance metrics are sufficient to provide us with insight of which thinning algorithm performs better for Arabic OCR.

In chapter 5, a new technique based on a cognitive insight used for extracting features from the skeleton of non-dotted Arabic text images has been developed using an agent-based NetLogo model. This method has been applied prior to the character segmentation phase in order to overcome the challenges of Arabic character segmentation. The method has been tested on a sample of 100 Arabic words images and has produced an average edit distance accuracy of 98.07%. Chapter 5 also concluded that most of the produced errors by the feature extraction method are related to the thinning algorithm.

Chapter 6 presented a new character segmentation technique for segmenting connected Arabic words into characters based on the extracted feature. The character segmentation method identified the segmentation point in the extracted feature string based on two features of Arabic script: the junction line and the branch point. The method has been tested on a sample of 100 Arabic words images and has produced a recall of 84.2% and a precision of 77.2%. Chapter 6 also concluded that the segmentation method has effectively segmented overlapping characters which is one of the main characteristics of Arabic text that causes difficulties to the development of character segmentation methods.

In Chapter 7, a compression based model has been applied for the first time to Arabic textual feature recognition. Two classification experiments were conducted in order to classify and recognise isolated Arabic character images and Arabic text images using PPM models. The method has produced an average edit distance accuracy of 77.3%. Chapter 7 also concluded that due to the challenge of the connectivity characteristic of Arabic script, the PPM classifier performs better in classifying pre-segmented Arabic characters than in classifying characters from paragraph-based text images.

In the final stage of the development of the PATRION system, a post-processing technique based on a PPM correction method was applied for correcting Arabic OCR texts, in an attempt to improve the recognition accuracy, as explained in chapter 8. This method has improved the edit distance accuracy of the PATRION system from 77.3% to 86.9%. The PPM correction method was also applied to a commercial Arabic OCR system and it was able to improve the edit distance accuracy for that system from 70.2% to 77.1%. Also, the PATRION system was evaluated against the four well-known Arabic OCR systems (mentioned above). The results show that the PATRION OCR system is very effective compared to the other four Arabic OCR systems, as it achieved the highest accuracy of 86.9%. Chapter 8 also concluded that the implementation of the PPM correction method in the PATRION OCR system has helped to overcome the challenges of the presence of dots and zigzag-shaped characters in Arabic text.

9.3 Review of Aims and Objectives

The aims and objectives of this work listed in Section 1.2 have all been successfully achieved. A novel OCR system (PATRION) for printed Arabic script has been developed. The PATRION OCR system has been evaluated and compared with four well-known Arabic OCR systems and the experimental results showed that the system substantially improves upon the other OCR systems.

The specific objectives addressed in Section 1.2 were achieved as follows:

- *perform a comprehensive review of printed Arabic text recognition.*

  A comprehensive review of printed Arabic text recognition has been performed. This objective was achieved in chapter 2.

- *develop an automated tool for evaluating the performance of Arabic OCR systems.*

  An automated evaluation tool with new objective metrics for the evaluation of Arabic OCR performance has been developed. This objective was achieved in chapter 3.

- *evaluate the effectiveness of the state-of-the-art printed Arabic text recognition systems.*

  The state-of-the-art printed Arabic text recognition systems have been evaluated in chapter 3.

- *design and implement a new thinning algorithm for Arabic text and evaluate the effectiveness of the new algorithm by comparing it with well-known thinning algorithms.*

A novel thinning algorithm for Arabic text has been developed and evaluated by comparing it with well-known thinning algorithms. This objective was achieved in chapter 4.

- *develop a new chain-code representation technique using an agent-based model for extracting features from non-dotted Arabic text images prior to the character segmentation phase.*

  A new technique based on chain-code representation used for extracting features from non-dotted Arabic text images prior to the character segmentation phase has been developed. This objective was achieved in chapter 5.

- *develop a new character segmentation technique for segmenting connected Arabic words into characters based on the extracted features.*

  A new character segmentation technique based on the features extracted from Arabic text has been developed. This objective was achieved in chapter 6.

- *develop a new compression-based method for classifying the segmented Arabic characters.*

  A novel compression based method has been applied as a classifier to Arabic textual feature recognition. This objective was achieved in chapter 7.

- *use compression-based post-processing techniques (for example, adding dots to Arabic text and correcting the OCR output).*

  Finally, a compression based post-processing technique has been applied to two problems: adding dots to Arabic text and correcting the OCR output. This objective was achieved in chapter 8.

9.4 Review of Research Questions

This section reviews the specific research questions of this work that were listed in Section 1.3.

- *Is the current OCR methodology which involves the five sequential stages (pre-processing; feature extraction; segmentation; classification and post-processing) the most effective for designing Arabic OCR?*

The literature review noted that most of the current studies in Arabic OCR used the general methodology that involves the five sequential stages. Some studies ignored the segmentation stage by developing Arabic OCR systems based on a free-segmentation or holistic approach which recognize words or sub-words as a whole unit with no segmentation in order to overcome the challenges of character segmentation. However, such systems are lexicon based systems. From reviewing the work on the five stages of printed Arabic OCR, we reached the conclusion that there is less opportunity for improvement for Arabic OCR by using the general OCR methodology, since most approaches used in each stage have been adopted from approaches that have been developed for other languages without considering the characteristics of Arabic script.

- *Are there alternative methodologies that might yield better results for Arabic OCR?*

  This study developed a new methodology for implementing printed Arabic OCR system. In particular, the developed system performs the feature extraction stage prior to the character segmentation stage, unlike the general methodology, in order to overcome the challenges of Arabic character segmentation. As indicated in the comparative evaluation results of the developed OCR (PATRION) and four other OCR systems (shown in chapter 9), the approach which was used in developing the PATRION OCR system seems to be promising as it achieves better recognition performance.

- *Are the general standard performance measurements of character accuracy sufficient to assess how Arabic OCR systems are coping with the challenges of Arabic script?*

  The literature showed that printed Arabic OCR poses great challenges due to the special characteristics of Arabic text. Therefore, different methods have been proposed for developing printed Arabic OCR systems. Most of the proposed methods have been evaluated in terms of a general metric which is character accuracy. By relying on such a metric, we cannot fully assess how Arabic OCR systems are overcoming the challenges of Arabic text. Therefore, we have suggested a new set of objective performance metrics for the evaluation of Arabic OCR systems with respect to the challenges of Arabic script. These are character accuracy based on character position, dot character accuracy, zigzag-shaped character accuracy, loop-shaped character accuracy and diacritics accuracy, as described in chapter 3, section 3.2.

- *Are the available OCR systems sufficient to recognise printed Arabic text images?*

  The experimental results, obtained from evaluating the four well-known Arabic OCR systems discussed in chapter 3 (section 3.6), showed that all the evaluated Arabic OCR systems have low performance accuracy rates, below 75%, which means that the OCR systems are insufficient to recognise printed Arabic text images.

- *Can Arabic text be effectively recognised by considering Arabic text as non-dotted text and then corrected later on in order to recover the dotted from?*

  In this work, the PATRION OCR system was designed to recognise non-dotted Arabic text images. In particular, the new feature extraction method, implemented in the PATRION system (see chapter 5), has been developed for extracting features from non-dotted text, as it was assumed that all Arabic words in the text images are non-dotted. The feature extraction method was effectively able to extract features from non-dotted Arabic text. Then the recognition of dots has been considered in the final stage of the development of the system by applying a post-processing method to recover unrecognised dots. The post-processing method was effectively able to recover the dotted form, as shown in chapter 9.

- *Can a compression based method be effectively applied as a classifier for the recognition of the text features?*

  Our approach in this work was effectively able to recognise Arabic characters by applying the PPM compression method as a classifier in the PATRION OCR system in order to classify the feature strings obtained from the feature extraction stage. The explanation of the implementation of the PPM compression method as a classifier is shown in chapter 7, section 7.2.

- *Can a compression based method be applied to Arabic OCR output that contributes to significant improvement in Arabic text recognition accuracy?*

A compression based correction method was applied as a post-processing method to correct Arabic OCR output. This method has been applied to two problems of Arabic OCR systems: adding dots to Arabic text and correcting the OCR output. The method has successfully corrected OCR errors and dotted characters in Arabic OCR output. The post-processing method was able to increase the edit distance accuracy for the PATRION OCR system from 77.3% to 86.9%, and for a commercial Arabic OCR system from 70.2% to 77.1%, which are significant improvements. The explanation of the implementation of the compression method as a post-processing method is provided in chapter 8, section 8.2.

9.5 Limitations of the Work

Various limitations of this work have occurred for various reasons. Firstly, the experimental evaluation (presented in chapter 3) was limited to four Arabic OCR systems, since we were not be able to obtain other Arabic OCR systems. Also, the evaluation study was only concerned with the output of the Arabic OCR systems rather than how the output is produced, since accessing the submodules of the Arabic OCR systems studied in chapter 3 was not possible.

Another limitation is that due to the limited availability and accessibility of Arabic corpora, this work used a training corpus of only 31-millon Arabic words, which is the BACC Corpus, for constructing the language model used in the post-processing method. Using a larger Arabic corpus might result in better performance.

Finally, the performance of the PATRION OCR system was evaluated using only two pitch sizes (12 and 24) in two styles (normal and italic) for four font types (Arabic Transparent, AdvertisingBold, Simplified Arabic and Traditional Arabic) at 300 dots per inch (dpi). The OCR system can be evaluated using more font types and pitch sizes by training various character models of different font types and pitch sizes, as explained in chapter 7 (section 7.2.1). However, extracting feature strings from pre-segmented character images for training purposes would be a time consuming task.

9.6 Future Work

In this work, various achievements have been made in the area of Arabic text recognition. However, several observations have been noted for possible future directions. The following areas should be considered for future work:

- *Creating a noisy page-level text image dataset.*

This study used the KFAD dataset for evaluating printed Arabic OCR systems as it is freely available, paragraph-based, and comprises text images of different font types, pitch sizes, and resolutions. However, the KFAD dataset is made up of noise-free text images. Thus, a noisy Arabic text image dataset is required to assess the effectiveness of Arabic OCR systems on poor quality text images.

- *Implementing the thinning algorithm for different OCR systems.*

This study has proposed an efficient thinning algorithm. It will be interesting to measure the impact of the proposed thinning algorithm on the performance accuracy of different Arabic OCR systems.

- *Including more objective performance metrics into the proposed evaluation tool.*

This study provided an evaluation tool with a new set of objective performance metrics for assessing the performance of Arabic OCR with respect to the challenges of Arabic script. This tool can be enhanced by including different performance metrics that can provide a better insight into the effectiveness of Arabic OCR systems regarding the challenges of Arabic text. Since Arabic script has some characters that share an identical visual shape, performance metrics related to visual shape can be a further possible performance metric.

- *Applying the feature extraction method to handwritten text recognition.*

The feature extraction method, which is presented in chapter 5, resembles the way a human might write a word or a character with a pen. Therefore, this method could be applied to handwritten text images and it may contribute significantly to the enhancement of the field of handwritten text recognition.

- *Using the PPM based method for diacritising OCR output text.*

The experimental results presented in chapter 3 (section 3.6) concluded that recognising Arabic text with diacritics is still an open research problem. To solve this problem, the PPM compression based method could be used for diacritising Arabic text in the same way it is used for adding dots to non-dotted Arabic text, as described in chapter 8 (section 8.2). However, an Arabic corpus with diacritised text is required for training.

- *Evaluating Arabic OCR approaches in terms of execution time.*

This research focused on improving and evaluating the recognition accuracy of printed Arabic OCR. However, it will be interesting to study the effectiveness of OCR approaches in terms of both the execution time and recognition accuracy.

References

ABBYY OCR [WWW document], n.d. URL https://www.abbyy.com/en-gb/ (accessed 4.15.17).

Abd, M., Paschos, G., 2007. Effective Arabic character recognition using support vector machines. In Innovations and Advvanced Techniques in Computer and Information Sciences and Engineering. Springer, pp. 7–11.

Abd, M., Al Rubeaai, S., Paschos, G., 2012. Hybrid features for an Arabic word recognition system. Computer Technology and Application. David publishing, 3(10), pp. 685–691.

AbdelRaouf, A.M., 2012. Offline printed Arabic character recognition. PhD thesis, University of Nottingham.

Abdelraouf, A., Higgins, C.A., Khalil, M., 2008. A database for Arabic printed character recognition. In International Conference Image Analysis and Recognition. Springer, pp. 567–578. DOI:10.1007/978-3-540-69812-8_56

Aboaisha, H., Xu, Z., El-Feghi, I., 2012. An investigation on efficient feature extraction approaches for Arabic letter recognition. In: Proceedings of the Queen's Diamond Jubilee Computing and Engineering Annual Researchers' Conference, pp. 80-85.

Abu-Ain, W., Abdullah, S.N.H.S., Bataineh, B., Abu-Ain, T., Omar, K., 2013. Skeletonization algorithm for binary images. Procedia Technol, 11, 704–709. DOI:10.1016/j.protcy.2013.12.248

Abu-Ain, W., Abdullah, S.N.H.S, Omar, K., 2014. A simple iterative thinning algorithm for text and shape binary images. Journal of Theoretical & Applied Information Technology, 63(2), pp. 274–281.

Abu-Ain, W.A.K., Abdullah, S.N.H.S., Omar, K., Rahman, S.Z.A., 2018. Automatic multi-lingual script recognition application. GEMA Online®Journal of Language Studies, 18(3), pp. 203–221.

Abuhaiba, I.S.I., 2003. Skew correction of textural documents. Journal of King Saud University-Computer and Information Sciences. Elsevier, 15, pp. 73–93. DOI:10.1016/S1319-1578(03)80003-4

Abuhaiba, I.S.I., Mahmoud, S.A., Green, R.J., 1994. Recognition of handwritten cursive Arabic characters. IEEE Transactions on Pattern Analysis and Machine Intelligence. IEEE, 16(6), pp. 664–672. DOI:10.1109/34.295912

Ahmad, I., 2016. Modeling and training options for handwritten Arabic text recognition. PhD thesis, Technische Universität Dortmund.

Ahmad, I., Mahmoud, S.A., Fink, G.A., 2016. Open-vocabulary recognition of machine-printed Arabic text using hidden Markov models. Pattern Recognition. Elsevier, 51, pp. 97–111. DOI:10.1016/j.patcog.2015.09.011

Ahmad, I., Rothacker, L., Fink, G.A., Mahmoud, S.A., 2013. Novel sub-character HMM models for Arabic text recognition. In International

Conference on Document Analysis and Recognition. IEEE, pp. 658–662. DOI:10.1109/ICDAR.2013.135

Ahmed, S. Bin, Naz, S., Razzak, M.I., Yusof, R., 2019. Arabic Cursive Text Recognition from Natural Scene Images. Applied Sciences, 9(2), p. 236.

Ahmed, I., Mahmoud, S.A., Parvez, M.T., 2012. Printed Arabic text recognition. In: Märgner, V., El Abed, H. (Eds.), Guide to OCR for Arabic Scripts. Springer London, London, pp. 147–168. DOI:10.1007/978-1-4471-4072-6_7

Al-ani, H., Ban, N., Abass, H.M., 2014. Printed Arabic character recognition using neural network. Journal of Emerging Trends in Computing and Information Sciences. 5, pp. 64–66.

Al-Ayyoub, M., Nuseir, A., Alsmearat, K., Jararweh, Y., Gupta, B., 2018. Deep learning for Arabic NLP: A survey. Journal of Computer Science, 26, pp. 522–531.

Al-Badr, B.H., 1995. A segmentation-free approach to text recognition with application to Arabic text. PhD thesis, University of Washington, Seattle, WA, USA.

Al-Badr, B., Haralick, R.M., 1995. Segmentation-free word recognition with application to Arabic. In: Proceedings of 3rd International Conference on Document Analysis and Recognition. IEEE, 1, pp. 355–359. DOI:10.1109/ICDAR.1995.599012

Al-Badr, B., Haralick, R.M., 1998. A segmentation-free approach to text recognition with application to Arabic text. International Journal on Document Analysis and Recognition, 1, pp. 147–166.

Al-Badr, B., Mahmoud, S.A., 1995. Survey and bibliography of Arabic optical text recognition. Signal Processing, 41(1), 49–77. DOI:10.1016/0165-1684(94)00090-M

Al-Helali, B.M., Mahmoud, S.A., 2017. Arabic Online Handwriting Recognition (AOHR): A survey. ACM Computing Surveys, 50(3), p. 33.

Al-Muhtaseb, H., Mahmoud, S.A., Qahwaji, R.S., 2008. Recognition of off-line printed Arabic text using hidden Markov models. Signal Processing, 88(12), pp. 2902–2912.

Al-Muhtaseb, H., Qahwaji, R., 2011. Arabic Optical Character Recognition: Recent trends and future directions. In: Applied Signal and Image Processing: Multidisciplinary Advancements. IGI Global, pp. 324–346.

Al-Rashaideh, H., 2006. Preprocessing phase for Arabic Word Handwritten Recognition. Information Process, 6, pp. 11–19.

Al-Shatnawi, A.M., Al-Zawaideh, F.H., Al-Salameh, S., Omar, K., 2011. Offline Arabic Text Recognition: An overview. World Computing Science and Information Technology, 1, pp. 184–192.

Al-Shatnawi, A.M., Alfawwaz, B.M., Omar, K., Zeki, A.M., 2014. Skeleton extraction: Comparison of five methods on the Arabic IFN/ENIT database. In: 6th International Conference on Computer Science and Information

Technology (CSIT). IEEE, pp. 50–59.

Al-Shatnawi, A., Omar, K., 2008. Methods of Arabic language baseline detection: The state of art. In International Journal of Computer Science and Network Security, 8, pp. 137–143.

Al-Shatnawi, A.M., Omar, K., 2009. Skew detection and correction technique for Arabic document images based on centre of gravity. Department of System Science and Management, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. Journal of Computer Science, 5(5), pp. 363–368. DOI:10.3844/jcssp.2009.363.368

Al-Shatnawi, A.M., Omar, K., 2014. The thinning problem in Arabic text recognition: A comprehensive review. International Journal of Computer Applications, 103(3), pp. 35–42.

Al-Yousefi, H., Udpa, S.S., 1992. Recognition of Arabic characters. IEEE Transactions on Pattern Analysis & Machine Intelligence, 14, pp. 853–857. DOI:10.1109/34.149585

Alginahi, Y., 2004. Computer analysis of composite documents with non-uniform background. PhD thesis, University of Windsor, Ontario, Canada.

Alginahi, Y., 2010. Preprocessing techniques in character recognition. In Character Recognition. InTech, pp. 1–20.

Alginahi, Y.M., 2013. A survey on Arabic character segmentation. International Journal on Document Analysis and Recognition (IJDAR), 16(2), pp. 105–26. DOI:10.1007/s10032-012-0188-6

Alginahi, Y., 2017. Document processing and Arabic optical character recognition: A user perspective. International Journal of Computer Science and Information Security, 15(12), pp. 86–97.

Alhawiti, K.M., 2014. Adaptive models of Arabic text. Phd thesis, Prifysgol Bangor University, Bangor, UK.

Ali, M.A., 2012. An efficient thinning algorithm for Arabic OCR systems. Signal Image Processing, 3(3), pp. 31–38.

Ali, M., bin Jumari, K., 2003. Skeletonization algorithm for an Arabic handwriting. WSEAS Transactions on Computers, 2(3), pp. 662–667.

Aljarrah, I., Al-Khaleel, O., Mhaidat, K., Alrefai, M., Alzu'Bi, A., Rabab'Ah, M., 2012. Automated system for Arabic optical character recognition with lookup dictionary. Journal of Emerging Technologies in Web Intelligence, 4(4), pp. 362–370. DOI:10.4304/jetwi.4.4.362-370

AlKhateeb, J., 2010. Word based off-line handwritten Arabic classification and recognition. PhD thesis, School of Computing, Informatics and Media, University of Bradford.

AlKhateeb, J.H., Jiang, J., Ren, J., Ipson, S., 2009. Component-based segmentation of words from handwritten Arabic text. International Journal of Computer Systems Science and Engineering, 5(1), pp. 54–58.

Alkhazi, I.S., Alghamdi, M., Teahan, W.J., 2017. Tag based models for Arabic text compression. In Intelligent Systems Conference (IntelliSys). IEEE, pp. 697–705.

Alkhazi, I.S., Teahan, W.J., 2017. Classifying and segmenting Classical and Modern Standard Arabic using minimum cross-entropy. In International Journal of Advanced Computer Science and Applications (IJACSA), 8(4), pp. 421–430. DOI:10.14569/IJACSA.2017.080456

Alkholy, M.D.A.-Z., 2016. Arabic optical character recognition using local invariant features. PhD thesis, Faculty of Computers and Information, Menoufia University.

Almahdawi, A., Teahan, W.J., 2017. Emotion recognition in text using PPM. In: International Conference on Innovative Techniques and Applications of Artificial Intelligence. Springer, pp. 149–155. DOI:10.1007/978-3-319-71078-5_13

Almohri, H., Gray, J.S., Alnajjar, H., 2007. A real-time DSP-based optical character recognition system for isolated Arabic characters using the TI TMS320C6416T. PhD thesis, University of Hartford.

AlSalamah, S., King, R., 2018. Towards the Machine Reading of Arabic Calligraphy: A Letters Dataset and Corresponding Corpus of Text. In: IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR). IEEE, pp. 19–23. DOI:10.1109/ASAR.2018.8480228

Altamimi, M.J., Teahan, W., 2017. Gender and authorship categorisation of Arabic text from Twitter using PPM. In International Journal of Computer Science & Information Technology (IJCSIT), 9, pp. 131–140. DOI:10.5121/ijcsit.2017.9212

Altuwaijri, M., Bayoumi, M., 1995. A new thinning algorithm for Arabic characters using self-organizing neural network. In: IEEE International Symposium on Circuits and Systems (ISCAS '95). IEEE, 3, pp. 1824–1827.

Amara, M., Ghedira, K., Zidi, K., Zidi, S., 2015. A comparative study of multi-class support vector machine methods for Arabic characters recognition. In: IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA). IEEE, pp. 1–6.

Amara, M., Zidi, K., Ghedira, K., Zidi, S., 2016. New rules to enhance the performances of histogram projection for segmenting small-sized Arabic words. In: Hybrid Intelligent Systems. Springer, pp. 167–176.

Amara, M., Zidi, K., Zidi, S., Ghedira, K., 2014. Arabic character recognition based M-SVM. In: International Conference on Advanced Machine Learning Technologies and Applications. Springer, pp. 18–25.

Amin, A., 1998. Off-line Arabic character recognition. Pattern Recognition, 31(5), pp. 517–530. DOI:10.1016/S0031-3203(97)00084-8

Amin, A., Masini, G., 1986. Machine recognition of multi font printed {Arabic} texts. In: Eighth International Conference on Pattern Recognition. Paris, France, pp. 392–395.

Arica, N., Yarman-Vural, F.T., 2002. Optical character recognition for cursive handwriting. IEEE Transactions on Pattern Analysis and Machine Intelligence. IEEE, 24(6), pp. 801–813. DOI:10.1109/TPAMI.2002.1008386

Arica, N., Yarman-Vural, F.T., 2001. An overview of character recognition focused on off-line handwriting. IEEE Transactions on Systems, Man, and Cybernetics, Part C Applications and Reviews, 31(2), pp. 216–233. DOI:10.1109/5326.941845

Arora, S., Bhattacharjee, D., Nasipuri, M., Malik, L., Kundu, M., Basu, D.K., 2010. Performance comparison of SVM and ANN for handwritten Devnagari character recognition. International Journal of Computer Science, 7, pp. 1–10.

Awaida, S.M., Khorsheed, M.S., 2012. Developing discrete density: Hidden Markov models for Arabic printed text recognition. Proc. 2012 IEEE International Conference on Computational Intelligence and Cybernetics (CyberneticsCom). IEEE, pp. 35–39. DOI:10.1109/CyberneticsCom.2012.6381612

Baik, M.K., 1992. History of Arabic Language. Dar Sa'ad Aldin, Damascus 1.

Baird, H.S., 1995. The skew angle of printed documents. In: Document Image Analysis. pp. 204–208.

Bassil, Y., Alwani, M., 2012. OCR Post-processing error correction algorithm using Google's online spelling suggestion. Journal of Emerging Trends in Computing and Information Sciences, 3(1), pp. 90–99.

Ben Amor, N., Ben Amara, N.E. 2006. Multifont Arabic characters recognition using HoughTransform and HMM/ANN Classification. Journal of Multimedia, 1(2), pp. 50–54.

Ben Amor, N., Ben Amara, N.E. 2012. A novel method for multifont Arabic characters features extraction. INTECH Open Access Publisher.

Ben Moussa, S., Zahour, A., Benabdelhafid, A., Alimi, A.M., 2010. New features using fractal multi-dimensions for generalized Arabic font recognition. Pattern Recognition Letters. Elsevier, 31(5), pp. 361–371. DOI:10.1016/j.patrec.2009.10.015

Bobik, J., Sayre, K.M., 1963. Pattern recognition mechanisms and St. Thomas' theory of abstraction. Revue philosophique de Louvain, 61(69), pp. 24–43.

Buades, A., Coll, B., Morel, J., 2005. A review of image denoising algorithms, with a new one. Multiscale Modeling & Simulation, 4(2), pp. 490–530. DOI:10.1137/040616024

Bukhari, S.S., Shafait, F., Breuel, T.M., 2011. High performance layout analysis of Arabic and Urdu document images. In: International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp. 1275–1279. DOI:10.1109/ICDAR.2011.257

Carrasco, R.C., 2014. An open-source OCR evaluation tool. In: First International Conference on Digital Access to Textual Cultural Heritage

(DATeCH ’14). ACM, New York, NY, USA, pp. 179–184. DOI:10.1145/2595188.2595221

Chatbri, H., Kameyama, K., 2014. Using scale space filtering to make thinning algorithms robust against noise in sketch images. Pattern Recognition Letters. Elsevier, 42, pp. 1–10. DOI:10.1016/j.patrec.2014.01.011

Chen, C.H., DeCurtins, J.L., 1993. Word recognition in a segmentation-free approach to OCR. In: Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR'93). IEEE, pp. 573–576. DOI:10.1109/ICDAR.1993.395670

Cheriet, M., Kharma, N., Liu, C., Suen, C., 2007. Character recognition systems: A guide for students and practitioners. John Wiley & Sons. DOI:10.1002/9780470176535

Cheung, A., Bennamoun, M., Bergmann, N.W., 2001. Arabic optical character recognition system using recognition-based segmentation. Pattern Recognition. Elsevier, 34(2), pp. 215–233. DOI:10.1016/S0031-3203(99)00227-7

Cho, W., Lee, S.W., Kim, J.H., 1995. Modeling and recognition of cursive words with hidden Markov models. Pattern Recognition. Elsevier, 28(12), pp. 1941–1953. DOI:10.1016/0031-3203(95)00041-0

Choudhary, A., 2014. A review of various character segmentation techniques for cursive handwritten words recognition. International Journal of Information & Computation Technology, 4(6), 559–564.

Cleary, J., Witten, I., 1984. Data compression using adaptive coding and partial string matching. IEEE Transactions on Communications, 32(4), 396-402.

Cowell, J., Hussain, F., 2001. Thinning Arabic characters for feature extraction. In: Proceedings of the International Conference on Information Visualisation. IEEE, pp. 181–185. DOI:10.1109/IV.2001.942056

Dahi, M., Semary, N.A., Hadhoud, M.M., 2015. Primitive printed Arabic Optical Character Recognition using statistical features. In: IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS). IEEE, pp. 567–571. DOI:10.1109/IntelCIS.2015.7397278

Damien, P., Wakim, N., Egea, M. 2009. Phoneme-viseme mapping for Modern, Classical Arabic language. In: International Conference on Advances in Computational Tools for Engineering Applications (ACTEA). IEEE, pp. 547–552. DOI:10.1109/ACTEA.2009.5227875

Devi, H., 2006. Thinning: A preprocessing technique for an OCR system for the Brahmi script. Ancient Asia, 1, pp. 167–172.

Dong, J.X., Dominique, P., Krzyzak, A., Suen, C.Y., 2005. Cursive word skew/slant corrections based on Radon transform. In Eighth International Conference on Document Analysis and Recognition (ICDAR'05). IEEE, pp. 478–483. DOI:10.1109/ICDAR.2005.83

Doush, I.A., Al-Trad, A.M., 2016. Improving post-processing optical character recognition documents with Arabic language using spelling error detection and correction. International Journal of Reasoning-based Intelligent Systems, 8(4), pp. 91–103. DOI:10.1504/IJRIS.2016.082957

Doush, I.A., AlKhateeb, F., Gharaibeh, A.H., 2018a. A novel Arabic OCR post-processing using rule-based and word context techniques. International Journal on Document Analysis and Recognition, 21(2), pp. 77–89. DOI:10.1007/s10032-018-0297-y

Doush, I.A., AIKhateeb, F., Gharibeh, A.H., 2018b. Yarmouk Arabic OCR dataset. In: 8th International Conference on Computer Science and Information Technology (CSIT). IEEE, pp. 150–154.

Drira, F., Lebourgeois, F., 2012. Denoising textual images using local/non-local smoothing filters: A comparative study. In International Conference on Frontiers in Handwriting Recognition. IEEE, pp. 521–526. DOI:10.1109/ICFHR.2012.198

Due Trier, Ø., Jain, A.K., Taxt, T., 1996. Feature extraction methods for character recognition: A survey. Pattern Recognition, 29(4), pp. 641–662. DOI:10.1016/0031-3203(95)00118-2

El-Dabi, S.S., Ramsis, R., Kamel, A., 1990. Arabic character recognition system: A statistical approach for recognizing cursive typewritten text. Pattern Recognition, 23(5), pp. 485–495. DOI:10.1016/0031-3203(90)90069-W

El-Hajj, R., Likforman-Sulem, L., Mokbel, C., 2005. Arabic handwriting recognition using baseline dependant features and hidden Markov modeling. In International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp. 893–897. DOI:10.1109/ICDAR.2005.53

El-Mahallawy, M.S.M., 2008. A large scale HMM-based Omni font-written OCR system for cursive scripts. PhD thesis, Faculty of Engineering, Cairo University, Giza, Egypt.

Elgammal, A.M., Ismail, M.A. 2001. A graph-based segmentation and feature extraction framework for Arabic text recognition. In: Proceedings of the Sixth International Conference on Document Analysis and Recognition. IEEE, pp. 622–626. DOI:10.1109/ICDAR.2001.953864

Elrube, I.A., El Sonni, M.T., Saleh, S.S., 2010. Printed Arabic sub-word recognition using moments. *World Academy of Science, Engineering and Technology, International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, 4(6), pp. 959–963.

Erlandson, E.J., Trenkle, J.M., Vogt, R.C., 1996. Word-level recognition of multifont Arabic text using a feature vector matching approach. In: Document Recognition III. International Society for Optics and Photonics, pp. 63–71.

Etidal, 2018. Etidal – Global center for combating extremist ideology [WWW document]. URL https://etidal.org/en/home/ (accessed 3.6.19).

Freeman, H., 1961. On the encoding of arbitrary geometric configurations. IRE Transactions on Electronic Computers, pp. 260–268. DOI:10.1109/TEC.1961.5219197

Gao, X., Xiao, B., Tao, D., Li, X., 2010. A survey of graph edit distance. Pattern Analysis and Applications, 13(1), pp. 113–129. DOI:10.1007/s10044-008-0141-y

Garcia, A.L., 2005. Arabic Learning Materials [WWW Document]. URL https://www.scribd.com/document/294105424/Arabic-Writing-Sheet-Revised (accessed 3.2.18).

Ghosh, D., Dube, T., Shivaprasad, A., 2010. Script recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(2), pp. 2142–2161. DOI:10.1109/TPAMI.2010.30

Gonzalez, R., Woods, R., 2002. Digital image processing. Prentice Hall. DOI:10.1016/0734-189X(90)90171-Q

Gouda, A.M., Rashwan, M.A., 2004. Segmentation of connected Arabic characters using hidden Markov models. In: 2004 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA). IEEE, pp. 115–119.

Goyal, G., Dutta, M., 2016a. Design of Pixel neighborhood based Offline Handwritten Thinning Framework for Devnagri Numeral Script using Elman Neural Network. International Journal of Computer Science and Information Security, 14(7), p. 369.

Goyal, G., Dutta, M., 2016b. Experimental approach for performance analysis of thinning algorithms for Offline Handwritten Devnagri Numerals. Indian Journal of Science and Technology, 9(30) pp. 1–10.

Guo, Z., Hall, R.W., 1992. Fast fully parallel thinning algorithms. CVGIP Image Understanding, 55(3), pp. 317–328. DOI:10.1016/1049-9660(92)90029-3

Hamza, A.A., 2008. Back propagation neural network Arabic characters classification module utilizing Microsoft word. Journal of Computer Science, 4(9), p. 744.

Hassin, A.H., Tang, X.-L., Liu, J.-F., Zhao, W., 2004. Printed Arabic character recognition using HMM. Journal of Computer Science and Technology, 19(4), pp. 538–543. DOI:10.1007/BF02944755

Hilditch, C., 1969. Linear skeletons from square cupboards. In: Meltzer, B., Michie, D. (Eds.), Machine Intelligence 4. Edinburgh University Press, p. 403.

Hilditch, C., 1983. Comparison of thinning algorithms on a parallel processor. Image and Vision Computing, 1(3), pp. 115–132. DOI:10.1016/0262-8856(83)90063-X

Hossain, M.Z., 2012. Rapid feature extraction for Optical Character Recognition. arXiv preprint, pp. 1–5.

Hosseini, H.M.M., 1997. Analysis and recognition of Persian and Arabic handwritten characters. PhD thesis, Department of Electrical and Electronic Engineering, University of Adelaide.

Huang, L., Wan, G., Liu, C., 2003. An improved parallel thinning algorithm. Proc. Int. Conference on Document Analysis and Recognition (ICDAR), pp.

780–783. DOI:10.1109/ICDAR.2003.1227768

Hull, J., 1998. Document image skew detection: Survey and annotated bibliography. Ser. Mach. Perception and Artificial Intelligence. Document Analysis Systems II, pp. 40–64.

Jain, M., Mathew, M., Jawahar, C. V., 2018. Unconstrained OCR for Urdu using deep CNN-RNN hybrid networks, in: Proceedings of the 4th Asian Conference on Pattern Recognition (ACPR 2017). IEEE, pp. 747–752. DOI:10.1109/ACPR.2017.5

Jambi, H., 2014. King Abdullah Bin Abdulaziz AL-Saud Initiative for Arabic Content [WWW document]. URL https://archive.org/details/KingAbdullahBinAbdulazizAl-saudInitiativeForArabicContent (accessed 2.10.19).

Jang, B.K., Chin, R.T., 1992. One-pass parallel thinning: Analysis, properties, and quantitative evaluation. IEEE Transactions on Pattern Analysis & Machine Intelligence, 14, pp. 1129–1140. DOI:10.1109/34.166630

Jumari, K., Ali, M., 2012. A survey and comparative evaluation of selected off-line Arabic handwritten character recognition systems. Jurnal Teknologi, 36(1), pp. 1–18.

Kandel, S., Orliaguet, J.P., Viviani, P., 2000. Perceptual anticipation in handwriting: The role of implicit motor competence. Perception & Psychophysics, 62(4), pp. 706–716. DOI:10.3758/BF03206917

Kannan, R.J., Subramanian, S., 2015. An adaptive approach of Tamil character recognition using deep learning with big data: A survey. In Emerging ICT for Bridging the Future-Proceedings of the 49th Annual Convention of the Computer Society of India (CSI). Springer, 1, pp. 557–567. DOI:10.1007/978-3-319-13728-5_63

Kanungo, T., Marton, G.A., Bulbul, O. 1999a. Performance evaluation of two Arabic OCR products. In 27th AIPR Workshop: Advances in Computer-Assisted Recognition. International Society for Optics and Photonics, pp. 76–83. DOI:10.1117/12.339809

Kanungo, T., Marton, G.A., Bulbul, O. 1999b. OmniPage vs. Sakhr: Paired model evaluation of two Arabic OCR products. In Document Recognition and Retrieval. International Society for Optics and Photonics, pp. 109–120.

Katz, S.M., 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. IEEE Transactions on Acoustics, Speech, and Signal Processing, 35(3), pp. 400–401. DOI:10.1109/TASSP.1987.1165125

Kef, M., Chergui, L., Chikhi, S., 2012. Comparative study of the use of geometrical moments for Arabic handwriting recognition. In: 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT). IEEE, pp. 303–308.

Khmelev, D.V, Teahan, W.J., 2003. A repetition based measure for verification of text collections and for text categorization. In: Proceeding of the 26th

Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 104–110.

Khorsheed, M.S., 2004. A lexicon based system with multiple HMMs to recognise typewritten and handwritten Arabic words. The 17th National Computer Conference, Madinah, Saudi Arabia. pp. 5–8.

Khorsheed, M.S., 2002. Off-line Arabic character recognition: A review. Pattern Analysis & Applications, 5(1), pp. 31-45. DOI:10.1007/s100440200004

Khorsheed, M.S., 2007. Offline recognition of omnifont Arabic text using the HMM ToolKit (HTK). Pattern Recognition Letters, 28(12), pp. 1563–1571. DOI:10.1016/j.patrec.2007.03.014

Khorsheed, M.S., 2015. Recognizing cursive typewritten text using segmentation-free system. The Scientific World Journal, 1, pp. 7–14.

Khorsheed, M.S., Clocksin, W.F., 1999. Structural features of cursive Arabic script. In: British Machine Vision Conference (BMVC). pp. 1–10.

Khorsheed, M., Clocksin, W., 2000. Multi-font Arabic word recognition using spectral features. In: 15th International Conference on Pattern Recognition. IEEE, 4, pp. 543–546.

Ko, D., Lee, C., Han, D., Ohk, H., Kang, K., Han, S., 2018. Approach for machine-printed Arabic character recognition: The-state-of-the-art deep-learning method. Electronic Imaging, pp. 1–8.

Kocyigit, P., 2012. Agent based optical character recognition, Master's dissertation. Prifysgol Bangor University, Bangor, UK.

Koteswara Rao, D., Negi, A., 2016. Advances in signal processing and intelligent recognition systems: Proceedings Second International Symposium on Signal Processing and Intelligent Recognition Systems (SIRS-2015) December 16-19, 2015, Trivandrum, India. In: Thampi, M.S., Bandyopadhyay, S., Krishnan, S., Li, K.-C., Mosin, S., Ma, M. (Eds.). Springer International Publishing, Cham, pp. 633–644. DOI:10.1007/978-3-319-28658-7_54

Krayem, A.G., 2013. A high level approach to Arabic sentence recognition. PhD thesis, Nottingham Trent University.

Kumar, G., Bhatia, P.K., 2014. A detailed review of feature extraction in image processing systems. In: International Conference on Advanced Computing and Communication Technologies (ACCT). IEEE, pp. 5–12. DOI:10.1109/ACCT.2014.74

Lam, L., Lee, S.W., 1992. Thinning methodologies—a comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(9), pp. 869–885. DOI:10.1109/34.161346

Lawgali, A., 2015. A survey on Arabic character recognition. International Journal of Signal Processing, Image Processing and Pattern Recognition, 8(2), pp. 401–426.

Levenshtein, V.I., 1966. Binary codes capable of correcting deletions, insertions, and reversals. In Soviet physics doklady, 10(8), pp. 707–710. DOI:citeulikearticle-id:311174

Liu, C.L., Fujisawa, H., 2008. Classification and learning methods for character recognition: Advances and remaining problems. In Machine Learning in Document Analysis and Recognition. Springer, pp. 139–161. DOI:10.1007/978-3-540-76280-5_6

Lorigo, L., Govindaraju, V., 2005. Segmentation and pre-recognition of Arabic handwriting. In: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp. 605–609. DOI:10.1109/ICDAR.2005.207

Lorigo, L.M., Govindaraju, V., 2006. Offline Arabic handwriting recognition: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(5), pp. 712–724. DOI:10.1109/TPAMI.2006.102

Luqman, H., Mahmoud, S.A., Awaida, S., 2014. KAFD Arabic font database. Pattern Recognition, 47, pp. 2231–2240. DOI:10.1016/j.patcog.2013.12.012

Magdy, W., Darwish, K., 2008. Effect of OCR error correction on Arabic retrieval. Information Retrieval, Boston, 11(5), pp. 405–425.

Mahmoud, S.A., 1994. Arabic character recognition using Fourier descriptors and character contour encoding. Pattern Recognition, 27(6), pp. 815–824. DOI:10.1016/0031-3203(94)90166-X

Mahmoudb, S.A., 1995. Survey and bibliography of Arabic optical text recognition Signal processing, 41(1), pp. 49-77.

Margner, V., El Abed, H., 2009. Arabic word and text recognition: Current developments. In: Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt, MEDAR Consortium.

Mari, J.F., 1989. Machine recognition and correction of printed Arabic text. IEEE Transactions on systems, man, and cybernetics, 19(5), pp. 1300–1306. DOI:10.1109/21.44052

Mehran, R., Pirsiavash, H., Razzazi, F., 2005. A front-end OCR for omni-font Persian/Arabic cursive printed documents. In Digital Image Computing: Techniques and Applications (DICTA'05). IEEE, p. 56.

Melhi, M., Ipson, S.S., Booth, W., 2001. A novel triangulation procedure for thinning hand-written text. Pattern Recognition Letters, 22(10), pp. 1059–1071. DOI:10.1016/S0167-8655(01)00038-1

Mihov, S., Schulz, K.U., Ringlstetter, C., Dojchinova, V., Nakova, V., Kalpakchieva, K., Gerasimov, O., Gotscharek, A., Gercke, C., 2005. A corpus for comparative evaluation of OCR software and postcorrection techniques. In the International Conference on Document Analysis and Recognition (ICDAR2005). IEEE, pp. 162–166. DOI:10.1109/ICDAR.2005.6

Moubtahij, E., Hicham, A.H., Satori, K., 2014. Review of feature extraction techniques for offline handwriting Arabic text recognition. International Journal of Advances in Engineering & Technology, 7(1), pp. 50–58.

Naccache, N.J., Shinghal, R., 1984. SPTA: A proposed algorithm for thinning binary patterns. IEEE ransactions on Systems, Man, and Cybernetics. SMC, 3, pp. 409–418. DOI:10.1109/TSMC.1984.6313233

Nagy, G., Seth, S., Viswanathan, M., 1992. A prototype document image analysis system for technical journals. Computer (Long Beach, CA), 25(7), pp. 10–22. DOI:10.1109/2.144436

Najoua, B.A., Noureddine, E., 1995. A robust approach for Arabic printed character segmentation. In: Proceedings of the Third International Conference on Document Analysis and Recognition. IEEE, 2, pp. 865–868.

Nartker, T.A., Rice, S.V, Lumos, S.E., 2005. Software tools and test data for research and testing of page-reading OCR systems. In: International Symposium on Electronic Imaging Science and Technology, pp. 37–48.

Nashwan, F., Rashwan, M., Al-Barhamtoshy, H., Abdou, S., Moussa, A., 2017. A holistic technique for an Arabic OCR system. Journal of Imaging, 4(1), pp. 6–17.

Natarajan, P., Saleem, S., Prasad, R., MacRostie, E., Subramanian, K., 2008. Multi-lingual offline handwriting recognition using hidden Markov models: A script-independent approach. In: Arabic and Chinese Handwriting Recognition. Springer, pp. 231–250. DOI:10.1007/978-3-540-78199-8_14

Nawaz, S.N., Sarfraz, M., Zidouri, A., Al-Khatib, W.G., 2003. An approach to offline Arabic character recognition using neural networks. In: Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems (ICECS). IEEE, 3. pp. 1328–1331. DOI:10.1109/ICECS.2003.1301760

Naz, S., Hayat, K., Imran Razzak, M., Waqas Anwar, M., Madani, S.A., Khan, S.U., 2014. The optical character recognition of Urdu-like cursive scripts. In: Pattern Recognition, 47(3), pp. 1229–1248. DOI:10.1016/j.patcog.2013.09.037

Naz, S., Umar, A.I., Shirazi, S.H., Ahmed, S.B., Razzak, M.I., Siddiqi, I., 2015. Segmentation techniques for recognition of Arabic-like scripts: A comprehensive survey. Education and Information Technologies, 21 (5), pp. 1–17.

Neuhaus, M., Bunke, H., 2007. A quadratic programming approach to the graph edit distance problem. Graph-Based Represent. Pattern Recognition. Springer, pp. 92–102. DOI:10.1007/978-3-540-72903-7_9

Nixon, M.S., Aguado, A.S., 2008. Feature extraction and image processing. Academic Press. DOI:10.1016/B978-0-12-396549-3.00001-X

O’Gorman, L., 1993. The document spectrum for page layout analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(11), 1162–1173. DOI:10.1109/34.244677

Omar, K., Mahmod, R. bin Sulaiman, M. N., bin Ramli, A. R., 2000. The removal of secondaries of Jawi characters. 2000 TENCON Proceedings. Intelligent Systems and Technologies for the New Millennium (Cat.

No.00CH37119). IEEE, 2, pp. 149–152. DOI:10.1109/TENCON.2000.888408

Optical Character Recognition (OCR) [WWW document], n.d. URL http://www.sakhr.com/index.php/en/solutions/ocr (accessed 4.15.17).

Otus, N., 1979. A threshold selection method from gray-level histograms. IEEE Transactions on Systems, 9(1), pp. 62–66.

Oujaoura, M., El Ayachi, R., Fakir, M., Bouikhalene, B., Minaoui, B., 2012. Zernike moments and neural networks for recognition of isolated Arabic characters. International Journal of Computer Engineering Science, 2(3), pp. 17–25.

Pal, U., Wakabayashi, T., Kimura, F., 2009. Comparative study of Devnagari handwritten character recognition using different features and classifiers. In: 10th International Conference on Document Analysis and Recognition. IEEE, pp. 1111–1115. DOI:10.1109/ICDAR.2009.244

Parhami, B., Taraghi, M., 1981. Automatic recognition of printed Farsi texts. Pattern Recognition, 14, pp. 395–403.

Parker, J.R., 2011. Algorithms for image processing and computer vision. John Wiley & Sons. DOI:10.1002/1521-3773(20010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C

Parvez, M.T., Mahmoud, S.A., 2013. Offline Arabic handwritten text recognition: A survey. ACM Computing Surveys, 45(2), pp. 23:1–23:35. DOI:10.1145/2431211.2431222

Pati, P.B., Ramakrishnan, A.G., 2005. OCR in Indian scripts: A survey. IETE Technical Review, 22(3), pp. 217–227.

Pavlidis, T., 1993. Recognition of printed text under realistic conditions. Pattern Recognition Letters, 14(4), pp. 317–326. DOI:10.1016/0167-8655(93)90097-W

Pechwitz, M., Margner, V., 2002. Baseline estimation for Arabic handwritten words. In: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR). IEEE, pp. 479–484. DOI:10.1109/IWFHR.2002.1030956

Pi, Y., Liao, W., Liu, M., Lu, J., 2008. Theory of cognitive pattern recognition. In Pattern Recognition Techniques, Technology and Applications. InTech. DOI:10.5772/6251

Plamondon, R., Srihari, S.N., 2000. On-line and off-line handwriting recognition: A comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1), pp. 63–84. DOI:10.1109/34.824821

Prasad, R., Saleem, S., Kamali, M., Meermeier, R., Natarajan, P., 2008. Improvements in hidden Markov model based Arabic OCR. In: Proceedings of the19th International Conference on Pattern Recognition. IEEE, pp. 1–4. DOI:10.1109/ICPR.2008.4761446

Radwan, M.A., Khalil, M.I., Abbas, H.M., 2018. Neural networks pipeline for offline machine printed Arabic OCR. Neural Processing Letters, 48(2), pp. 769–787. DOI:10.1007/s11063-017-9727-y

Rahal, N., Tounsi, M., Alimi, A.M., 2018. Auto-Encoder-BoF/HMM System for Arabic Text Recognition. arXiv Preprint.

Rahman, A.F.R., Fairhurst, M.C., 2003. Multiple classifier decision combination strategies for character recognition: A review. Document Analysis and Recognition. Springer, 5(4), pp. 166–194. DOI:10.1007/s10032-002-0090-8

Rashid, S.F., 2014. Optical character recognition: A combined ANN/HMM approach. PhD thesis, Technische Universität Kaiserslautern.

Rashid, S.F., Schambach, M.-P., Rottland, J., von der Nüll, S., 2013. Low resolution Arabic recognition with multidimensional recurrent neural networks. In: Proceedings of the 4th International Workshop on Multilingual OCR. ACM.

Rice, S.V., 1996. Measuring the accuracy of page-reading systems. University of Nevada, Las Vegas.

Saabni, R., 2015. Efficient recognition of machine printed Arabic text using partial segmentation and Hausdorff distance. In: the 6th International Conference of Soft Computing and Pattern Recognition (SoCPaR). IEEE, pp.284–289. DOI:10.1109/SOCPAR.2014.7008020

Saad, M., Ashour, W., 2010. OSAC: Open Source Arabic Corpora. In: the 6th International Symposium on Electrical and Electronics Engineering and Computer Science (EECS'10), Lefke, Cyprus. pp. 118–123. DOI:10.13140/2.1.4664.9288

Sabbour, N., Shafait, F., 2013. A segmentation-free approach to Arabic and Urdu OCR. In Document Recognition and Retrieval. International Society for Optics and Photonics, pp. 86580N–86580N-12.

Saber, S., Ahmed, A., Elsisi, A., Hadhoud, M., 2016. Performance evaluation of Arabic optical character recognition engines for noisy inputs. In: 1st International Conference on Advanced Intelligent System and Informatics (AISI2015). Springer, Beni Suef, Egypt, pp. 449–459.

Saber, S., Ahmed, A., Hadhoud, M., 2014. Robust metrics for evaluating Arabic OCR systems. In: First International Image Processing, Applications and Systems Conference (IPAS). IEEE, pp. 1–6. DOI:10.1109/IPAS.2014.7043272

Saeed, K., Tabędzki, M., Rybnik, M., Adamski, M., 2010. K3M: A universal algorithm for image skeletonization and a review of thinning techniques. International Journal of Applied Mathematics and Computer Science, 20(2), 317–335. DOI:10.2478/v10006-010-0024-4

Sarfraz, M., Nawaz, S.N., Al-Khuraidly, A., 2003. Offline Arabic text recognition system. In: Proceedings International Conference on Geometric Modeling and Graphics. IEEE, pp. 30–35. DOI:10.1109/GMAG.2003.1219662

Saudagar, A.K.J., Mohammed, H.V., 2016. Open CV based implementation of Zhang-Suen thinning algorithm using Java for Arabic text recognition. In: Information Systems Design and Intelligent Applications. Springer, pp. 265–271.

Sawy, A. El, El-Bakry, H., Loey, M., 2017. CNN for handwritten Arabic digits recognition based on LeNet-5. In: International Conference on Advances Intelligent Systems and Informatics. Springer.

Shaaban, Z., 2008. A new recognition scheme for machine-printed Arabic texts based on neural networks. In: Proceedings of World Academy of Science, Engineering and Technology, pp. 25–27.

Shafait, F., Adnan-ul-Hasan, Keysers, D., Breuel, T.M., 2006. Layout analysis of Urdu document images. In: 10th IEEE International Multitopic Conference (INMIC). IEEE, pp. 293–298. DOI:10.1109/INMIC.2006.358180

Shafii, M., 2014. Optical character recognition of printed Persian/Arabic documents. PhD thesis, University of Windsor.

Shaker, A.S., 2018. Recognition of Off-line Printed Arabic Text Using Hidden Markov Models. Ibn AL-Haitham Journal for Pure and Applied Science, 31(2), 230–238.

Sharma, D.V., Saini, G., Joshi, M., 2012. Statistical feature extraction methods for isolated handwritten Gurumukhi script. International Journal of Engineering Research and Application, 2, 380–384.

Shi, Z., Setlur, S., Govindaraju, V., 2012. Guide to OCR for Arabic scripts. In: Märgner, V., El Abed, H. (Eds.), Springer London, London, pp. 79–102. DOI:10.1007/978-1-4471-4072-6_4

Slimane, F., Ingold, R., Hennebert, J., 2018. ICDAR 2017 competition on multi-font and multi-size digitally represented Arabic text. In: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR). IEEE, 1, pp. 1466–1472. DOI:10.1109/ICDAR.2017.239

Slimane, F., Ingold, R., Kanoun, S., Alimi, A.M., Hennebert, J., 2009. A new Arabic printed text image database and evaluation protocols. In: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp. 946–950. DOI:10.1109/ICDAR.2009.155

Slimane, F., Kanoun, S., El-Abed, H., Alimi, A.M., Ingold, R., Hennebert, J., 2011. ICDAR 2011: Arabic recognition competition: Multi-font multi-size digitally represented text. In: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp. 1449-1453. DOI:10.1109/ICDAR.2011.288

Slimane, F., Kanoun, S., Hennebert, J., Alimi, A.M., Ingold, R., 2013. A study on font-family and font-size recognition applied to Arabic word images at ultra-low resolution. Pattern Recognition Letters, 34(2), pp. 209–218. DOI:10.1016/j.patrec.2012.09.012

Smith, R., 2007. An overview of the Tesseract OCR engine. In: Proceedings of the International Conference on Document Analysis and Recognition, IEEE.

Smith, R., Antonova, D., Lee, D.-S., 2009. Adapting the Tesseract open source OCR engine for multilingual OCR. In: Proceedings of the International Workshop on Multilingual OCR. ACM.

Srihari, S.N., Ball, G., 2012. An assessment of Arabic handwriting recognition technology. In: Guide to OCR for Arabic Scripts. Springer, pp. 3–34.

Srinivas, B.A., Agarwal, A., Rao, C.R., 2008. An overview of OCR research in Indian scripts. International Journal of Computer Sciences and Engineering Systems (IJCSES), 2(2), pp. 141–153.

Stolyarenko, A. V, Dershowitz, N., 2018. OCR for arabic using SIFT descriptors with online failure prediction. Tel Aviv University.

Sun, C., Si, D., 1997. Skew and slant correction for document images using gradient direction. In: Proceedings of the Fourth International Conference on Document Analysis and Recognition. IEEE, 1, pp. 142–146. DOI:10.1109/ICDAR.1997.619830

Supriana, I., Nasution, A., 2013. Arabic character recognition system development. Procedia Technology, 11, pp. 334–341. DOI:10.1016/j.protcy.2013.12.199

Taghva, K., Stofsky, E., 2001. OCRSpell: An interactive spelling correction system for OCR errors in text. International Journal on Document Analysis and Recognition, 3, pp. 125–137. DOI:10.1007/PL00013558

Taha, S., Babiker, Y., Abbas, M., 2012. Optical character recognition of Arabic printed text. In: IEEE Student Conference on Research and Development (SCOReD). IEEE, pp. 235–240. DOI:10.1109/SCOReD.2012.6518645

Tamen, Z., Drias, H., 2010. How to overcome some segmentation problems in a constrained handwritten Arabic character recognition system. In: 10th International Conference on Information Sciences, Signal Processing and Their Applications, (ISSPA 2010). IEEE, pp. 634–637. DOI:10.1109/ISSPA.2010.5605419

Tarábek, P., 2008. Performance measurements of thinning algorithms. Journal of Information, Control and Management, 6(2), pp. 125–132.

Teahan, W.J., 2000. Text classification and segmentation using minimum cross-entropy. In: Content-Based Multimedia Information Access, 2, pp. 943-961.

Teahan, W., 2018. A compression-based toolkit for modelling and processing natural language text. Information, 9(12), pp.294-323. DOI:10.3390/info9120294

Teahan, W.J., Harper, D.J., 2003. Using compression-based language models for text categorization. Language Modeling for Information Retrieval. Springer, pp. 141-165. DOI:10.1007/s10639-015-9377-5

Teahan, W.J., Inglis, S., Cleary, J.G., Holmes, G., 1998. Correcting English text using PPM models. In: Proceedings Data Compression Conference. IEEE, pp. 289-298.

Teahan, W.J., Wen, Y., McNab, R., Witten, I.H., 2000. A compression-based algorithm for Chinese word segmentation. Computational Linguistic, 26(3), 375–393.

Touj, S., ben Amara, N.E., Amiri, H., 2005. Generalized Hough transform for Arabic printed optical character recognition. International Arab Journal of Information Technology, 2(4), pp. 326–333.

Tse, P.U., Cavanagh, P., 2000. Chinese and Americans see opposite apparent motions in a Chinese character. Cognition, 74(3), pp. 27–32. DOI:10.1016/S0010-0277(99)00065-7

Viterbi, A., 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory, 13(2), pp.260-269.

Vinter, A., Chartrel, E., 2010. Effects of different types of learning on handwriting movements in young children. Learning and Instruction, 20(6), pp. 476–486.

Wang, P.S.P., Zhang, Y.Y., 1989. A fast and flexible thinning algorithm. IEEE Transactions on Computers, 38(5), pp. 741–745. DOI:10.1109/12.24276

Witten, I. H., Frank, E., Hall, M. A., 2005. Data mining: Practical machine learning tools and techniques. In: Data mining, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Wu, P., 2007. Adaptive models of Chinese text. Prifysgol Bangor University of Wales.

Xiu, P., Peng, L., Ding, X., Wang, H., 2006. Offline handwritten Arabic character segmentation with probabilistic model. In: Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer, pp. 402–412. DOI:10.1007/11669487_36

Ymin, A., Aoki, Y., 1996. On the segmentation of multi-font printed Uygur scripts. In: Proceedings of the 13th International Conference on Pattern Recognition. IEEE, 3, pp. 215–219.

Younis, K.S., Alkhateeb, A.A., 2017. A new implementation of deep neural networks for optical character recognition and face recognition. In: Proceedings of the New Trends in Information Technology. Jordan, 157–162.

Zeki, A.M., 2005. The segmentation problem in Arabic character recognition: The state of the art. In: Proceedings of the 1st International Conference on Information and Communication Technology (ICICT). IEEE, pp. 11–26. DOI:10.1109/ICICT.2005.1598538

Zeki, A.M., Zakaria, M.S., Liong, C.-Y., 2011. Segmentation of Arabic characters: A comprehensive survey. International Journal of Technology Diffusion, 2(4), pp. 48–82. DOI:http://dx.doi.org/10.4018/jtd.2011100104

Zhang, D., Lu, G., 2001. A comparative study on shape retrieval using Fourier descriptors with different shape signatures. In: Proceedings of International Conference on Intelligent Multimedia and Distance Education, 1, pp. 1–9.

Zhang, T.Y., Suen, C.Y., 1984. A fast parallel algorithm for thinning digital patterns. Communications of th*e* ACM, 27(3), pp. 236–239.

Zhang, Y.Y., Wang, P.S.P., 1988. A modified parallel thinning algorithm. In: Proceedings of the 9th international Conference on Pattern Recognition. IEEE, pp. 1023–1025. DOI:10.1109/ICPR.1988.28429

Zheng, L., Hassin, A.H., Tang, X., 2004. A new algorithm for machine printed Arabic character segmentation. Pattern Recognition Letters, 25(15), pp. 1723–1729. DOI:10.1016/j.patrec.2004.06.015

Zhou, R.W., Quek, C., Ng, G.S., 1995. A novel single-pass thinning algorithm and an effective set of performance criteria. Pattern Recognition Letters, 16(12), pp. 1267–1275.

Zidouri, A., Sarfraz, M., Nawaz, S.N., Ahmad, M.J., 2003. PC based offline Arabic text recognition system. In: Proceedings of the Seventh International Symposium on Signal Processing and Its Applications. IEEE, pp. 431–43 DOI:10.1109/ISSPA.2003.1224906

Appendices

Printed Arabic Script Recognition: A Survey

Mansoor Alghamdi
Department of Computer Science
Community College
University of Tabuk
Tabuk, Saudi Arabia

William Teahan
School of Computer Science
Bangor University
United Kingdom

***Abstract*—Optical character recognition (OCR) is essential in various real-world applications, such as digitizing learning resources to assist visually impaired people and transforming printed resources into electronic media. However, the development of OCR for printed Arabic script is a challenging task. These challenges are due to the specific characteristics of Arabic script. Therefore, different methods have been proposed for developing Arabic OCR systems, and this paper aims to provide a comprehensive review of these methods. This paper also discusses relevant issues of printed Arabic OCR including the challenges of printed Arabic script and performance evaluation. It concludes with a discussion of the current status of printed Arabic OCR, analyzing the remaining problems in the field of printed Arabic OCR and providing several directions for future research.**

Keywords—Optical character recognition; arabic printed OCR; arabic text recognition; arabic OCR survey; feature extraction; segmentation; classification

I. Introduction

Optical Character Recognition (OCR) is a technique that transforms a printed or handwritten text image into an electronic format. OCR development is considered a challenging task in the field of pattern recognition. Many OCR approaches have been proposed for Latin and non-Latin scripts. However, printed Arabic OCR still poses great challenges because of the special characteristics of Arabic script [1].

Arabic OCR is highly desirable in various real-world applications, such as digitising learning resources to assist visually impaired people, bank cheque processing and mail sorting[2], [3]. Furthermore, there are many initiatives for Arabic digital content enrichment [4]. One of these initiatives is King Abdullah's Initiative for Arabic Content. Therefore, a robust and efficient Arabic OCR is required to support this initiative by increasing Arabic content on the Internet.

Numerous methods have been proposed for recognising printed Arabic script from an image, yet we are unaware of comprehensive surveys of printed Arabic OCR during the last fifteen years. Two surveys have been conducted on printed Arabic OCR [2], [5]. However, these reviews do not reflect the current progress in printed Arabic OCR. Therefore, establishing a guide and baseline for future directions remains important for Arabic OCR researchers.

This work will establish this guide and baseline for Arabic OCR researchers by providing a comprehensive literature review of printed Arabic text recognition research. It reviews techniques that have been utilized for developing printed Arabic OCR with emphasis on the issues related to Arabic script. It also highlights the current status of printed Arabic OCR and provides several directions for future research.

This paper is organised as follows. In section 2, Arabic script characteristics and challenges are discussed. Section 3 presents the methodologies of printed Arabic OCR, with subsections that review the five stages of the development of printed Arabic OCR: preprocessing, segmentation, feature extraction, classification and post-processing. Section 4 discuses performance evaluation issues of printed Arabic OCR. Section 5 concludes with a discussion about open problems and future directions.

II. Arabic Script Characteristics and Challenges

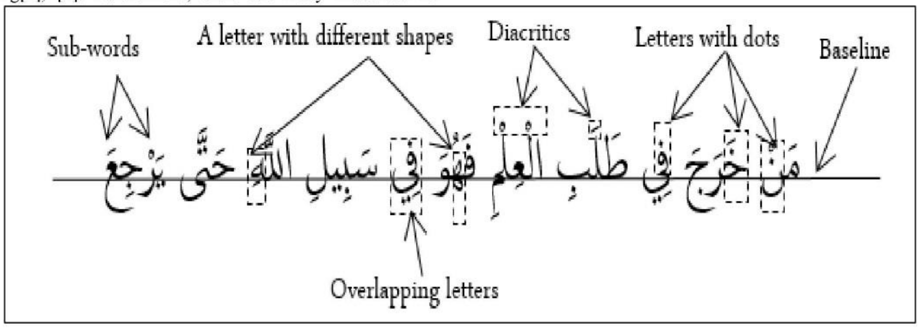

Sub-words
A letter with different shapes
Diacritics
Letters with dots
Baseline
مَنْ خَرَجَ فِي طَلَبِ الْعِلْمِ فَهُوَ فِي سَبِيلِ اللَّهِ حَتَّى يَرْجِعَ
Overlapping letters

Fig. 1. Arabic script characteristics

TABLE I. ARABIC CHARACTERS WITH DIFFERENT POSITIONS AND SHAPES

Isolated	Initial	Middle	End
ا	ا	ـا	ـا
ب	بـ	ـبـ	ـب
ت	تـ	ـتـ	ـت
ث	ثـ	ـثـ	ـث
ج	جـ	ـجـ	ـج
ح	حـ	ـحـ	ـح
خ	خـ	ـخـ	ـخ
د	د	ـد	ـد
ذ	ذ	ـذ	ـذ
ر	ر	ـر	ـر
ز	ز	ـز	ـز
س	سـ	ـسـ	ـس
ش	شـ	ـشـ	ـش
ص	صـ	ـصـ	ـص
ض	ضـ	ـضـ	ـض
ط	ط	ـطـ	ـط
ظ	ظ	ـظـ	ـظ
ع	عـ	ـعـ	ـع
غ	غـ	ـغـ	ـغ
ف	فـ	ـفـ	ـف
ق	قـ	ـقـ	ـق
ك	كـ	ـكـ	ـك
ل	لـ	ـلـ	ـل
م	مـ	ـمـ	ـم
ن	نـ	ـنـ	ـن
ه	هـ	ـهـ	ـه
و	و	ـو	ـو
ي	يـ	ـيـ	ـي

There is no doubt that printed Arabic OCR faces a number of challenges and there is still an intensive need for more research [6]. However, most challenges facing the development of Arabic OCR are due to the characteristics of Arabic script. Arabic script has some features that distinguish it from other languages. Compared to English, the most obvious feature of Arabic script is that it is written cursively from right to left in both printed and handwritten. The greatest challenges are due to the more complex characteristics of Arabic script. In the following section, the characteristics of Arabic script that may complicate recognition will be discussed:

A. Shapes and Positions

The Arabic alphabet has 28 basic letters (see Table 1). However, an Arabic letter may contain four dissimilar shapes in relation to its location inside a word: whether it is an isolated letter, an initial letter (in which a letter is inked from the right side, an ending letter (in which a letter is linked form the left side) or a middle letter (in which a letter is linked from the right and left sides). Thus, the number of letters to be recognized will increase from 28 letters to 125 letters.

B. Overlapping characters and Ligatures

Characters in an Arabic word might be overlapped vertically with or without touching each other (see Figure 1). In particular, some characters are combined and written as a ligatures such as (لا) which is a combination of two letters Lam (ل) and Alf (ا). However, ligatures occurs in Arabic script depending on the type of fonts being used. For instance, in Traditional Arabic font, there are about 220 ligatures whereas Simplified Arabic incorporates about 150 ligatures, [7].

C. Diacritics

Characters in an Arabic word can exist with diacritics or short vowels such as Fat-hah, Dhammah, Mada'ah, Kasrah and Sukkun, as illustrated in Figure 1. These can be placed either over or below the letters as strokes. In addition, Tanwen is considered as a diacritic which is indicated by double Fat-hah, double Dhammah and double Kasrah. One more diacritic that Arabic script has is Shaddah which is similar to the number 3 as it is rotated 90° clockwise.

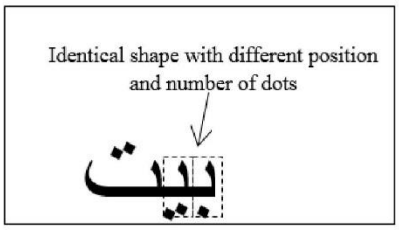


Fig. 2. Two characters (*Ba* and *Ya*) with an identical shape and a different number of dots.

D. Cursive

As mentioned above, Arabic script is a cursive script which means that a word is composed of connected characters. However, six characters (ا, د, ذ, ر, ز, و) of the Arabic alphabet are not linked with succeeding letters. This can present a challenge because these characters can divide a word into one or more units as sub-words (see Figure 1).

E. Presence of dots

The Arabic alphabet relies on number and position of dots in order to differentiate between similar letters (see Figure 2). Fifteen characters in the Arabic alphabet have dots. They can be placed below the character, above it or in the middle. Ten of these characters are dotless, three have two dots and two have three dots, as shown in Table 1.

III. GENERAL ARABIC OCR METHODOLOGY (MODEL)

This section will focus on the methodologies used by printed Arabic OCR systems. Published approaches and systems for Arabic OCR indicate that the process of implementing Arabic OCR consists of five phases: (1) pre-processing; (2) segmentation; (3) feature extraction; (4) classification and (5) post-processing, see figure 3.

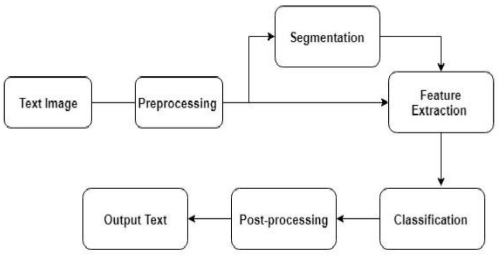


Fig. 3. General printed Arabic OCR methodology.

A. Preprocessing Phase

This is the first phase of OCR methodology which is responsible for enhancing the readability of the input image. Preprocessing is a combination of algorithms that are applied to the input image in order to reduce noise and alterations, thus simplifying the subsequent phases of OCR methodology [2]. There are various factors that affect the quality of the input image. A study lists the history of image, the printing process, the kind of font, the quality of paper, the condition of the image and the image acquisition as the vital factors that influence the input image quality [2].

Researchers emphasize that the downstream OCR accuracy relies on the quality of the input image [8]. Furthermore, a study states that OCR systems, which report high recognition accuracy on some input images, will report less recognition accuracy on input images that are poor in quality [9]. Thus, the preprocessing phase is a critical stage in OCR development that simplifies the data for the subsequent phases to operate accurately. Generally, several preprocessing operations are employed on the input image: binarization, layout analysis, thinning, smoothing and filtering, size and slant normalization, slant detection, skew detection and baseline detection. However, the selection of these operations, to be applied in the preprocessing, relies upon the conditions of the input image, such as the amount of noise and skew in the input image [10]. In the following section, the preprocessing techniques which are applied in Arabic OCR, will be clarified.

1) Binarization

For character recognition, the binarization (sometimes called thresholding) process involves converting an input gray scale image into a binary image, in which a pixel has only two values 0 and 1. The binary image has the critical information, such as the shape of characters. It has been found that increasing processing speed and reducing storage capacity are the key benefits of binarization technique [7], [11]. Researchers suggest selecting the most appropriate method for binarization might separate connected objects or joining isolated objects [12]. A number of studies have confirmed the efficiency of computing the histogram of the gray scale of an image and then detecting a cut-off point as the binarization method [13] [12]. However, some researchers work on recognition without applying binarization methods, such as [14], [15].

2) Size Normalization

Since Arabic characters differ in size, as described earlier, size normalization is commonly applied to characters or words by scaling the characters or the words to an adjusted size. This process is crucial for the recognition or classification phase, since some recognition methods are sensitive to dissimilarity in size and position, such as template matching and correlation approaches [16]. A study classified normalization methods into two approaches: moment-based normalization; and nonlinear normalization [17]. It is argued that normally a character is normalized to a standard size for classification [18]. However, in terms of word normalization, applying normalization to a word instead of a character will result in losing critical information [18], [19].

3) De-noising

Noise may be presented during the acquisition process via scanners which results in distortions and variations in the input text image. Besides this, very small items in the text image can be reflected as noise [11], which are byproducts of image scanning or binarization and which are not parts of the text. Such noise may has a major impact on the performance of OCR systems [20]. Noise removal is an operation for enhancing the visual quality of the input image [21].

As a solution, several techniques have been introduced that are considered as noise removal methods [22]. These methods include filtering and morphological operations (smoothing) which are conditioning processes in terms of OCR development [2], [23]; for instance, dilation algorithms, which are applied to broken letters, and erosion algorithms which are applied to text images with touching letters [17]. In addition, the median filter approach is commonly used in both printed text images and handwritten text images. For example, a study apply this approach for removing noise in printed Arabic text images [24]. Another example of a study that applies a median filter algorithm in handwritten Arabic text images is in [25].other researchers [26] applied a morphological noise removal method for Arabic printed OCR proposed in [27]. However, a study discovered that letter holes could be filled while applying this method, with lower thresholds, to Arabic text images [26].

In fact, the review suffers from the fact that some printed Arabic OCR studies applied noise removal algorithms without providing information of the applied algorithm, for instance, in [28]. Such approaches, however, should be selected carefully when considering OCR systems. That is, because of the similarity between Arabic letters, any alteration of a letter might change it to another letter. Thus, a perfect noise removal method is able to eliminate noise while preserving the shape of the character [20].

4) Skew Detection and Correction

Initially, a text image has zero rotation, yet when physically scanning the image manually, rotation of images up to 20° might occur [5]. This rotation is called skew which results in non-zero skew text images (see Figure 4). The skew can lead to incorrect recognition and baseline detection [29]. It is impossible to segment a text if the text is rotated [30]. As a result, detecting and correcting the skew is critical to OCR applications that rely on segmentation approaches to recognize characters.

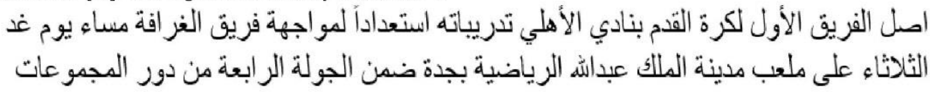

اصل الفريق الأول لكرة القدم بنادي الأهلي تدريباته استعداداً لمواجهة فريق الغرافة مساء يوم غد
الثلاثاء على ملعب مدينة الملك عبدالله الرياضية بجدة ضمن الجولة الرابعة من دور المجموعات

Fig. 4. An example of Arabic text skew.

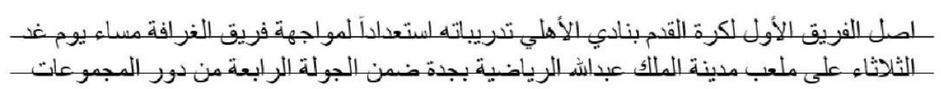

اصل الفريق الأول لكرة القدم بنادي الأهلي تدريباته استعداداً لمواجهة فريق الغرافة مساء يوم غد
الثلاثاء على ملعب مدينة الملك عبدالله الرياضية بجدة ضمن الجولة الرابعة من دور المجموعات

Fig. 5. Baseline detection of a printed Arabic text image.

The process of estimating the skew angle is known as skew detection, whereas the process of rotating the image with the purpose of correcting the skew is called skew correction. A wide variety of skew detection and correction methods have been proposed. A study groups these methods into five groups: projection profile, Hough transform, Fourier transform, nearest neighbor clustering and correlation [31]. The Hough transform is the standard approach for detecting the skew [32]. A method based on the projection profile was introduced in [33]. A researcher has provided a comprehensive review of twenty–five skew detection and correction approaches [34]. The author concludes that further work on more sophisticated methods is still required. The Radon transform method has shown its efficiency for skew correction [35]. Some methods are designed for specific applications and image type. For example, a new method has emerged for Arabic text images in [36]. One study concludes that selecting a skew detection and correction method relies on the image type [37].

5) Baseline Detection

As described in the previous section, Arabic characters are joined through a horizontal line called the baseline (see Figure 5). Graphically, the baseline can be described as the line which has the maximal amount of black pixels [38]. This line contains critical information about the text, such as text orientation and position of connection points between Arabic letters [2]. Thus, detecting the baseline is beneficial for many OCR stages, for instance, skew normalization [39], segmentation [40], [41] and structural features extraction such as the character's dots [42].

It has been reported that most Arabic OCR has applied baseline detection methods as a preprocessing step [25] .The baseline detection techniques for Arabic script has been classified into four groups in [36]; namely, horizontal projection methods, the word skeleton method, contour tracing and principle component analysis. Among these, the horizontal projection technique is widely implemented for determine the baseline in Arabic OCR, such as in [43], [2]. Several studies implement a horizontal projection approach in OCR systems for detecting the baseline, such as in [26], [42], [44], [45]. It has been emphasized that the horizontal projection method is simple and efficient for Arabic printed text [43], [46]. However, this method is applicable only for noise-free images, as it fails for unclean images [47].

Another baseline detection approach is the x-y cut proposed in [48] which is based on a horizontal projection method. This method works well for Arabic noisy images, though it fails in the presence of large amounts of noise and skew [47]. Consequently, researchers proposed using a ridge-based text line detecting approach for Arabic text [47]. The former method's efficiency has been tested and recommended for different types of Arabic text images, since it was found to achieve above 96% text line detection accuracy [47].

Researchers summarize the state of the art of baseline detection methods in Arabic script [49]. In summary, for printed Arabic text, the standard horizontal projection method is sufficient for detecting the baseline, since the baseline in printed text is straight. Whereas for handwriting, the baseline is not straight, thus more sophisticated approaches should be considered [50].

6) Thinning and Skeletonization

Thinning " skeletonization" can be defined as "the process of peeling off a pattern as many pixels as possible without affecting the general shape of the pattern" [18]. In other words, it involves operations that can be implemented in order to produce the skeleton of text images. Thinning is a crucial processing step for text recognition, in particular for such OCR applications in which extracting the skeleton of a character is essential [2], [32]. However, in terms of the obtaining the skeleton, it must be as thin as possible, connected, and centered [18]. Thinning simplifies the process of the segmentation, future extraction and classification phases as a result of reducing the amount of data that needs to be considered in the input image [25].

Most of the existing thinning algorithms have been designed for general purpose or other text languages [51]–[53]. However, when applying thinning algorithms to Arabic scripts, various obstacles are encountered [49]. One problem is the reduction in the number of dots in some Arabic characters as a result of the thinning process for which the number of dots is a crucial aspect in differentiating between these characters [54].

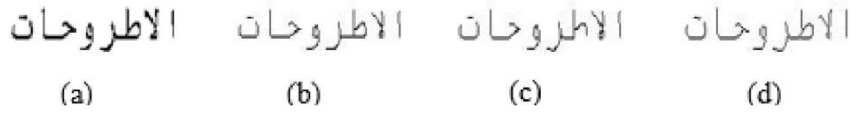


Fig. 6. Example results of different thinning algorithms: (a) original word, (b), (c) and (d) thinned word.

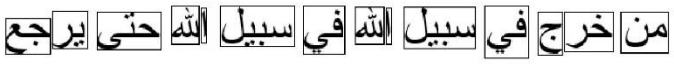


Fig. 7. An example of line segmentation.

Also, dots in Arabic characters are likely to be vulnerable to noise. However, some researchers extract dots of Arabic characters before applying thinning algorithms, in order to overcome this problem [55], [12]. Another problem of thinning algorithms when considering Arabic script concerns preserving the connectedness of Arabic text. Some thinning approaches may not cope well with Arabic text due to its connectivity characteristic [49]. Thus, this should be taken into consideration, when selecting thinning algorithms for Arabic text. Also, since Arabic characters consist of different shapes such as loops and lines, the selected thinning algorithm must be capable of preserving these different shapes.

Therefore as a consequence of specific characteristics of Arabic script detailed above, direct adoption of thinning algorithms, which have been developed for other languages, may not be as effective [24]. As a result of these difficulties, there is comparatively little published work on developing thinning algorithms for Arabic [25], [24].

Some studies introduce thinning algorithms for Arabic letters [56], [57]. However, the proposed algorithms can only deal with isolated Arabic characters. One study provide a thinning algorithm which is designed specifically for printed Arabic script recognition to overcome dis-connectivity and loss of information [58]. This algorithm is applied on Arabic text to illustrate the efficiency of reducing the outline of each word's characters (its number of pixels) thereby overcoming the challenges of Arabic script. Also, the authors propose an experimental framework with new performance measures for the evaluation of thinning algorithms. Figure 6 shows the output of three different thinning algorithms.

B. Segmentation Phase

After the preprocessing phase, an enhanced text image in the sense of low noise and variation, and a necessary amount of character information [2], has been produced. During the segmentation phase, the text image is segmented into small components, with a page being segmented into lines, a line into words and a word into letters [59]–[61]. Segmentation is a crucial step in Arabic OCR system development because of the fact that it plays a vital role in ensuring the success of the subsequent feature extraction and classification stages [3], [46]. However, the author in [46] stresses that misrecognition can arise by applying a poor segmentation method. As a result, this stage will have a critical impact on the recognition rate of the text [7].

As explained previously, one of the main challenges facing Arabic OCR development is the cursiveness of Arabic script. Segmentation of Arabic text thus can be more difficult and time consuming for the development of Arabic OCR systems [3]. Correspondingly, segmentation has been considered as the main contribution for increasing the recognition error rate in Arabic OCR systems [46], [62], [63].

Generally, segmenting a text image can be graded into two types: external segmentation; and internal segmentation [64]. While the former type deals with the isolation of different writing objects such as, paragraphs, sentences and words, the latter deals with the isolation of characters [64], [65].

1) External Segmentation

External segmentation refers to the document layout analysis, in particular page decomposition. Document layout analysis is accomplished in order to identify the physical structure of a page [66]. As far as offline OCR development is concerned, page analysis is a basic step which segments the image into its different logical parts with the identical type of information, such as graphs, text and tables. Page layout analysis is performed in two approaches: structural analysis by which a page is decomposed into blocks of the page elements, such as paragraphs and words; and functional analysis by which a page is decomposed into functional elements such as title and abstract [41], [65], [66].

With respect to Arabic document processing, page decomposition refers to the isolation of text lines of a texture region and the segmentation of words and sub-words [5], [7], [67], since it is restricted to text images [5]. Applying a fixed threshold to Arabic text documents to determine text lines is the standard method [5], [68], [69]. However, this method fails with a skewed text image [40].

Methods based on histogram projection are considered as conventional approaches for isolating lines and words in Arabic text documents [68], [70]. Several studies have relied on horizontal projection techniques for segmenting Arabic text images into lines, such as in [71]–[76], [28]. [72] It is recommended horizontal projection be applied for text images because of its advantages in reducing computational load and its simplicity of implementation [71]. Moreover, horizontal projection is an appropriate method for locating text lines in Arabic printed text, since the text lines in printed text are straight [50].

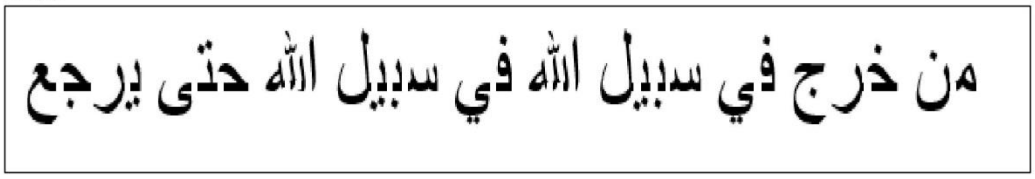

من خرج في سبيل الله حتى يرجع

Fig. 8. Segmenting Arabic words into their characters.

For line segmentation, researchers in Arabic OCR determine words in a line of text by inspecting the vertical projection [14], [28]. (See Figure 7). This method depends on the estimation of the minimum space between words. However, it was pointed out in the Arabic script characteristics section above that some Arabic characters are not linked with succeeding letters, thus this results in a word having with one or more connected components (sub-words), as shown in

Figure 7. To overcome this issue, methods based on vertical projection consider that the width of spaces between sub words is smaller than the width of the spaces between words [14].

Generally, it is relatively easy to segment a text line into words in printed text images, compared to handwritten text images which involve overlapping and touching characters by using vertical projection histogram profiles [37], [59], [25]. However, some Arabic fonts contain characters that vertically overlap, such as the Traditional font type. Thus, Arabic script even in printed form can contain touching and overlapping characters, so algorithms that have been designed to overcome this challenge for handwritten script may be utilized for printed Arabic. For example, the authors in [77] have developed a method based on the connected components that analyses the distance between connected components in order to segment handwritten words.

2) Internal segmentation

Internal segmentation deals with segmenting a word into characters. When reviewing segmentation methods in the literature, a major complication arises concerning the classification of word segmentation approaches. For instance,[5] a study classifies Arabic OCR systems based on word segmentation into 'segmentation based systems', which is based on analytical techniques where a word is segmented into characters, and 'segmentation–free systems', which is based on recognizing a word as a unit without segmentation [72]. Some researchers discuss word segmentation in terms of implicit and explicit segmentation [73], [78]. Others classify word segmentation in terms of techniques which have been applied to segmenting a word, such as [59], [46], [3]. Researchers organize segmentation methods for Arabic script into holistic approaches and analytical approaches [25].

Mostly, Arabic OCR systems have been developed by two main paradigms: holistic approaches (segmentation–free) which require a large lexicon of Arabic words, and analytical approaches (segmentation based) where a word is segmented into units and each unit is recognized separately.

C. Holistic Approach

Segmentation-free or holistic Arabic OCR systems perform the recognition on the entire word as a unit without segmenting the word or recognizing characters separately [2]. Several studies have investigated the holistic approach for printed Arabic scrip OCR such as in [79], [16], [8]. OCR systems based on a holistic approach require tracing the feature of the entire word and dealing with words instead of characters. As a result, this approach is restricted to recognizing a word against a lexicon [2]. Moreover, this approach has the challenge of how to deal with the large lexicon size of Arabic words. It is claimed that systems based on this type of segmentation are not useful for general text recognition. A study suggests this approach for systems in which a lexicon is statically defined, such as bank cheque recognition where vocabulary is limited [80].

D. Analytical Approach

For the analytical or segmentation based approach, Arabic OCR systems segment words into smaller units like characters (see Figure 8). In the typical Arabic OCR system, the analytical approach is divided into two approaches: explicit segmentation and implicit segmentation.

1) Explicit Segmentation

The explicit segmentation approach, which is also called dissection segmentation, attempts to segment a word into smaller units. These units could be characters, strokes or loops. Researchers argue that there are two classes of explicit segmentation, which are: direct segmentation and indirect segmentation [81]. In the former, a word is directly segmented into characters exploiting a set of heuristics, while in the latter, a word is divided into smaller segments which can be characters or marks that over segmented characters, such as strokes.

Projection analysis is considered as one of the earliest applied dissection methods on Arabic character segmentation [46], [68], [70]. The projection method of the text image aims to reduce 2D information into 1D in order to simplify the character segmentation process. A method based on a modulated histogram of the image has been proposed in [82]. However, this method has been tested on specific Arabic fonts which do not contain overlapping and ligatures. Consequently, this method would not be appropriate for Arabic fonts that have ligatures, such as traditional Arabic font [3], [62].

Another histogram projection method is presented for printed Farsi word segmentation in [83] which is also applicable to Arabic script, as Arabic script is similar to Farsi script [3]. However, this method is font dependent and ineffective in segmenting small font sizes. Although many of the other techniques based on projection analysis have been devolved for Arabic script such as in [84], [62], [85], it seems that no projection based segmentation algorithm is accurate in segmenting Arabic text [50].

Instead of applying projection analysis methods, contour–based algorithms, which are used for dissection segmentation that rely on the skeleton or contour of Arabic words, are used to simplify the Arabic word segmentation such as in [78]. Other methods rely on white space and pitch finding techniques for segmenting Arabic words [46], [74]. However, a major criticism of the explicit approach is that it is expensive because of the requirement of finding the optimum word from the arrangement of segmented units [81]. The researchers in [71] conclude that an accurate segmentation may not be acquired by relying on dissection segmentation approaches.

2) Implicit Segmentation

In OCR systems based on implicit segmentation, the segmentation phase and recognition phase are performed simultaneously [3]. In other words, a word is segmented into characters while being recognized without segmentation in advance [46]. Straight segmentation and recognition based segmentation are also referred to as implicit segmentation [46]. This segmentation approach searches the text image for components that match predefined classes. The principle of implicit segmentation is to utilize a sliding widow to segment the word image into frames of fixed width on which classification relies to make a decision [86]. Owing to challenges in segmentation of cursive scripts such as Arabic, researchers use the implicit segmentation approach in order to overcome the problems of word segmentation [80]. In principle, by applying this type of segmentation, there is no need for a specific dissection algorithm for Arabic script segmentation and the accuracy performance relates to the classification performance [87]. Thus, some researchers implement techniques based on implicit segmentation in order to improve recognition accuracy of Arabic OCR, such as in [88], [12], [89].

E. Feature Extraction Phase

Once the text image is segmented into isolated regions (such as character, part of character), the next step is feature extraction which is the process of obtaining distinguishing attributes of the segmented character to be utilized by the next phase which is classification [90]. Feature extraction is the most significant level that heavily influences overall OCR performance [11], [25], [60]. The feature extraction stage is correlated with other OCR stages, such as preprocessing and classification stage. In other words, the authors in [91] point out that the selection of feature extraction methods depends on the output of the preprocessing stage. For instance, some techniques for feature extraction work on skeletons, whereas others work on grayscale images. Moreover, the set of features extracted must match the specification of the selected classifier [2].

In terms of OCR performance, feature extraction plays a critical role in achieving high accuracy performance [11], since the feature extraction stage has the contributes to the success of the classification step [60]. However, selection of feature types is a major issue in OCR development [92]. Researchers recommend that the feature extraction methods should be independent of scalable font characteristics such as font styles, font types, font sizes and should be able to describe and distinguish different patterns effectively [92], [93]. In other words, a study emphasizes that the key purpose of selecting good features is to maximize the effectiveness and the efficiency of the OCR system minimizing the complexity and processing time simultaneously [94].

Among OCR system development, researchers propose various types of features. Such features can be categorized into three groups: structural features; statistical features; and global transformation feature [5]. In the following, these features will be discussed in the context of recognizing Arabic script.

1) Structural Features

Structural features illustrate a text image in terms of its topological and geometrical characteristics by using its local and global properties [2], [92]. In case of Arabic script, lines, dots, loops, holes, strokes and zigzags are some structural features [92]–[99]. Considering Arabic script characteristics, some characters have common primary shapes and they can only be differentiated by the number and location of their dots. Thus, the researchers in [59] claim that structural features have been commonly used for Arabic script in order to capture the dot information of characters explicitly.

On the other hand, A study argues that structural feature methods are not capable of discriminating between characters having similar shapes [100]. Similarly, a study reports that relying on the structural features of Arabic script may result in misrecognition, owing to the small difference between Arabic letters [92]. It is mentioned in [5] that extracting structural features of Arabic characters is a challenging task. Furthermore, it is claimed that Arabic OCR systems implementing structural feature methods are processed exhaustively [101]. Likewise, various studies have reported that another complication of applying structural features is that it involves expensive preprocessing techniques, such as skeletonization which may result in character shape distortion and loss of structural feature data [60], [59], [102]. Therefore, research on Arabic OCR has been carried out on other feature extraction approaches, as will be discussed below, that are effective in reducing process time and improving performance accuracy [101].

2) Statistical Features

Statistical features are derived from statistical representation of patterns which provide a measurable event of interested patterns. Researchers in Arabic OCR systems adopt different approaches to produce statistical features. Some examples of the approaches, which have been applied for representing Arabic characters, are zoning, moments, characteristic loci, histograms and crossing [92], [5], [2], [25].

The zoning method divides the character image into serval overlapping and non-overlapping regions. Then, the density of each region pixel is analysesd and used as a feature [92], [103].

The moment method is a common statistical feature approach that has been applied in patter recognition applications [26]. Moments, including Legendre moments, Zernike moments, central moments, pseudo-Zernike moments and Hu moments, extract geometric features in an image, such as, the shape area of a pattern and the center of the mass [17], [104] and [105]. Several studies in printed Arabic script, such as, [106], [107]–[109], have applied moment invariants as a feature vector.

In short, it is claimed that statistical features for pattern representation are easy to extract [92], [2]. Moreover, such features can be effective in recognition systems and providing high speed and low complexity implementation [60], [110]. However, special attention to the prepossessing techniques should be given, since misrecognition may accrue due to poor prepossessing techniques [5]. Nevertheless, the fundamental issue is to determine a set of statistical features, which need to be the most representative data of a pattern, maximizing the performance accuracy and minimizing the processing time simultaneously.

As a result, researchers call for investigating other statistical features which maximize the performance accuracy and minimize the processing time [2], [64].

3) Global Transformation Feature

The global transformation method is applied to convert a skeleton or contour of a pattern by a linear transform into a form that reflects the most relevant features of the transformed pattern [64]. Numerous global transformation methods have been used in developing Arabic OCR systems. An example of such methods is the Fourier descriptor which represents the characteristic of a pattern in a frequency domain [111]. The Fourier descriptor has been applied to Arabic script, such as in [8], [27]. Another method is the Hough transform which detects lines in binary images and then define the parameter of the lines [18]. Other, such as in [112], [113], utilized the Hough transform for extracting features from Arabic script. Also, some other global transformation methods that have been applied for Arabic OCR for feature extraction are the direction codes method such as Freeman's chain code in [28], Wavelets in [114] and Walsh transformation in [107].

Overall, it is claimed that global transformation feature techniques have several advantages over structural and statistical approaches. For example, they are applicable for new fonts and easily implemented. Another advantage is that they are robust to noise and variation. However, they might require the implementation of other features in order to obtain high accuracy performance.

In conclusion, the feature extraction stage plays a critical role in Arabic OCR development in which distinguishing attributes are extracted and it is clear that each Arabic OCR developer needs to apply different feature extraction approaches. Still, good features are required, which assist in distinguishing a character from other characters and maximize the accuracy performance simultaneously. Furthermore, these features must be selected specifically for a selected classifier. Some researchers apply different feature extraction methods in combination. However, this may cause extra complications for the implementation [8].

F. Classification Phase

The classification phase has the responsibility for assigning a pattern into a pre-classified class based on the features of the pattern which have been extracted in the previous phase [18]. The pre-classified classes can be words, sub-words, characters or strokes, based on the OCR approach used [6]. There are a number of different classification approaches that have been applied for Arabic OCR, such as Hidden Markov Models (HMM), Support Vector Machines (SVM), K-nearest neighbour.

SVM, which is a binary classifier, has been used in the implementation of printed Arabic OCR systems [106], [95], [115]. (For a comprehensive review of applying SVM to Arabic OCR, refer to [116]). However, classifiers based on SVM are mostly applied to a small set of data due to the high complexity of training and processing time [117], [118]. Another classification technique that has been applied to printed Arabic OCR are Hidden Markov Model (HMM) based techniques. HMMs are statistical models that are considered as being one of the most efficient for recognition applications especially for speech recognition [17]. Therefore, researchers in OCR have implemented HMMs for OCR in order to obtain high performance OCR systems, such as in [72], [119]–[122].

G. Post-processing Phase

Post-processing is the final stage of the development of Arabic OCR. The objective of this step is to enhance the recognition accuracy by detecting and correcting linguistic misspellings in the produced OCR text without human intervention. Research studies on Arabic OCR have implemented post-progressing methods in order to improve the output, such as [123], [124]. It is worth mentioning that three main elements should be considered in correcting OCR output: non-word errors correction; isolated word errors correction; and context–based word correction [50]. Generally, post-processing methods can be categorized into two main approaches: lexicon-based methods; and context-based (statistical) methods [125].

The typical technique for correcting the mistakes of Arabic OCR outputs is the lexicon-based method which requires the utilization of an Arabic dictionary, such as in [126], [127]. This technique corrects errors without considering any contextual information in which the errors appear. Therefore, a problem might occur with using this approach when a word is misrecognized by an OCR system and is also in the lexicon (these are called real-word errors) such as, *Fear* for *Tear*. This occurs in many languages such as Arabic in which a large fraction of three characters sequences are corrected words. Consequently, only non-word errors can be corrected, since this method is comparing the recognized words with the words that are in the dictionary. Also, this approach requires a wide-ranging lexicon that consists of all single words. However, the Arabic language has various dialects and it is also a triglossic language with three forms – modern standard Arabic and classical Arabic [128] and mixtures of the two. Therefore, this approach is less appropriate for Arabic language since building a single lexicon for Arabic language is more complicated.

On the other hand, context-based (statistical) methods take into account the contextual information in which the misrecognized words appear. A few studies have implemented statistical language models for improving the recognition accuracy of Arabic OCR systems, such as in [129], [130]. Using such methods will help overcome the problem of correcting real-word errors. Moreover, they are also useful in correcting word errors that might have several potential corrections, since these techniques can correct word errors based on grammatical concepts and semantic context [131].

Recently, there have been several attempts to provide systems for correcting Arabic OCR output. For instance, the authors in [123] propose a system for Arabic OCR output correction based on Google online suggestions within Microsoft Office Word. On the other hand, the authors in [131] describe a context-based technique for detecting and correcting Arabic OCR errors. Although there are some studies on applying context-based methods for correcting Arabic OCR output, more research is needed on investigating the use of Arabic contextual information for OCR output correction [132].

IV. Performance Evaluation

OCR performance evaluation can be classified into two types: black-box evaluation and white-box evaluation. In the former, an entire OCR engine is treated as an indivisible unit, so the submodules of the OCR system are not known to the evaluator, whereas with the white-box evaluation, each submodule of the OCR system is evaluated if the submodules are accessible [133]. Performance evaluation of OCR systems is essential for monitoring progress of OCR systems development, assessing the effectiveness of OCR algorithms, identifying open areas for further research and providing scientific justification for the performance of OCR systems [134], [135]. Although the performance evaluation of OCR systems is important, there has been very little work focus on empirical evaluation of Arabic OCR systems, such as in [135], [136] and a recent study in [1]. Furthermore, these evaluation studies have conducted a black-box evaluation on Arabic OCR systems as the submodules are not accessible. Thus, only the overall performances of Arabic OCR have been reported.

Performance evaluation in research areas of pattern recognition is facing several obstacles [137]. For Arabic OCR, conducting performance evaluation is challenging as no standard dataset is available [138], [94]. Moreover, most Arabic OCR systems are evaluated in terms of character accuracy which is a general metric, such as performance results reported in [138], [135], [136]. This accuracy metric is insufficient to assess how Arabic OCR systems are overcoming the challenges of Arabic text. However, a study suggests a new set of objective performance metrics for evaluation Arabic OCR with respect to the challenges of Arabic script which are character accuracy based on character position, dot character accuracy, zigzag-shaped character accuracy, loop-shaped character accuracy and diacritics accuracy [139].

V. Discussion and Future Directions

This paper has overviewed the main stages used in printed Arabic OCR. It main aim is to reveal the current status of printed Arabic OCR. Although there are various attempts to solve the problems of Arabic text recognition, there is still a crucial need for more research.

In an attempt to evaluate the status of printed Arabic OCR and support the claim that more research is needed in many areas, we used Google scholar to search for scientific research publications using phrases that are related to Arabic text recognition. The findings are summarized in Table 2. The table shows the search phrases used and the search results returned by Google Scholar. It is apparent from the table that there is a lack of Arabic OCR research as comparatively very little research has focused on Arabic OCR compared to studies in OCR for other languages. For example, there 322,000 results were returned for the more general search query 'OCR', whereas there were only 956 results returned for the more specific search query 'Arabic OCR'.

In order to provide a measure of the coverage of research in a particular area, we can estimate the probability that a particular research paper will be in a more specific topic area compared to the more general topic area. For example, we may be interested in the general topic area "single font OCR" and wish to see how much research has been published in the more specific topic area "single font Arabic OCR" in comparison. We can estimate the probability p that the more general topic will be concerned with the more specific topic as $p = s / g$ where s is the count of the number of papers found for the specific topic compared to the count g of the number of papers found for the more general topic. Then we can define the 'Information Coverage' I associated with the specific topic in relation to the more general topic as $I = -\log p$. If this value is high compared to other specific vs. general comparisons for the same overall topic (e.g. in relation to Arabic OCR vs. OCR in general), then this reflects that research may under-represented in this area.

This analysis has been done using the values from Table 2 and graphed in Figure 9. In the Figure, we see that except for papers on Arabic OCR concerning easy fonts and diacritics, the remaining topics have higher Information Coverage values meaning that there have been less papers published in these areas proportionately compared to papers published in the more general (non-Arabic) areas. We can use Figure 9 to help gauge the present status of printed Arabic OCR research as it highlights some open areas which need more research. This is based on the number of publications for single, omni and multi font OCR concerning various elements that are related to text recognition concerning easy fonts, complicated fonts, diacritics, page layout, multi-language and noisy documents. In particular, for Arabic text images which contain complicated fonts, there are still many gaps in the research. Furthermore, for single, omni and multi font Arabic OCR on multi-language text images, intensive further research is needed.

Figure 10 plots the number of papers per 5-year period for the top 100 Google Scholar searches using the Arabic OCR related phrases. A number of striking results are apparent in Figure 10. For example, publications for Arabic OCR peak since 2005. Also, the numbers of papers for printed Arabic OCR decrease since 2005 (this could be because researchers have focussed on handwritten Arabic OCR). Furthermore, it is apparent that the smallest numbers of papers in the period of Arabic OCR research are papers related to noisy documents.

Note that the earliest research papers for the 'Arabic OCR' search query are from 1985, which is a result of the top-100 ranking returned by Google Scholar. If, however, we restrict the range of years for which we search, we find that the first papers returned by Google Scholar appear in the 1970 to 1980 period. In contrast, the author in [9] states that text recognition research first originated in 1940, and papers related to the 'text recognition' query appear in Google Scholar from the 1960s.

TABLE II. GOOGLE SCHOLAR SEARCH RESULTS FOR ARABIC OCR RELATED PHRASES

Google Scholar search phrase		Number of papers
1	+"Arabic OCR"	956
	+"OCR"	322,000
2	+"OCR" + "Arabic printed text"	247
	+"OCR" + "printed text"	7,190
3	+"Arabic OCR" + "diacritics"	302
	+"OCR" + "diacritics"	1,800
4	+"Arabic OCR" + "page layout"	54
	+"OCR" + "page layout"	3,360
5	+"Arabic OCR" + "multi-language "	20
	+"OCR" + "multi-language"	866
6	+"Arabic OCR" + "Omni font"	60
	+"OCR" + "Omni font"	380
7	+"Arabic OCR" + "single font"	61
	+"OCR" + "single font"	717
8	+"Arabic OCR" + "multi-font "	149
	+"OCR" + "multi-font"	1270
9	+"Arabic OCR" + "noisy document"	12
	+"OCR" + "noisy document"	515
10	+"Arabic OCR" + " Simplified Arabic " + "single font"	20
	+"Arabic OCR" + " Simplified Arabic " + "Omni font"	22
	+"Arabic OCR" + " Simplified Arabic " + "multi font"	48
11	+"Arabic OCR" + " Advertising Bold " + "single font"	5
	+"Arabic OCR" + " Advertising Bold " + "Omni font"	5
	+"Arabic OCR" + " Advertising Bold " + "multi font"	15
12	+"Arabic OCR" + "diacritics" + "single font"	30
	+"Arabic OCR" + "diacritics" + "Omni font"	36
	+"Arabic OCR" + "diacritics" + "multi font"	62
13	+"Arabic OCR" + " page layout " + "single font"	8
	+"Arabic OCR" + " page layout " + "Omni font"	9
	+"Arabic OCR" + " page layout " + "multi font"	11
14	+"Arabic OCR" + " multi-language " + "single font"	17
	+"Arabic OCR" + " multi-language " + "Omni font"	9
	+"Arabic OCR" + " multi-language " + "multi font"	29
15	+"Arabic OCR" + " noisy document " + "single font"	1
	+"Arabic OCR" + " noisy document " + "Omni font"	2
	+"Arabic OCR" + " noisy document " + "multi font"	3

From the review of each stage used in Arabic OCR, the following observations have been noted for possible research directions:

- All the reviewed research of printed Arabic OCR have used the general OCR methodology which involves the five stages; pre-processing; feature extraction; segmentation; classification and post-processing. However, the following are still open questions: 'Is the current OCR methodology the most effective for designing Arabic OCR?' and 'Are there alternative methodologies that might yield better results for Arabic OCR'?
- From the review of each OCR stage, it is apparent that the most challenging task in the development of Arabic OCR is the segmentation task. Although previous studies have presented different segmentation techniques for Arabic OCR, these studies have not provided the accuracy performance of these techniques. Since only the overall OCR performances have been reported, it is difficult to gain an insight into which segmentation techniques perform better for printed Arabic OCR. A performance evaluation tool should be developed to assess the different segmentation techniques.
- The pre-processing stage review given in this study reached the conclusion that direct adoption of pre-processing methods which are designed for general purposes might be not applicable for Arabic script. Thus, developing pre-processing methods that consider the specific characteristics of Arabic script is needed.
- Most of the proposed methods for feature extraction in Arabic OCR have been adopted from methods that have been developed for other languages without considering the characteristics of Arabic script. Such

methods may not be the most appropriate for accurate recognition. The characteristics of Arabic script should be taken into consideration when selecting a feature extraction method that is able to distinguish between Arabic characters.

- The studies on performance evaluation of printed Arabic OCR have used a black-box evaluation method which can only provide the overall performance of OCR systems. For more insight into which OCR stage is causing the most problems, a white-box evaluation, where each component of the system is accessible, is required.
- Performance evaluation of printed Arabic OCR suffers from the lack of public datasets. For objective performance evaluation, an accurate and free printed Arabic dataset is essential.
- Our investigation indicates the need for more research on Arabic OCR output correction with the use of Arabic contextual information.

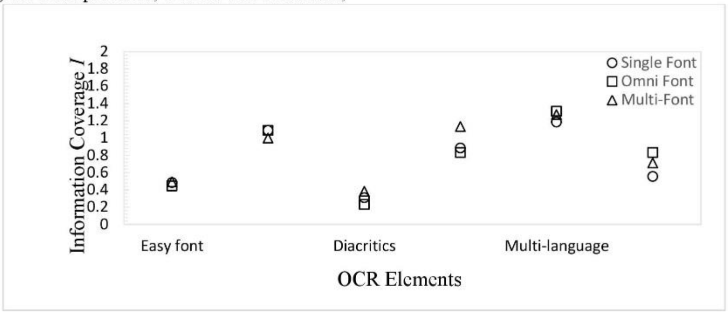


Fig. 9. The present status of printed Arabic OCR based on the number of publications for different OCR elements.

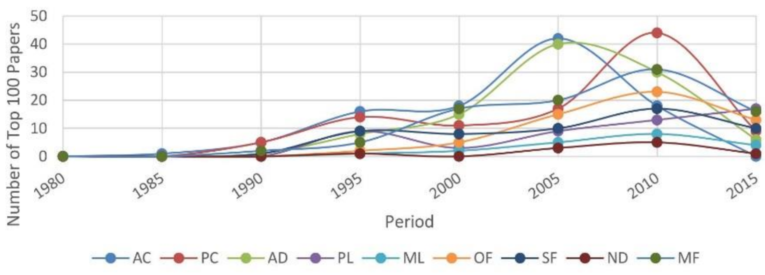


Fig. 10. Number of papers per 5-year period in the top 100 results returned by Google Scholar for different Arabic OCR related search phrases. AC = 'Arabic OCR'; PC = 'Arabic printed text'; AD = 'Arabic diacritics'; PL = 'Arabic OCR + page layout'; ML= 'Arabic OCR + multi-language'; OF = 'Arabic OCR + omni font'; SF = 'Arabic OCR + single font'; MF = 'Arabic OCR + multi font'; ND = 'Arabic OCR + noisy document'.

VI. Conclusion

This paper has provided a comprehensive literature review of printed Arabic text recognition. At first, the specific characteristics of Arabic script that challenge the recognition process have been discussed. Then, the general methodology of printed Arabic OCR has been presented. This methodology was divided into five stages: preprocessing; segmentation; feature extraction; classification; and post-processing. Techniques applied at each stage of Arabic OCR have been discussed. Also, the issues related to the performance evaluation have been reviewed. Finally, we analyzed the remaining problems in the field of printed Arabic OCR and provide several direction for future research.

References

[1] M. Alghamdi and W. Teahan, "Experimental evaluation of Arabic OCR systems," PSU Res. Rev., vol. 1, no. 3, pp. 229–241, 2017.

[2] B. Al-Badr and S. A. Mahmoud, "Survey and bibliography of Arabic optical text recognition," Signal Processing, vol. 41, no. 1, pp. 49–77, 1995.

[3] Y. M. Alginahi, "A survey on Arabic character segmentation," International Journal on Document Analysis and Recognition, vol. 16, no. 2. pp. 105–126, 2013.

[4] M. Saad and W. Ashour, "OSAC: Open Source Arabic Corpora," 6th Int. Conf. Electr. Comput. Syst. (EECS'10), Nov 25-26, 2010, Lefke, Cyprus., pp. 118–123, 2010.

[5] M. S. Khorsheed, "Off-line Arabic character recognition - A review," Pattern Analysis and Applications, vol. 5, no. 1. pp. 31–45, 2002.

[6] B. M. Al-Helali and S. A. Mahmoud, "Arabic Online Handwriting Recognition (AOHR): A Survey," ACM Comput. Surv., vol. 50, no. 3, p. 33, 2017.

[7] M. S. M. El-Mahallawy, "A Large Scale HMM-Based Omni Font-Written OCR System for Cursive Scripts," Faculty of Engineering, Cairo University Giza, Egypt, 2008.

[8] M. Khorsheed and W. Clocksin, "Multi-font Arabic word recognition using spectral features," Pattern Recognition, 2000. …, no. 2, pp. 543–546, 2000.

[9] B. H. Al-Badr, "A Segmentation-free Approach to Text Recognition with Application to Arabic Text," University of Washington, Seattle, WA, USA, 1995.

[10] R. J. Kannan and S. Subramanian, "An adaptive approach of Tamil character recognition using deep learning with big data-A survey," Adv. Intell. Syst. Comput., vol. 337, pp. 557–567, 2015.

[11] A. Lawgali, "A Survey on Arabic Character Recognition," vol. 8, no. 2, pp. 401–426, 2015.

[12] B. Al-Badr and R. M. Haralick, "Segmentation-free word recognition with application to Arabic," Proc. 3rd Int. Conf. Doc. Anal. Recognit., vol. 1, pp. 355–359, 1995.

[13] K. Jumari and M. Ali, "A survey and comparative evaluation of selected off-line arabic handwritten character recognition systems," J. Teknol., vol. 36, pp. 1–17, 2012.

[14] M. Sarfraz, S. N. Nawaz, and A. Al-Khuraidly, "Offline Arabic text recognition system," 2003 International Conference on Geometric Modeling and Graphics 2003 Proceedings. pp. 30–35, 2003.

[15] T. Pavlidis, "Recognition of printed text under realistic conditions," Pattern Recognit. Lett., vol. 14, no. 4, pp. 317–326, 1993.

[16] B. Al-Badr and R. M. Haralick, "A segmentation-free approach to text recognition with application to Arabic text," Int. J. Doc. Anal. Recognit., vol. 1, no. 3, pp. 147–166, 1998.

[17] A. M. AbdelRaouf, "Offline printed Arabic character recognition," 2012.

[18] M. Cheriet, N. Kharma, C. Liu, and C. Suen, Character recognition systems: a guide for students and practitioners. 2007.

[19] W. Cho, S. W. Lee, and J. H. Kim, "Modeling and recognition of cursive words with hidden Markov models," Pattern Recognit., vol. 28, no. 12, pp. 1941–1953, 1995.

[20] F. Drira and F. Lebourgeois, "Denoising textual images using local/non-local smoothing filters : A comparative study," Proc. - Int. Work. Front. Handwrit. Recognition, IWFHR, pp. 521–526, 2012.

[21] Z. Shi, S. Setlur, and V. Govindaraju, "Guide to OCR for Arabic Scripts," V. Märgner and H. El Abed, Eds. London: Springer London, 2012, pp. 79–102.

[22] A. Buades, B. Coll, and J. Morel, "A Review of Image Denoising Algorithms, with a New One," Multiscale Model. Simul., vol. 4, no. 2, pp. 490–530, 2005.

[23] R. Gonzalez and R. Woods, Digital image processing. 2002.

[24] H. Al-ani, N. Ban, and H. M. Abass, "Journal of Computing::Printed Arabic Character Recognition using Neural Network," vol. 5, no. 1, pp. 64–66, 2014.

[25] A. M. Al-Shatnawi, F. H. Al-Zawaideh, S. Al-Salameh, and K. Omar, "Offline Arabic Text Recognition -- An Overview," World Comput. Sci. Inf. Technol., vol. 1, no. 5, pp. 184–192, 2011.

[26] I. Ahmed, S. A. Mahmoud, and M. T. Parvez, "Guide to OCR for Arabic Scripts," V. Märgner and H. El Abed, Eds. London: Springer London, 2012, pp. 147–168.

[27] S. A. Mahmoud, "Arabic character recognition using fourier descriptors and character contour encoding," Pattern Recognit., vol. 27, no. 6, pp. 815–824, 1994.

[28] S. Taha, Y. Babiker, and M. Abbas, "Optical character recognition of arabic printed text," SCOReD 2012 - 2012 IEEE Student Conf. Res. Dev., pp. 235–240, 2012.

[29] A. M. Al-Shatnawi and K. Omar, "Skew Detection and Correction Technique for Arabic Document Images Based on Centre of Gravity Atallah Mahmoud Al-Shatnawi and Khairuddin Omar Department of System Science and Management , Faculty of Information Science and Technology," J. Comput. Sci., vol. 5, no. 5, pp. 363–368, 2009.

[30] I. S. I. Abuhaiba, "Skew Correction of Textural Documents," J. King Saud Univ. - Comput. Inf. Sci., vol. 15, pp. 73–93, 2003.

[31] C. Sun and D. Si, "Skew and slant correction for document images using gradient direction," in Proceedings of the Fourth International Conference on Document Analysis and Recognition, 1997, vol. 1, pp. 142–146.

[32] J. R. Parker, "Algorithms for Image Processing and Computer Vision," Vasa, p. 504, 2011.

[33] H. S. Baird, "The Skew Angle of Printed Documents," in Document Image Analysis, 1995, pp. 204–208.

[34] J. Hull, "Document image skew detection: Survey and annotated bibliography," Ser. Mach. Percept. Artif. …, pp. 40–64, 1998.

[35] J. X. Dong, P. Dominique, A. Krzyzak, and C. Y. Suen, "Cursive word skew/slant corrections based on Radon transform," Proc. Int. Conf. Doc. Anal. Recognition, ICDAR, vol. 2005, pp. 478–482, 2005.

[36] A. Al-Shatnawi and K. Omar, "Methods of Arabic Language Baseline Detection – The State of Art," J. Comput. Sci., vol. 8, no. 10, pp. 137–143, 2008.

[37] Y. Alginahi, "Preprocessing Techniques in Character Recognition," pp. 1–20, 2010.

[38] F. Shafait, Adnan-ul-Hasan, D. Keysers, and T. M. Breuel, "Layout analysis of urdu document images," in 10th IEEE International Multitopic Conference 2006, INMIC, 2006, pp. 293–298.

[39] K. Bin Omar, R. Bin Mahmoud, M. N. Bin Sulaiman, and a. R. Bin Ramli, "The removal of secondaries of Jawi characters," 2000 TENCON Proceedings. Intell. Syst. Technol. New Millenn. (Cat. No.00CH37119), vol. 2, pp. 149–152, 2000.

[40] A. Amin, "Off-line Arabic character recognition," Pattern Recognit., vol. 31, no. 5, pp. 517–530, 1998.

[41] N. Arica and F. T. Yarman-Vural, "Optical character recognition for cursive handwriting," IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 6, pp. 801–813, 2002.

[42] R. El-Hajj, L. Likforman-Sulem, and C. Mokbel, "Arabic handwriting recognition using baseline dependant features and hidden Markov modeling," in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 2005, vol. 2005, pp. 893–897.

[43] M. Pechwitz and V. Margner, "Baseline estimation for Arabic handwritten words," in Proceedings - International Workshop on Frontiers in Handwriting Recognition, IWFHR, 2002, pp. 479–484.

[44] A. Zidouri, M. Sarfraz, S. N. Nawaz, and M. J. Ahmad, "PC based offline Arabic text recognition system," Seventh Int. Symp. Signal Process. Its Appl. 2003. Proceedings., vol. 2, 2003.

[45] H. Al-rashaideh, "Preprocessing phase for Arabic Word Handwritten Recognition," Inf. Transm. Comput. Networks J., vol. 6, no. 1, pp. 11–19, 2006.

[46] A. M. Zeki, "The segmentation problem in arabic character recognition the state of the art," in Proceedings of 1st International Conference on Information and Communication Technology, ICICT 2005, 2005, vol. 2005, pp. 11–26.

[47] S. S. Bukhari, F. Shafait, and T. M. Breuel, "High performance layout analysis of Arabic and Urdu document images," in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 2011, pp. 1275–1279.

[48] G. Nagy, S. Seth, and M. Viswanathan, "A prototype document image analysis system for technical journals," Computer (Long. Beach. Calif)., vol. 25, no. 7, pp. 10–22, 1992.

[49] A. M. Al-shatnawi and K. Omar, "The Thinning Problem in Arabic Text Recognition - A Comprehensive Review," Int. J. Comput. Appl., vol. 103, no. 3, pp. 35–42, 2014.

[50] M. T. Parvez and S. a. Mahmoud, "Offline arabic handwritten text recognition: A Survey," ACM Comput. Surv., vol. 45, no. 2, p. 23:1–23:35, 2013.

[51] Z. Guo and R. W. Hall, "Fast fully parallel thinning algorithms," CVGIP Image Underst., vol. 55, no. 3, pp. 317–328, 1992.

[52] C. Hilditch, "Comparison of thinning algorithms on a parallel processor," Image Vis. Comput., vol. 1, no. 3, pp. 115–132, 1983.

[53] P. S. P. Wang and Y. Y. Zhang, "A Fast and Flexible Thinning Algorithm," IEEE Trans. Comput., vol. 38, no. 5, pp. 741–745, 1989.

[54] S. A. Mahmoud, I. AbuHaiba, and R. J. Green, "Skeletonization of Arabic characters using clustering based skeletonization algorithm (CBSA)," Pattern Recognit., vol. 24, no. 5, pp. 453–464, 1991.

[55] M. Melhi, S. S. Ipson, and W. Booth, "A novel triangulation procedure for thinning hand-written text," Pattern Recognit. Lett., vol. 22, no. 10, pp. 1059–1071, 2001.

[56] J. Cowell and F. Hussain, "Thinning Arabic characters for feature extraction," in Proceedings of the International Conference on Information Visualisation, 2001, vol. 2001–Janua, pp. 181–185.

[57] M. Altuwaijri and M. Bayoumi, "A new thinning algorithm for Arabic characters using self-organizing neural network," in Circuits and Systems, 1995. ISCAS'95., 1995 IEEE International Symposium on, 1995, vol. 3, pp. 1824–1827.

[58] M. A. Alghamdi and W. J. Teahan, "A New Thinning Algorithm for Arabic Script," Int. J. Comput. Sci. Inf. Secur. (IJCSIS), vol. 15, no. 1, pp. 204–211, 2017.

[59] L. M. Lorigo and V. Govindaraju, "Offline arabic handwriting recognition: A survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 5. pp. 712–724, 2006.

[60] S. Naz, K. Hayat, M. Imran Razzak, M. Waqas Anwar, S. A. Madani, and S. U. Khan, "The optical character recognition of Urdu-like cursive scripts," in Pattern Recognition, 2014, vol. 47, no. 3, pp. 1229–1248.

[61] R. Plamondon and S. N. Srihari, "On-line and off-line handwriting recognition: A comprehensive survey," IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, pp. 63–84, 2000.

[62] M. Amara, K. Zidi, K. Ghedira, and S. Zidi, "New Rules to Enhance the Performances of Histogram Projection for Segmenting Small-Sized Arabic Words," in Hybrid Intelligent Systems, Springer, 2016, pp. 167–176.

[63] S. N. Nawaz, M. Sarfraz, A. Zidouri, and W. G. Al-Khatib, "An approach to offline Arabic character recognition using neural networks," in Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003 10th IEEE International Conference on, 2003, vol. 3, p. 1328–1331 Vol.3.

[64] N. Arica and F. T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, vol. 31, no. 2. pp. 216–233, 2001.

[65] J. Alkhateeb, "Word Based Off-line Handwritten Arabic Classification and Recognition," Ph. D. thesis, School of Computing, Informatics and Media, University of Bradford, 2010.

[66] L. O'Gorman, "The Document Spectrum for Page Layout Analysis," IEEE Trans. Pattern Anal. Mach. Intell., vol. 15, no. 11, pp. 1162–1173, 1993.

[67] S. N. Srihari and G. Ball, "An assessment of Arabic handwriting recognition technology," in Guide to OCR for Arabic Scripts, Springer, 2012, pp. 3–34.

[68] A. Amin and G. Masini, "Machine Recognition of Multi Font Printed {Arabic} Texts," in Proceedings, Eighth International Conference on Pattern Recognition (Paris, France, October 27--31, 1986), 1986, pp. 392–395.

[69] J. F. Mari, "Machine Recognition and Correction of Printed Arabic Text," IEEE Trans. Syst. Man Cybern., vol. 19, no. 5, pp. 1300–1306, 1989.

[70] A. Ymin and Y. Aoki, "On the segmentation of multi-font printed Uygur scripts," in Pattern Recognition, 1996., Proceedings of the 13th International Conference on, 1996, vol. 3, pp. 215–219.

[71] A. Cheung, M. Bennamoun, and N. W. Bergmann, "Arabic optical character recognition system using recognition-based segmentation," Pattern Recognit., vol. 34, no. 2, pp. 215–233, 2001.

[72] M. S. Khorsheed, "Offline recognition of omnifont Arabic text using the HMM ToolKit (HTK)," Pattern Recognit. Lett., vol. 28, no. 12, pp. 1563–1571, 2007.

[73] A. Amin, "Off-line Arabic character recognition: the state of the art," Pattern Recognit., vol. 31, no. 5, pp. 517–530, 1998.

[74] H. Al-Yousefi and S. S. Udpa, "Recognition of arabic characters," IEEE Trans. Pattern Anal. Mach. Intell., vol. 14, no. 8, pp. 853–857, 1992.

[75] I. S. I. Abuhaiba, S. A. Mahmoud, and R. J. Green, "Recognition of handwritten cursive Arabic characters," IEEE Trans. Pattern Anal. Mach. Intell., vol. 16, no. 6, pp. 664–672, 1994.

[76] I. Supriana and A. Nasution, "Arabic Character Recognition System Development," Procedia Technol., vol. 11, no. Iceei, pp. 334–341, 2013.

[77] J. H. AlKhateeb, J. Jiang, J. Ren, and S. Ipson, "Component-based Segmentation of words from handwritten Arabic text," Int. J. Comput. Syst. Sci. Eng., vol. 5, no. 1, pp. 54–58, 2009.

[78] P. Xiu, L. Peng, X. Ding, and H. Wang, "Offline handwritten Arabic character segmentation with probabilistic model," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2006, vol. 3872 LNCS, pp. 402–412.

[79] E. J. Erlandson, J. M. Trenkle, and R. C. Vogt, "Word-level recognition of multifont Arabic text using a feature vector matching approach," in Document Recognition III, 1996, vol. 2660, pp. 63–71.

[80] A. Choudhary, "A review of various character segmentation techniques for cursive handwritten words recognition," Int. J. Inf. Comput. Technol, vol. 4, no. 6, pp. 559–564, 2014.

[81] S. Naz, A. I. Umar, S. H. Shirazi, S. B. Ahmed, M. I. Razzak, and I. Siddiqi, "Segmentation techniques for recognition of Arabic-like scripts: A comprehensive survey," Educ. Inf. Technol., pp. 1–17, 2015.

[82] B. A. Najoua and E. Noureddine, "A robust approach for Arabic printed character segmentation," in Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on, 1995, vol. 2, pp. 865–868.

[83] B. Parhami and M. Taraghi, "Automatic recognition of printed Farsi texts," Pattern Recognit., vol. 14, no. 1, pp. 395–403, 1981.

[84] L. Zheng, A. H. Hassin, and X. Tang, "A new algorithm for machine printed Arabic character segmentation," Pattern Recognit. Lett., vol. 25, no. 15, pp. 1723–1729, 2004.

[85] L. Lorigo and V. Govindaraju, "Segmentation and pre-recognition of arabic handwriting," in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 2005, vol. 2005, pp. 605–609.

[86] D. Koteswara Rao and A. Negi, "Advances in Signal Processing and Intelligent Recognition Systems: Proceedings of Second International Symposium on Signal Processing and Intelligent Recognition Systems (SIRS-2015) December 16-19, 2015, Trivandrum, India," M. S. Thampi, S. Bandyopadhyay, S. Krishnan, K.-C. Li, S. Mosin, and M. Ma, Eds. Cham: Springer International Publishing, 2016, pp. 633–644.

[87] A. M. Zeki, M. S. Zakaria, and C.-Y. Liong, "Segmentation of Arabic Characters: A Comprehensive Survey," Int. J. Technol. Diffus., vol. 2, no. 4, pp. 48–82, 2011.

[88] A. Cheung, M. Bennamoun, and N. W. Bergmann, "An Arabic optical character recognition system using recognition-based segmentation," Pattern Recognit., vol. 34, no. 2, pp. 215–233, 2001.

[89] C. H. Chen and J. L. DeCurtins, "Word recognition in a segmentation-free approach to OCR," Proc. 2nd Int. Conf. Doc. Anal. Recognit. ICDAR 93, pp. 573–576, 1993.

[90] B. A. Srinivas, A. Agarwal, and C. R. Rao, "An Overview of OCR Research in Indian Scripts," vol. 2, no. 2, 2008.

[91] Ø. Due Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition-A survey," Pattern Recognit., vol. 29, no. 4, pp. 641–662, 1996.

[92] E. Moubtahij, A. H. Hicham, and K. SATORI, "Review of Feature Extraction Techniques for Offline Handwriting Arabic Text Recognition," Int. J. Adv. Eng. Technol., vol. 7, no. 1, pp. 50–58, 2014.

[93] D. V. Sharma, G. Saini, and M. Joshi, "Statistical Feature Extraction Methods for Isolated Handwritten Gurumukhi Script," vol. 2, no. 4, pp. 380–384, 2012.

[94] H. Al-Muhtaseb and R. Qahwaji, "Arabic Optical Character Recognition: Recent Trends and Future Directions," in Applied Signal and Image Processing: Multidisciplinary Advancements, IGI Global, 2011, pp. 324–346.

[95] M. Abd, S. Al Rubeaai, and G. Paschos, "Hybrid Features for an Arabic Word Recognition System," Davidpublishing.Com, vol. 3, no. 1, pp. 685–691, 2012.

[96] R. Saabni, "Efficient recognition of machine printed Arabic text using partial segmentation and Hausdorff distance," 6th Int. Conf. Soft Comput. Pattern Recognition, SoCPaR 2014, pp. 284–289, 2015.

[97] a. M. Elgammal and M. a. Ismail, "A graph-based segmentation and feature extraction framework for\nArabic text recognition," Proc. Sixth Int. Conf. Doc. Anal. Recognit., 2001.

[98] M. S. Khorsheed and W. F. Clocksin, "Structural Features of Cursive Arabic Script.," in BMVC, 1999, pp. 1–10.

[99] I. Ahmad, L. Rothacker, G. A. Fink, and S. A. Mahmoud, "Novel sub-character HMM models for arabic text recognition," Proc. Int. Conf. Doc. Anal. Recognition, ICDAR, pp. 658–662, 2013.

[100] D. Ghosh, T. Dube, and A. Shivaprasad, "Script Recognition-a review," IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 12, pp. 2142–2161, 2010.

[101] H. Almohri, J. S. Gray, and H. Alnajjar, A real-time DSP-based optical character recognition system for isolated arabic characters using the TI TMS320C6416T. University of Hartford, 2007.

[102] M. Z. Hossain, "Rapid Feature Extraction for Optical Character Recognition," pp. 1–5, 2012.

[103] G. Kumar and P. K. Bhatia, "A detailed review of feature extraction in image processing systems," in International Conference on Advanced Computing and Communication Technologies, ACCT, 2014, pp. 5–12.

[104] H. Aboaisha, Z. Xu, and I. El-Feghi, "An investigation on efficient feature extraction approaches for Arabic letter recognition," 2012.

[105] M. Kef, L. Chergui, and S. Chikhi, "Comparative study of the use of geometrical moments for Arabic handwriting recognition," in Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), 2012 6th International Conference on, 2012, pp. 303–308.

[106] M. A. Abd and G. Paschos, "Effective arabic character recognition using support vector machines," Innov. Adv. Tech. Comput. Inf. Sci. Eng., pp. 7–11, 2007.

[107] M. Oujaoura, R. El Ayachi, M. Fakir, B. Bouikhalene, and B. Minaoui, "Zernike moments and neural networks for recognition of isolated Arabic characters," Int. J. Comput. Eng. Sci., vol. 2, pp. 17–25, 2012.

[108] Z. Shaaban, "A new recognition scheme for machine-printed Arabic texts based on neural networks," in Proceedings of World Academy of Science, Engineering and Technology, 2008, vol. 31, pp. 25–27.

[109] I. A. Elrube, M. T. El Sonni, and S. S. Saleh, "Printed Arabic sub-word recognition using moments," World Acad. Sci. Eng. Technol., vol. 66, pp. 737–741, 2010.

[110] P. B. Pati and A. G. Ramakrishnan, "OCR in Indian scripts: A survey," IETE Tech. Rev., vol. 22, no. 3, pp. 217–227, 2005.

[111] D. Zhang and G. Lu, "A Comparative Study on Shape Retrieval Using Fourier Descriptors with Different Shape Signatures," Int. Conf. Intell. Multimed. Distance Educ., vol. 1, pp. 1–9, 2001.

[112] N. Ben Amor and N. E. Ben Amara, "Multifont Arabic Characters Recognition Using HoughTransform and HMM/ANN Classification," J. Multimed., vol. 1, no. 2, pp. 50–54, 2006.

[113] S. Touj, N. E. Ben Amara, and H. Amiri, "Generalized Hough Transform for Arabic Printed Optical Character Recognition.," Int. Arab J. Inf. Technol., vol. 2, no. 4, pp. 326–333, 2005.

[114] N. Ben Amor and N. E. Ben Amara, A Novel Method for Multifont Arabic Characters Features Extraction. INTECH Open Access Publisher, 2012.

[115] R. Mehran, H. Pirsiavash, and F. Razzazi, "A front-end OCR for omni-font Persian/Arabic cursive printed documents," in Digital Image Computing: Techniques and Applications, 2005. DICTA'05. Proceedings 2005, 2005, p. 56.

[116] M. Amara, K. Zidi, S. Zidi, and K. Ghedira, "Arabic Character Recognition Based M-SVM," in International Conference on Advanced Machine Learning Technologies and Applications, 2014, pp. 18–25.

[117] M. Shafii, "Optical Character Recognition of Printed Persian/Arabic Documents," 2014.

[118] S. Arora, D. Bhattacharjee, M. Nasipuri, L. Malik, M. Kundu, and D. K. Basu, "Performance Comparison of SVM and ANN for Handwritten Devnagari Character Recognition," Int. J. Comput. Sci. Issues, vol. 7, no. 3, pp. 1–10, 2010.

[119] M. S. Khorsheed, "Recognizing Cursive Typewritten Text Using Segmentation-Free System," vol. 2015, 2015.

[120] S. M. Awaida and M. S. Khorsheed, "Developing discrete density Hidden Markov Models for Arabic printed text recognition," Proceeding - 2012 IEEE Int. Conf. Comput. Intell. Cybern. Cybern. 2012, pp. 35–39, 2012.

[121] F. Slimane, S. Kanoun, J. Hennebert, A. M. Alimi, and R. Ingold, "A study on font-family and font-size recognition applied to Arabic word images at ultra-low resolution," Pattern Recognit. Lett., vol. 34, no. 2, pp. 209–218, 2013.

[122] H. A. Al-Muhtaseb, S. A. Mahmoud, and R. S. Qahwaji, "Recognition of off-line printed Arabic text using Hidden Markov Models," Signal Processing, vol. 88, no. 12, pp. 2902–2912, 2008.

[123] I. A. Doush and A. M. Al Trad, "Improving post-processing optical character recognition documents with Arabic language using spelling error detection and correction," Int. J. Reason. Intell. Syst., vol. 8, no. 3/4, p. 91, 2016.

[124] W. Magdy and K. Darwish, "Effect of OCR error correction on Arabic retrieval," Inf. Retr. Boston., vol. 11, no. 5, pp. 405–425, 2008.

[125] K. Taghva and E. Stofsky, "OCRSpell: An interactive spelling correction system for OCR errors in text," Int. J. Doc. Anal. Recognit., vol. 3, no. 3, pp. 125–137, 2001.

[126] A. H. Hassin, X.-L. Tang, J.-F. Liu, and W. Zhao, "Printed Arabic character recognition using HMM," J. Comput. Sci. Technol., vol. 19, no. 4, pp. 538–543, 2004.

[127] I. Aljarrah, O. Al-Khaleel, K. Mhaidat, M. Alrefai, A. Alzu'Bi, and M. Rabab'Ah, "Automated system for arabic optical character recognition with lookup dictionary," J. Emerg. Technol. Web Intell., vol. 4, no. 4, pp. 362–370, 2012.

[128] P. Damien, N. Wakim, and M. Eg??a, "Phoneme-viseme mapping for modern, classical arabic language," in 2009 International Conference on Advances in Computational Tools for Engineering Applications, ACTEA 2009, 2009, pp. 547–552.

[129] R. Prasad, S. Saleem, M. Kamali, R. Meermeier, and P. Natarajan, "Improvements in hidden Markov model based Arabic OCR," Proc. 19th Int. Conf. Pattern Recognit., pp. 1–4, 2008.

[130] P. Natarajan, S. Saleem, R. Prasad, E. MacRostie, and K. Subramanian, "Multi-lingual Offline Handwriting Recognition Using Hidden Markov Models: A Script-Independent Approach," in Arabic and Chinese Handwriting Recognition, 2008, pp. 231–250.

[131] Y. Bassil and M. Alwani, "OCR Post-Processing Error Correction Algorithm Using Google ' s Online Spelling Suggestion," J. Emerg. Trends Comput. Inf. Sci., vol. 3, no. 1, pp. 90–99, 2012.

[132] A. G. Krayem, "A high level approach to Arabic sentence recognition," Nottingham Trent University, 2013.

[133] S. V Rice, "Measuring the accuracy of page-reading systems," University of Nevada, Las Vegas, 1996.

[134] S. Mihov, K. U. Schulz, C. Ringlstetter, V. Dojchinova, V. Nakova, K. Kalpakchieva, O. Gerasimov, A. Gotscharek, and C. Gercke, "A corpus for comparative evaluation of OCR software and postcorrection techniques," Proc. Int. Conf. Doc. Anal. Recognition, ICDAR, vol. 2005, pp. 162–166, 2005.

[135] T. Kanungo, G. A. Marton, and O. Bulbul, "OmniPage vs. Sakhr: Paired model evaluation of two Arabic OCR products," in Electronic Imaging'99, 1999, pp. 109–120.

[136] T. Kanungo, G. a Marton, and O. Bulbul, "Performance evaluation of two Arabic OCR products," Proc. SPIE, pp. 76–83, 1999.

[137] K. Jung, K. I. Kim, and A. K. Jain, "Text information extraction in images and video: A survey," Pattern Recognit., vol. 37, no. 5, pp. 977–997, 2004.

[138] I. Ahmad, S. A. Mahmoud, and G. A. Fink, "Open-vocabulary recognition of machine-printed Arabic text using hidden Markov models," Pattern Recognit., vol. 51, pp. 97–111, 2016.

[139] M. A. Alghamdi, I. S. Alkhazi, and W. J. Teahan, "Arabic OCR evaluation tool," in 2016 7th International Conference on Computer Science and Information Technology (CSIT), 2016, pp. 1–6.

2016 7th International Conference on Computer Science and Information Technology (CSIT) ISBN: 978-1-4673-8913-6

Arabic OCR Evaluation Tool

Mansoor A Alghamdi
Department of Computer Science
Community Collage
Tabuk University
Tabuk, Saudi Arabia
malghamdi@ut.edu.sa

Ibrahim S Alkhazi
College of Computers & Information Technology
Tabuk University
Tabuk, Saudi Arabia
i.alkhazi@ut.edu.sa

William J. Teahan
School of Computer Science
Bangor University
United Kingdom
w.j.teahan@bangor.ac.uk

***Abstract*—Performance evaluation of Optical Character Recognition (OCR) systems is an essential task for OCR systems development. However, studies in Arabic OCR suffer from the lack of proper performance evaluation metrics and the availability of evaluation tools. Although the literature provides typical performance metrics, such as character accuracy and word accuracy for OCR performance evaluation, these metrics are not sufficient for evaluating Arabic OCR. This paper presents an open source automated software tool with various metrics for the evaluation of Arabic OCR performance. The developed tool is available for OCR researchers, thus it can be applied for ranking different OCR algorithms.**

Keywords—performance evaluation; Arabic; OCR; Arabic OCR; evaluation metrics.

I. Introduction

Optical Character Recognition (OCR) is a process of converting a machine-printed or handwritten text image into a digital computer format that can be editable. OCR technology is considered as a challenging research area in the field of pattern recognition and artificial intelligence. Many studies have been proposed for Latin and non-Latin scripts. However, development of OCR applications for Latin script is easier than that of Arabic scripts because of Arabic writing characteristics [1]. Fig. 1 illustrates some characteristics of the printed Arabic script that contribute to the challenges in Arabic OCR evolution. Compared to printed Latin script, Arabic script is written cursively from right to left, and contains dots and diacritics. Also, a character of Arabic script may have four dissimilar shapes in relation to its location in an Arabic word.

Furthermore, when comparing the performance of Arabic OCR with the Latin script OCR, researchers call attention to the need for more research solutions for Arabic OCR [2], [3]. Consequently, evaluating the performance of OCR is required [4] in order to develop better OCR algorithms. In other words, having quantitative measures will result in monitoring progress in the OCR development field. Moreover, comparing the performance of OCR systems will contribute to providing a scientific explanation for the system performance, identifying open areas, and analysing the weakness and strengths of the OCR systems, as stated by [5] and [6].

In respect to performance evaluation of Arabic OCR systems, several researchers [7], [8], [9] report the accuracy performance of their systems in the absence of specifying the method of measuring the performance. Hence, researchers in Arabic OCR may not be able to assure the comparability of the performance of Arabic OCR systems. Besides, some studies, such as [10], [11], evaluate Arabic OCR systems only in terms of character accuracy, which is the standard performance metric. However, despite this it is claimed that the accuracy rates of Arabic OCR systems are comparable. Thus, different accuracy metrics to classify the Arabic OCR are required [12].

On the other hand, to obtain statistically meaningful results when evaluating the performance of OCR systems, a great number of text images are needed to be tested [4]. Although tools exist for other languages (e.g. English [13], [14]), none exists for Arabic. Therefore, a programmed test tool is needed to automate the evaluation test. Additionally, conducting the performance evaluation under an automated tool will result in eliminating human error, improve speed, precision and reduce repetition [12].

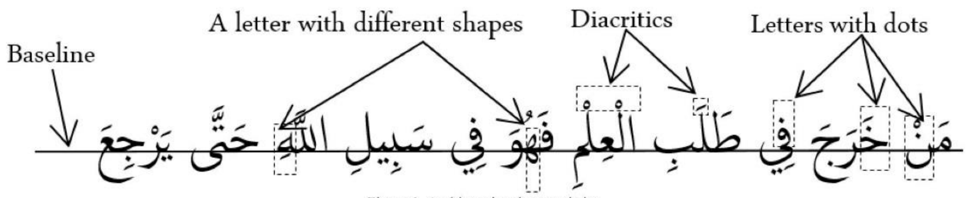

A letter with different shapes
Diacritics
Letters with dots
Baseline
مَنْ خَرَجَ فِي طَلَبِ الْعِلْمِ فَهُوَ فِي سَبِيلِ اللَّهِ حَتَّى يَرْجِعَ

Figure 1. Arabic script characteristics

978-1-4673-8914-3/16/$31.00 ©2016 IEEE

There are several experimental studies on Arabic OCR evaluation [15], [5] and [16]. Still, only two studies—[12], [16]—indicate that an automated tool was utilised to evaluate the performance of Arabic OCR systems. Unfortunately, the utilised tool is not available for researchers and is limited in accuracy metrics.

This paper first presents an overview of performance evaluation for OCR systems. Then, a set of objective accuracy metrics for Arabic OCR evaluation is discussed. Also, the developed tool for the evaluation of Arabic OCR performance is described.

II. AN OVERVIEW OF OCR EVALUATION

Evaluation of OCR systems can be classified into two types: black-box evaluation and white-box evaluation [15]. In the former, an entire OCR engine is treated as an indivisible unit, so the submodules of the OCR system are not known to the evaluator [15]. Further, the evaluation of this type is concerned with the output rather than how the output is produced, whereas with the latter evaluation, the submodules of the OCR system must be accessed by the evaluator in order to evaluate each submodule [5]. Thus, it is possible to evaluate different proposed OCR systems by relying on the white-box evaluation method, since it does not require having access to the submodules of OCR systems.

Evaluating the performance of OCR systems involves comparing an observed variable, which is the output text of OCR, with a reference variable, which is the original text, called ground truth, under controlled conditions [17], [18].

III. OCR ACCURACY EVALUATION

A general measure of OCR systems, when the ground truth text is available, is to determine the differences between the OCR output text and the ground truth text [18]. Since the total cost of correcting errors in the output of OCR system is an important factor of an OCR performance system, the most meaningful measurement of an OCR system is accuracy [17]. A general definition of the term "accuracy" of OCR systems is the number of items (characters or words) that have been recognised correctly on a text image and normalised by the total number of items in the ground truth text [5].

In order to assess accuracy, the number of correct, inserted, deleted and substituted symbols should be taken into consideration [19]. The accuracy of OCR engines can be determined by the *edit distance* between the OCR output text and the ground truth text. The authors in [18] clarify that the edit distance between the output text of OCR and the ground truth text can be defined as the minimum number of operations required to convert the output text into the ground truth text. The edit operations are: insertion, deletion and substitution.

For example, the edit distance between the two strings, the ground truth string (جامعة بانجور) and the OCR-generated string, (حامعة بيانجور) is 2. The required operations to transform the OCR-generated string into the correct one are: (1) substitute the underlined letter in the OCR output (حـ); for (جـ) and (2) delete the underlined letter (بـ). This number, which is the minimum number of single character edit operations, is known as the Levenshtein distance [20]. Regarding OCR evaluation studies, it is common to determine the edit distance in different levels: word level and character level. However, character level accuracy is useful for predicting improvements in OCR systems in which OCR developers are interested, whereas word level accuracy is useful for analysing the ease of human readability, as [5] emphasise. Word accuracy is outside the scope of this of this paper and is left for future work.

Owing to the inadequate quality of OCR systems, accuracy metrics that provide interesting measures of the OCR performance system have been demanded by many studies [17], [21], [13] and [22]. In particular, for Arabic OCR systems, accuracy rates are comparable, and thus some researchers call attention to the need for different performance accuracy metrics to classify Arabic OCR systems [16]. The next section will provide unique Arabic accuracy metrics that will be implemented in the tool.

IV. ARABIC ACCURACY METRICS

The literature suggests several possible performance measures for evaluating OCR, such as character accuracy, accuracy by character class and marked character accuracy [23], [22] and [12]. However, because of Arabic script characteristics, it is not sufficient to assess Arabic OCR performance by relying on such metrics. Thus, the authors propose different accuracy metrics for evaluation of Arabic OCR engines as discussed below.

A. Character accuracy

Accuracy in an OCR-generated text in respect to the ground truth text is computed by Levenshtein edit distance; that is, the minimum number of primitive operations that are required to correct the OCR output text to be matched with the ground truth text. These operations are substitution, deletion and insertion. Then, character accuracy is determined by:

$$\frac{m-e}{m}\times 100 \qquad (1)$$

where e is the edit distance, and m is the number of characters in the reference text.

B. Character accuracy based on character class

Considering Arabic script, some characters have features that allow them to be classified into a particular class. Consequently, it will be valuable to analyse the accuracy of each class. To determine the accuracy of this metric, Arabic characters have been classified into various classes as below:

1) Character position (form shape) class

As illustrated earlier, an Arabic letter may have various shapes depending on its position in a word, whether it is an isolated letter, an initial letter, a middle letter or a terminal letter. In order to analyse the accuracy of this class, the ground truth Arabic characters are categorised into four classes: isolated, initial, middle, and end, as shown in Table I. Then, the tool will compute the accuracy of each class by using equation (1).

TABLE I CHARACTER POSITION CLASS

Isolated	Initial	Middle	End
ا	ا	ـا	ـا
ب	بـ	ـبـ	ـب
ت	تـ	ـتـ	ـت
ث	ثـ	ـثـ	ـث
ج	جـ	ـجـ	ـج
ح	حـ	ـحـ	ـح
خ	خـ	ـخـ	ـخ
د	د	ـد	ـد
ذ	ذ	ـذ	ـذ
ر	ر	ـر	ـر
ز	ز	ـز	ـز
س	سـ	ـسـ	ـس
ش	شـ	ـشـ	ـش
ص	صـ	ـصـ	ـص
ض	ضـ	ـضـ	ـض
ط	ط	ـطـ	ـط
ظ	ظ	ـظـ	ـظ
ع	عـ	ـعـ	ـع
غ	غـ	ـغـ	ـغ
ف	فـ	ـفـ	ـف
ق	قـ	ـقـ	ـق
ك	كـ	ـكـ	ـك
ل	لـ	ـلـ	ـل
م	مـ	ـمـ	ـم
ن	نـ	ـنـ	ـن
ه	هـ	ـهـ	ـه
و	و	ـو	ـو
ي	يـ	ـيـ	ـي

2) Dot character class

Arabic OCR systems have faced challenges in recognising characters that contain dots, since some characters have the same shape and they can be distinguished only by the number of dots, as illustrated in Fig. 1. Thus, in order to evaluate the accuracy of this class, Arabic characters can be categorised into four classes: one dot, two dots, three dots and non-dot characters, as displayed in Table II. Each class accuracy is given by equation (1).

TABLE II DOT CHARACTER CLASS

One dot	Two dots	Three dots	Non-dot
ب ج خ ذ ز ض ظ غ ف ن	ت ق ي ة	ث ش	ا ح د ر س ص ط ع ل م ه و ك ى

3) Zigzag character class

Arabic script can be written with Hamza, which has the zigzag shape. It would be interesting to measure an Arabic OCR in terms of characters that contain the zigzag shape. Consequently, this tool analyses the accuracy of the zigzag character class that is shown in Fig. 2. The zigzag character accuracy is also computed by equation (1).

4) Dot Character based on baseline class

Dots of Arabic characters can be placed either above or below the baseline. Owing to the significance of the baseline in Arabic OCR, it would be valuable to compute the accuracy of characters that are most related to the baseline. Regarding the baseline, Arabic dot characters will be divided into two groups: above baseline and below baseline, as shown in Table III. Each group accuracy is expressed by equation (1).

TABLE III BASELINE CHARACTER CLASS

Above baseline	Below baseline
ق ن ف غ ظ ض ش ز ذ خ ث ت	ي ج ب

5) Loop Character class:

When considering the characteristics of Arabic script, several letters consist of a loop shape. Such a feature may be an obstacle in Arabic OCR development. Thus, it would be desirable to assess the accuracy of characters that have this feature. In order to assess the accuracy of this class, loop characters are defined according to how they are presented in Fig. 3. The accuracy of loop characters are also determined by equation (1).

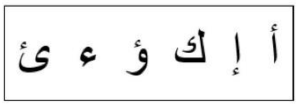

أ إ ك ؤ ء ئ

Figure 2. Zigzag characters

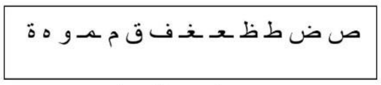

ص ض ط ظ ـعـ ـغـ ـف ق م ـمـ و ه ة

Figure 3. Loop characters

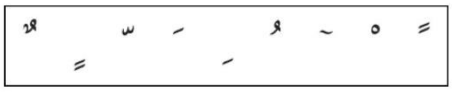


Figure 4. Diacritic marks

C. Diacritic group accuracy

A major feature of Arabic script is the appearance of diacritical marks, which influences the accuracy of Arabic OCR systems. Fig. 4 shows Arabic diacritics, which are analysed by the evaluation tool to provide the accuracy of Arabic OCR in terms of diacritics by equation (1).

D. Digit group accuracy

It is critical for developers to assess the recognition of numbers by Arabic OCR applications. In respect to Arabic script, both Arabic and English numerals may appear in any Arabic text. Thus, the evaluation considers all numerals in computing the accuracy of digits. Digit accuracy is provided by equation (1).

E. Punctuation group accuracy

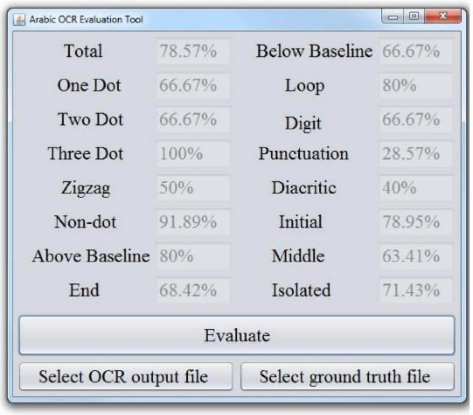
Several punctuation symbols resemble Arabic characters. For example, the full-stop punctuation (.) is quite identical to the Arabic Zero number (٠). This illustration confirms that the punctuation group has a vital effect on Arabic OCR accuracy. As a result, determining the accuracy of this group will be measured with the presented tool by equation (1).

V. ARABIC OCR EVALUATION TOOL

The Arabic OCR evaluation tool, which is programmed in Java, allows for comparison of an OCR-generated text file with a ground truth text file. The difference between the two text files is computed in terms of the minimum cost of converting the OCR output text to the ground truth. For this purpose, Levenshtein edit distance algorithm in [20] has been adopted.

The Levenshtein method calculates the edit distance between two strings where edit distance is the minimum number of insertions, substitutions or deletions that are necessary to convert one string into the other. For a comprehensive explanation of the Levenshtein edit distance algorithm refer to [20].

By using the software tool, an evaluator can quantitatively measure the performance of Arabic OCR systems according to various Arabic accuracy metrics. A Graphical User Interface (GUI) has been implemented to facilitate evaluators to select the OCR output text file and the ground truth text file, as demonstrated in Fig. 5. Also, as can be seen in Fig. 5, after clicking on the *Evaluate* button, the GUI displays statistics of all the accuracy metrics discussed.


Arabic OCR Evaluation Tool

Total	78.57%	Below Baseline	66.67%
One Dot	66.67%	Loop	80%
Two Dot	66.67%	Digit	66.67%
Three Dot	100%	Punctuation	28.57%
Zigzag	50%	Diacritic	40%
Non-dot	91.89%	Initial	78.95%
Above Baseline	80%	Middle	63.41%
End	68.42%	Isolated	71.43%

Figure 5. GUI of the Arabic OCR evaluation tool

The Arabic OCR evaluation tool is provided freely as open-source software at the following GitHub address: https://github.com/NLPBangor/OCREvaluationTool.

In order to illustrate how the tool works, an example is provided here. Fig. 6 shows the text image that contains Arabic text with digits, diacritics and punctuation marks. The open source OCR Tesseract engine[1], which supports Arabic, was run to convert the Arabic text image to an editable text. Fig. 7 displays the corresponding OCR-generated text. To evaluate the accuracy of the utilised OCR application, the tool was run on the OCR output text and the ground truth text. The result of the evaluation is presented in Fig. 5.

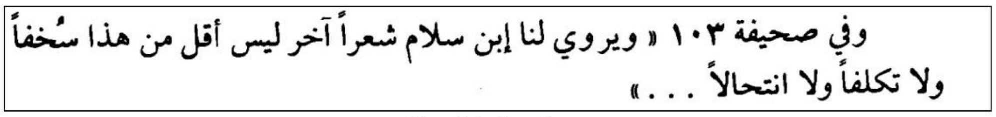

وفي صحيفة ١٠٣ « ويروي لنا إبن سلام شعراً آخر ليس أقل من هذا سُخفاً
ولا تكلفاً ولا انتحالاً . . . »

Figure 6. Arabic text image

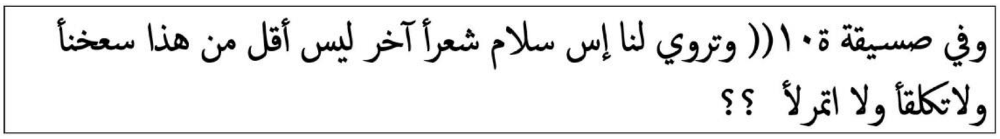

وفي صسيقة ة١٠((وتروي لنا إس سلام شعرأ آخر ليس أقل من هذا سعخنأ
ولاتكلقأ ولا اتمرلأ ؟؟

Figure 7. OCR output text for image in Fig. 6

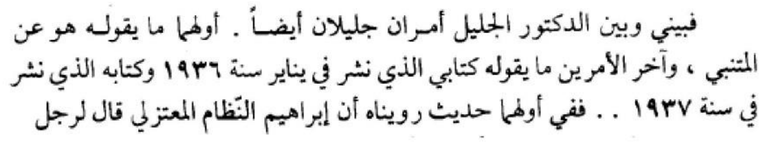

فبيني وبين الدكتور الجليل أمـران جليلان أيضـاً . أولهما ما يقولـه هو عن
المتنبي ، وآخر الأمرين ما يقوله كتابي الذي نشر في يناير سنة ١٩٣٦ وكتابه الذي نشر
في سنة ١٩٣٧ . . ففي أولهما حديث رويناه أن إبراهيم النّظام المعتزلي قال لرجل

Figure 8. Second example of an Arabic text image

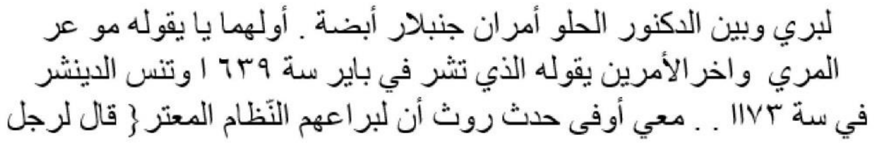
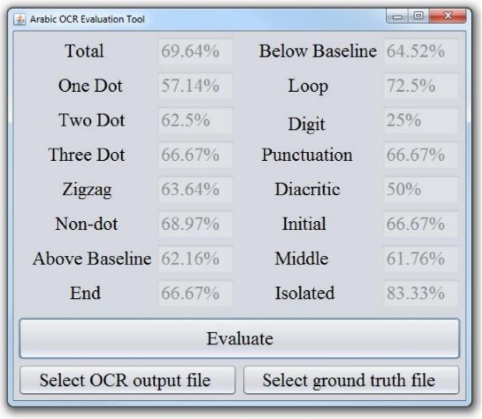

لبري وبين الدكنور الحلو أمران جنبلار أبضة . أولهما يا يقوله مو عر
المري واخرالأمرين يقوله الذي تشر في باير سة ٦٣٩ ا وتنس الدينشر
في سة ١١٧٣ . . معي أوفى حدث روث أن لبراعهم النّظام المعتر{ قال لرجل

Figure 9. OCR output text of second image text in Fig. 8

This example illustrates the idea that evaluation of Arabic OCR systems according to different accuracy measures is required. In other words, it can be seen from the results in Fig. 5 that the total character accuracy is about 78%. However, the accuracy results for individual character classes are significantly different. In particular, the accuracy of the one dot characters group and the two dots characters group are around 67%, whereas, the accuracy of the three dot character class is 100%. Moreover, the observed difference between the accuracy of the zigzag characters group and the loop characters group is significant, 50% and 80% respectively. Another important finding is that the accuracy of the above baseline characters class (80%) is very different to the accuracy of below baseline characters (67%). Also, it can clearly be seen that the punctuation class has the lowest accuracy rate.

Another example is provided in order to show the importance of the accuracy metrics, which have been provided in this paper, in evaluating the performance of Arabic OCR systems. The text image and the corresponding OCR-generated text are shown in Fig. 8 and Fig. 9 respectively. Fig. 10 presents the results obtained from the evaluation of the Tesseract OCR system on the text image which are clearly different to the previous example.



Total	69.64%	Below Baseline	64.52%
One Dot	57.14%	Loop	72.5%
Two Dot	62.5%	Digit	25%
Three Dot	66.67%	Punctuation	66.67%
Zigzag	63.64%	Diacritic	50%
Non-dot	68.97%	Initial	66.67%
Above Baseline	62.16%	Middle	61.76%
End	66.67%	Isolated	83.33%

Figure 10. Evaluation results of the sceond test of the Arabic Tesseract OCR software

Based on the findings above, it is possible to state that measuring Arabic OCR performance in terms of the accuracy metrics, which are implemented in the Arabic OCR evaluation tool, can help researchers to analyse the strengths and weaknesses of Arabic OCR systems, helps to determine open problems in Arabic OCR systems, and compare alternative OCR systems in order to enhance the accuracy of OCR systems.

VI. Conclusion

A software tool for performance evaluation, based on different objective accuracy metrics, has been described. This tool has been specifically developed to assist Arabic OCR researchers to calculate the accuracy of different Arabic OCR systems. Furthermore, by using this automated tool, an OCR evaluator can quickly process an enormous number of testing experiments. Thus, the authors believe that the Arabic OCR evaluation tool that has been presented in this paper will be very useful by the OCR community.

For future work, the evaluation tool will be enhanced. For example, more string similarity metric algorithms other than Levenshtein edit distance algorithms would allow comparison of the results of different algorithms.

References

[1] B. Al-Badr and R. M. Haralick, "Segmentation-free word recognition with application to Arabic," *Proc. 3rd Int. Conf. Doc. Anal. Recognit.*, vol. 1, pp. 355–359, 1995.

[2] M. T. Parvez and S. a. Mahmoud, "Offline arabic handwritten text recognition: A Survey," *ACM Comput. Surv.*, vol. 45, no. 2, pp. 23:1–23:35, 2013.

[3] F. Slimane, S. Kanoun, J. Hennebert, A. M. Alimi, and R. Ingold, "A study on font-family and font-size recognition applied to Arabic word images at ultra-low resolution," *Pattern Recognit. Lett.*, vol. 34, no. 2, pp. 209–218, 2013.

[4] S. V Rice, "Measuring the accuracy of page-reading systems," University of Nevada, Las Vegas, 1996.

[5] T. Kanungo, G. A. Marton, and O. Bulbul, "OmniPage vs. Sakhr: Paired model evaluation of two Arabic OCR products," in *Electronic Imaging'99*, 1999, pp. 109–120.

[6] S. Mihov, K. U. Schulz, C. Ringlstetter, V. Dojchinova, V. Nakova, K. Kalpakchieva, O. Gerasimov, A. Gotscharek, and C. Gercke, "A corpus for comparative evaluation of OCR software and postcorrection techniques," *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 2005, pp. 162–166, 2005.

[7] S. Taha, Y. Babiker, and M. Abbas, "Optical character recognition of arabic printed text," *SCOReD 2012 - 2012 IEEE Student Conf. Res. Dev.*, pp. 235–240, 2012.

[8] I. Supriana and A. Nasution, "Arabic Character Recognition System Development," *Procedia Technol.*, vol. 11, no. Iceei, pp. 334–341, 2013.

[9] H. Pirsiavash, R. Mehran, and F. Razzazi, "A robust free size OCR for omni-font persian/arabic printed document using combined MLP/SVM," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3773 LNCS, pp. 601–610, 2005.

[10] M. Dahi, N. A. Semary, and M. M. Hadhoud, "Primitive printed Arabic Optical Character Recognition using statistical features," in *2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, 2015, pp. 567–571.

[11] I. Ahmad, S. A. Mahmoud, and G. A. Fink, "Open-vocabulary recognition of machine-printed Arabic text using hidden Markov models," *Pattern Recognit.*, vol. 51, pp. 97–111, 2016.

[12] S. Saber, A. Ahmed, A. Elsisi, and M. Hadhoud, "Performance Evaluation of Arabic Optical Character Recognition Engines for Noisy Inputs," in *The 1st International Conference on Advanced Intelligent System and Informatics (AISI2015), November 28-30, 2015, Beni Suef, Egypt*, 2016, pp. 449–459.

[13] R. C. Carrasco, "An Open-source OCR Evaluation Tool," in *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, 2014, pp. 179–184.

[14] T. A. Nartker, S. V Rice, and S. E. Lumos, "Software tools and test data for research and testing of page-reading OCR systems," in *In International Symposium on Electronic Imaging Science and Technology*, 2005.

[15] T. Kanungo, G. a Marton, and O. Bulbul, "Performance evaluation of two Arabic OCR products," *Proc. SPIE*, pp. 76–83, 1999.

[16] S. Saber, A. Ahmed, and M. Hadhoud, "Robust metrics for evaluating arabic OCR systems," in *Image Processing, Applications and Systems Conference (IPAS), 2014 First International*, 2014, pp. 1–6.

[17] J. Kanai, T. Nartker, and S. Rice, "Performance Metrics for Document Understanding Systems," *Doc. Anal.*, pp. 424–427, 1993.

[18] W. J. Teahan, S. Inglis, J. G. Cleary, and G. Holmes, "Correcting English text using PPM models," in *Data Compression Conference Proceedings*, 1998.

[19] E. Borovikov and W. Lane, "A survey of modern optical character recognition techniques (DRAFT)," vol. 1, no. 301, pp. 1–37, 2004.

[20] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Sov. Phys. Dokl.*, vol. 10, no. 8, pp. 707–710, 1966.

[21] S. Tanner, T. Muñoz, and P. H. Ros, "Measuring Mass Text Digitization Quality and Usefulness," *D-Lib Mag.*, vol. 15, no. 7/8, 2009.

[22] S. V Rice, J. Kanai, and T. A. Nartker, "An evaluation of OCR accuracy," *Inf. Sci. Res. Institute, 1993 Annu. Res. Rep.*, vol. 9, p. 20, 1993.

[23] Y. Batawi and O. Abulnaja, "Accuracy Evaluation of Arabic Optical Character Recognition Voting Technique: Experimental Study," *Int. J. Electr. Comput. Sci.*, vol. 2, no. 1, pp. 29–33, 2012.

Experimental evaluation of Arabic OCR systems

Mansoor Alghamdi
University of Tabouk, Tabouk, Saudi Arabia, and
William Teahan
School of Computer Science, Bangor University, Bangor, UK

Received 4 May 2017
Revised 2 August 2017
Accepted 6 September 2017

Abstract

Purpose – The aim of this paper is to experimentally evaluate the effectiveness of the state-of-the-art printed Arabic text recognition systems to determine open areas for future improvements. In addition, this paper proposes a standard protocol with a set of metrics for measuring the effectiveness of Arabic optical character recognition (OCR) systems to assist researchers in comparing different Arabic OCR approaches.

Design/methodology/approach – This paper describes an experiment to automatically evaluate four well-known Arabic OCR systems using a set of performance metrics. The evaluation experiment is conducted on a publicly available printed Arabic dataset comprising 240 text images with a variety of resolution levels, font types, font styles and font sizes.

Findings – The experimental results show that the field of character recognition for printed Arabic still requires further research to reach an efficient text recognition method for Arabic script.

Originality/value – To the best of the authors' knowledge, this is the first work that provides a comprehensive automated evaluation of Arabic OCR systems with respect to the characteristics of Arabic script and, in addition, proposes an evaluation methodology that can be used as a benchmark by researchers and therefore will contribute significantly to the enhancement of the field of Arabic script recognition.

Keywords Performance evaluation, Performance metrics, Arabic OCR, OCR

Paper type Research paper

Introduction

Optical character recognition (OCR) is a technique that aims to automatically convert a machine-printed or handwritten text image into an editable text format (Alghamdi *et al.*, 2016). This technique is highly desirable in various real-world applications, such as digitising learning resources to assist visually impaired people, bank cheque processing and mail sorting (Alginahi, 2013; Al-Badr and Mahmoud, 1995). Generally, the process for developing OCR systems involves five stages: pre-processing, segmentation, feature extraction, classification and post-processing. In each stage, specific techniques are applied; for more details, see Khorsheed (2002).

Previous research on text recognition has focused primarily on Latin scripts, such as English and Chinese, and it has not been until the last two decades that recognition of non-Latin scripts, such as Arabic, have been researched (Alginahi, 2013). Although

PSU Research Review
Vol. 1 No. 3, 2017
pp. 229-241
Emerald Publishing Limited
2399-1747
DOI 10.1108/PRR-05-2017-0026

handwritten script is significantly more challenging than printed Arabic text for OCR, Arabic printed text OCR still poses significant challenges (Alghamdi *et al.*, 2016). Therefore, this study will deal only with Arabic printed text. Figure 1 illustrates the characteristics of Arabic printed text that contribute to the inadequate development of Arabic script recognition. Arabic script is written cursively through a baseline and contains loop-shaped characters, zigzag-shaped characters, dot characters and diacritics. Moreover, a character might have up to four different shapes in relation to its position in a word. Therefore, research is being undertaken seeking more solutions for Arabic OCR systems (Parvez and Mahmoud, 2013; Slimane *et al.*, 2013).

To produce an efficient Arabic OCR system, effective performance evaluation of current OCR systems is essential. Furthermore, evaluating OCR performance contributes to monitoring progress in OCR system development, analysing the effectiveness of OCR systems, identifying open areas and providing a scientific explanation for the performance of OCR systems (Kanungo *et al.*, 1999a; Mihov *et al.*, 2005).

Despite the significance of Arabic OCR system performance evaluation, relatively little work has been published on empirical analysis of the effectiveness of Arabic OCR systems. For instance, two studies provide an evaluation of two Arabic OCR systems (Kanungo *et al.*, 1999a, 1999b). However, as these studies were conducted in 1999, over 17 years ago, they do not reflect current progress in the field of Arabic OCR development (Alginahi, 2013). A more recent empirical study provides a comparative evaluation of the most common Arabic OCR systems (Saber *et al.*, 2016). However, the study by these authors only investigated the effectiveness of input quality images on the performance of Arabic OCR systems. In addition, exploring Arabic OCR systems in relation to their sensitivity to different levels of page quality may not be adequate in fairly assessing their success, as some OCR systems include a combination of image enhancement techniques. To the best of the authors' knowledge, no established work has gauged the current progress in the enhancement of Arabic OCR in terms of the challenges of Arabic script.

Moreover, the performance assessments of Arabic OCR systems are only reported by their developers: their results are derived from different datasets that might be small or might be used in developing the systems (Al-Badr and Mahmoud, 1995; Alginahi, 2013). Consequently, as these performance tests are statistically invalid, they cannot be used to compare the performance between Arabic OCR systems (Margner and El Abed, 2009). Evaluating the performance of Arabic OCR systems is also challenging as no standard dataset is available nor is a set of performance metrics freely available to the community of Arabic OCR developers (Al-Muhtaseb and Qahwaji, 2011; Abdelraouf *et al.*, 2008; Al-Badr and Mahmoud, 1995; Ahmad *et al.*, 2016). In addition, most reports on the performance of Arabic OCR systems are in terms of the general, standard performance measurement of character accuracy, such as in Dahi *et al.* (2015) and Ahmad *et al.* (2016). However, this performance metric is insufficient to assess how Arabic OCR systems are coping with the challenges of Arabic script.

Thus, the current work first attempts to provide a better insight into the effectiveness of the state-of-the-art printed Arabic OCR systems with possible interpretations for future performance enhancement. It then aims to propose a standard protocol with a set of metrics

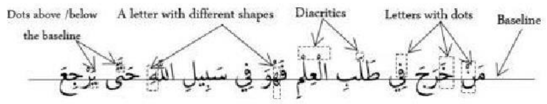

Dots above /below the baseline
A letter with different shapes
Diacritics
Letters with dots
Baseline
مَنْ خَرَجَ فِي طَلَبِ الْعِلْمِ فَهُوَ فِي سَبِيلِ اللهِ حَتَّى يَرْجِعَ

Figure 1.
Printed Arabic script characteristics

for measuring the effectiveness of Arabic OCR systems which we hope will be used as a benchmark by researchers in comparing between OCR algorithms.

This paper is organised as follows: in the first section below, the most common Arabic OCR systems are introduced. The Arabic OCR system evaluation background and performance metrics for Arabic OCR system evaluation are discussed in the second and third sections, respectively. An experimental protocol is presented in the fourth section. The experimental results are then presented and discussed. In the final section, future work is suggested and the conclusion is presented.

Arabic OCR systems

Only a handful of OCR systems claim that they are capable of recognising Arabic script. Our evaluation study is limited to the four most well-known Arabic OCR systems, namely, Automatic Reader 11.2 produced by the Sakhr Software Company; FineReader 12 produced by the ABBYY Company; Clever Page produced by RDI (Research & Development International) and Tesseract produced originally by Hewlett-Packard (HP). In the following subsections, the four Arabic OCR engines are briefly discussed.

Automatic reader 11.2 software

Automatic Reader is a commercial product first developed by the Sakhr Software Company in 1982 for text recognition of Arabic script. It supports Arabic language and several Arabic character-based languages, such as Arabic, Farsi and Urdu. Sakhr claims that Automatic Reader has been ranked as the best existing Arabic OCR software for high-quality text images by US government evaluators (Sakhr Software OCR, 2017). It supports multi-font type and multi-resolution images. However, font size 8 is not supported by the Automatic Reader OCR software (henceforth, referred to as Sahkr OCR).

FineReader 12 software

FineReader is produced commercially by a global company, called ABBYY, as advanced OCR software. The performance of FineReader has been enhanced by ABBYY for many years. FineReader 12 supports 190 languages including Arabic script using dictionary support (Abbyy OCR, 2017). It supports multi-font types, multi-size and multi-resolution images. Henceforth, FineReader will be referred to as ABBYY OCR.

Clever page software

Clever Page originally began as a PhD research study by El-Mahallawy (2008). Since 2008, Clever Page has been designed and developed as an Omni font-written Arabic OCR engine by Research & Development International (RDI). It supports multi-font types and multi-size text images. However, it is worth mentioning that the Clever Page OCR software only works on pages with 300 dots per inch (dpi). Henceforth, Clever Page will be referred to as RDI OCR.

Tesseract software

Tesseract is an OCR engine designed at Hewlett-Packard (HP) between 1984 and 1994. Since late 2005, it has been maintained by Google and released as open source OCR software. However, Arabic support has only been added recently (Sabbour and Shafait, 2013). Tesseract is the only Arabic OCR software that is freely available. It supports multi-font types, and multi-size and multi-resolution images.

Arabic OCR system performance evaluation background

Evaluation of OCR systems can be classified into two types: black-box evaluation and white-box evaluation (Kanungo *et al.*, 1999b). In black-box evaluation, an entire OCR system is treated as an indivisible unit; thus, the evaluators do not have access to the submodules of the OCR system. Furthermore, this evaluation type is only concerned with the output of the OCR system rather than how it is produced. On the other hand, in white-box evaluation, the evaluator must have access to the submodules of the OCR system to evaluate each submodule (Kanungo *et al.*, 1999a). As accessing the submodules of commercial Arabic OCR systems was not possible for the authors of this paper and as our interest is only in error analysis of Arabic OCR system output, the white-box evaluation type is outside the scope of this evaluation study.

To evaluate OCR systems, comparison between an observed variable, which is the output text of the OCR system, and a reference variable, which is the original text called "ground-truth", is required (Kanai *et al.*, 1993; Teahan *et al.*, 1998).

Metrics for Arabic OCR system performance evaluation

Generally, Arabic OCR systems are evaluated in terms of character accuracy with this obtained by identifying the differences between the ground-truth text and the OCR output text. These differences can be determined by the edit distance which is the minimum number of edit operations required to correct the OCR output text to be matched with the ground-truth text. These edit operations are: character insertion, character deletion and character substitution; for more details refer to Levenshtein (1966).

This general performance metric is not sufficient to measure the performance of an Arabic OCR system, as the accuracy rates of Arabic OCR systems are comparable (Saber *et al.*, 2016). Moreover, the character accuracy metric only provides us with a measure of how well an Arabic OCR system performs in text recognition in general terms. Thus, it cannot provide us with insight into which systems have overcome the various challenges of Arabic script.

On the other hand, a recent study (Alghamdi *et al.*, 2016) tackles this problem by suggesting various objective performance metrics for evaluating Arabic OCR systems which can provide us with more insight into the effectiveness of Arabic OCR systems. Therefore, the current study adopts these metrics to evaluate the performance of Arabic OCR systems. The adopted performance metrics are defined in the following subsections.

Overall character accuracy

Overall character accuracy determines the accuracy of Arabic OCR over all of the tested text images. Character accuracy is the percentage of ground-truth characters that are recognised correctly on an Arabic text image, by comparing the ground-truth text file with the OCR output text file. In accordance with Alghamdi *et al.* (2016), character accuracy is determined by equation (1):

$$\frac{m-e}{m} \times 100 \tag{1}$$

where m is the number of characters in the ground-truth text file and e is the edit distance. The cost for each edit operation is defined as 1.

Character accuracy based on character position

As previously mentioned, an Arabic character may have one to four shapes depending on its position in a word; for example, see Table I. The four possible shapes are isolated, initial, middle and end. Thus, it is valuable to analyse the effectiveness of Arabic OCR systems in recognising isolated, initial, middle and end characters. To analyse accuracy, Arabic characters have been categorised into isolated, initial, middle and end classes by Alghamdi *et al.* (2016), as shown in Table I. The accuracy of each class is determined by equation (1).

Dot character accuracy and no-dot character accuracy

One of the challenges of Arabic OCR is the presence of dots in Arabic script (Alghamdi and Teahan, 2017). Assessing the impact of dot and no-dot characters on the performance of Arabic OCR systems is therefore of interest. To do so, Arabic characters have been categorised into four classes: one dot class, two dot character class, three dot character class and no-dot character class by Alghamdi *et al.* (2016), as illustrated in Table II. The accuracy of each class is determined by equation (1).

Dot character accuracy based on baseline

The presence of a baseline is specific to Arabic script characteristics. The baseline is significant in developing Arabic OCR systems. Alghamdi *et al.* (2016) classify Arabic characters into two classes: dot character above the baseline and dot character below the baseline to compare the performance of Arabic OCR in each class, as illustrated in Table III. The accuracy of each class is also determined by equation (1).

Zigzag-shaped character accuracy

One of the distinguishing characteristics of Arabic script is the presence of a zigzag-shaped mark (ء), called *Hamza*, with some Arabic characters. The aim of using this metric is to expose the sensitivity of Arabic OCR systems to zigzag-shaped characters. Alghamdi *et al.* (2016) compute the zigzag-shaped character accuracy by using equation (1).

Isolated	Initial	Middle	End
ه	ﻫ	ـﻬـ	ﻪ

Table I. An Arabic character with dissimilar shapes

One dot	Two dots	Three dots	No-dot
ب ج خ ذ ز ض ظ غ ف ن	ﻘ ي ة	ش ث	ﺣ ر س ص ط ع ل م ه و ك

Table II. Examples of dot characters and no-dot characters

Dot character above baseline	Dot character below baseline
ق ن ف غ ظ ض ش ز ذ خ ث ت	ي ج ب

Table III. Examples of dot characters based on the baseline

Loop-shaped character accuracy
Several Arabic characters have a loop shape, such as *Saad* (ص), *Dhad* (ض), Fa (ف, *Meem* (م) and *Qaf* (ق). According to Alghamdi *et al.* (2016), an obstacle for Arabic OCR is recognising Arabic characters that contain a loop shape. To assess the effectiveness of Arabic OCR systems for recognising these characters, the accuracy of loop-shaped characters is also computed by equation (1).

Diacritics accuracy
Some Arabic text may be written with diacritical marks, as previously illustrated in Figure 1. Thus, it is essential to evaluate the performance of Arabic OCR systems in recognising Arabic diacritical marks. As before, diacritical mark accuracy is determined by equation (1).

Digit accuracy
The digit accuracy metric is used to determine the performance accuracy of Arabic OCR systems in recognising Hindi–Arabic digits (Alghamdi *et al.*, 2016). Digit accuracy is determined by equation (1).

Punctuation accuracy
The punctuation accuracy metric is used to assess the performance accuracy of Arabic OCR systems in recognising punctuation symbols (Alghamdi *et al.*, 2016). Punctuation accuracy is determined by equation (1).

Experimental protocol
Very few Arabic datasets are freely available to researchers. The most widely used dataset for evaluating Arabic OCR approaches is the Arabic Printed Text Image (APTI). This public dataset was developed by Slimane *et al.* (2009). However, APTI is a word-level dataset where each text image contains only one Arabic word. Another Arabic dataset is ALPH-REGIM which is provided by Ben Moussa *et al.* (2010). This dataset contains about 5,000 printed and handwritten Arabic text images. Compared to the APTI dataset, ALPH-REGIM is a paragraph-based text image dataset. However, the text images are only available in one font size and at one resolution level. Thus, the KAFD dataset, developed by Luqman *et al.* (2014), is used for evaluating the performance of the four Arabic OCR systems.

The KAFD dataset is freely available and is a page-level dataset where each text image consists of a text that resulted in 2,576,024 line images. It has Arabic printed text images and corresponding ground-truth text files. These text images are available at 100 dpi, 200 dpi and 300 dpi. Furthermore, it comprises text images of 40 Arabic font types, 10 pitch sizes and four styles, with resolutions of 100 dpi, 200 dpi, 300 dpi and 600 dpi.

For the current work, 10 different font types of the text image, in which the forms of the characters are of various types, are selected, namely, Andalus, Arabic Transparent, AdvertisingBold, Diwani Letter, DecoType Thuluth, Simplified Arabic, Tahoma, Traditional Arabic, DecoType Naskh and M Unicode Sara. These font types are selected based on the level of complexity of the writing style in printed Arabic script, ranging from simple fonts with no overlaps and ligatures, such as AdvertisingBold, to more complex fonts with overlaps, such as Diwani Letter. For each font type, 8, 12, 18 and 24 pitch sizes have been used to highlight the effectiveness of Arabic OCR systems on specific pitch sizes. Moreover, to enable the evaluation of the performance of Arabic OCR systems in terms of font style, normal and italic font styles are used. To study the influence of the page resolution level on Arabic OCR system performance, all text images in the above set have been randomly selected at 100 dpi, 200 dpi and 300 dpi. This resulted in 80 text images for each resolution. Thus, this experiment is performed on 240 text images in TIFF format.

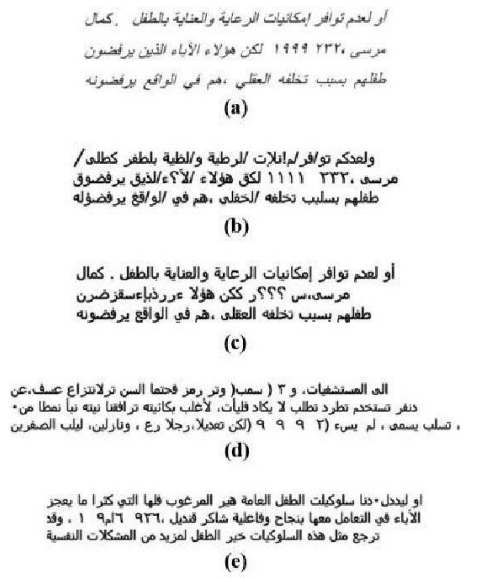
The experiment was conducted on the four Arabic OCR systems, as previously discussed, namely, Sakhr, ABBYY, RDI and Tesseract OCR systems. Sakhr and ABBYY OCR systems were kindly provided to the authors by their developers, whereas the RDI OCR system output was obtained by sending the dataset to the system's developer to apply the system to the test images.

To statistically assess the performance of each Arabic OCR system, the performance metrics described previously were used. In particular, the output text files of each Arabic OCR system were used to compute the quantities of each performance metric, corresponding to the ground-truth text files for the dataset. To eliminate human error, improve speed and precision and reduce repetition, an automated open access tool for evaluating Arabic OCR systems, provided by Alghamdi *et al.* (2016), was used to obtain the statistical data. A sample of a text image from the dataset and the corresponding Arabic OCR output are visually illustrated in Figure 2.

Experimental results and discussion

The experimental results, obtained from the evaluation experiment discussed in the previous section, are presented in this section to analyse the effectiveness of the evaluated Arabic OCR systems in printed Arabic text recognition.

The overall character accuracy scores for the four evaluated Arabic OCR systems over different resolutions are presented in Figure 3. It is apparent that the Arabic OCR systems are affected by the resolution of the text images. In particular, the results shows a gradual increase in the overall character accuracy of all Arabic OCR systems when increasing the



او لعدم توافر إمكانيات الرعاية والعناية بالطفل . كمال
مرسي، ٢٣٢ ١٩٩٩ لكن هؤلاء الأباء الذين يرفضون
طفلهم بسبب تخلفه العقلي ،هم في الواقع يرفضونه

(a)

ولعدكم تو/فر/م!نلات /لرطية و/لظية بلطفر كطلى/
مرسى ،٢٣٢ ١١١١ لكق هؤلاء /لآ؟ء/لذيق يرفضوق
طفلهم بسليب تخلفه /لخفلي ،هم في /لو/قع يرفضؤله

(b)

أو لعدم توافر إمكانيات الرعاية والعناية بالطفل . كمال
مرسى،س ؟؟؟ر ككن هؤلا ءرزذبإءسقزضرن
طفلهم بسيب تخلفه العقلى ،هم فِي الواقع يرفضونه

(c)

الى المستشفيات، و ٣ (سمب(وتر رمز فحتما السن ترلانتزاع عسف،عن
دنفر تستخدم تطرد تطلب لا يكاد قليأت، لأغلب بكانيته ترافقنا نيته نبأ نمطا من•
، تسلب يسمى ، لم يسء (٢ ٩ ٩ ٩ ٩ (لكن تعديلا،رجلا رع ، ونارلين، ليلب الصفرين

(d)

او ليدحل•دنا سلوكيلت الطفل العامة هير المرغوب فلها التي كترا ما يعجز
الأباء في التعامل معها بنجاح وفاعلية شاكر قنديل ،٩٣٦ ٦ام٩ ١ ، وقد
ترجع مثل هذه السلوكيات خير الطفل لمزيد من المشكلات النفسية

(e)

Figure 2.
(a) A text image from the KAFD dataset; (b) output of the Sakhr OCR system; (c) output of the ABBYY OCR system; (d) output of the RDI OCR system and (e) output of the tesseract OCR system

resolution of text images from 100 dpi to 300 dpi. In addition, a clear upward trend is apparent in the overall character accuracy of Sakhr OCR when the resolution of text images increased from 100 dpi to 200 dpi. Interestingly, high scores of character accuracy at 100 dpi, 200 dpi and 300 dpi were obtained by the ABBYY OCR system. Crucially, this system among the four Arabic OCR systems has the highest character accuracy at 100 dpi, 200 dpi and 300 dpi of 46, 60 and 62 per cent, respectively.

The accuracy of Arabic OCR systems in recognising Arabic characters based on their position in a word are provided in Figure 4. As has been mentioned, the connectivity feature of Arabic script is an obstacle to Arabic text recognition. This is supported by the experiment data which indicate that all of the evaluated Arabic OCR systems have higher accuracy in recognising isolated characters when they are not connected with other characters in a word. On the other hand, the performance of the Arabic OCR systems decreased in recognising initial, middle and end characters, as shown in Figure 4. Compared to the Arabic OCR systems' accuracy in recognising isolated characters, the low recognition accuracy of initial, middle and end characters by the evaluated systems is possibly because of the segmentation algorithms implemented by these systems, as the segmentation process is not required for recognising isolated characters. However, a new methodology is needed to evaluate the segmentation stage in Arabic OCR systems to gain a better understanding of these results.

The results of the recognition accuracy performance of the Arabic OCR systems in terms of one dot, two dot, three dot and no-dot characters are illustrated in Figure 5. It is obvious that the recognition accuracy rates of no-dot characters are significantly better for all evaluated Arabic OCR systems, compared to one, two and three dot characters. This confirms that one of the challenges in Arabic OCR development is the presence of dots in Arabic script, as discussed previously. It has been hypothesised that a thinning algorithm, as a pre-processing technique, has an influence on the recognition accuracy of dot characters (Hosseini, 1997; Alghamdi and Teahan, 2017). In particular, some thinning algorithms may remove the dots of Arabic characters which results in misrecognising these characters, as

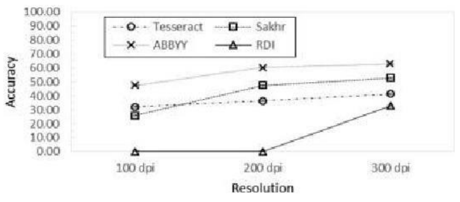


Figure 3.
Overall accuracy versus resolution results from the OCR system evaluation

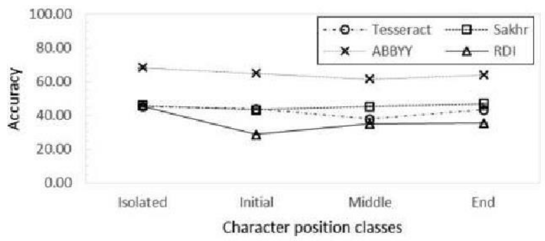


Figure 4.
Character accuracy in terms of character position results from the OCR system evaluation

stated in Alghamdi and Teahan (2017). Therefore, these results are likely to be related to thinning algorithms used in the pre-processing stage of the Arabic OCR system.

The results of analysing the performance of the evaluated Arabic OCR systems on characters that have a dot above or below the baseline are presented in Figure 6. These results highlight that Arabic OCR systems perform much better in identifying characters with a dot below the baseline than characters with a dot above the baseline. The reasons for these results are not entirely clear. However, one technique used in the pre-processing stage for developing Arabic OCR systems is page decomposition which separates the lines of a text block in a text image. To be specific, some researchers emphasise that considerable attention must be paid in the page decomposition of Arabic text images to ensure that dots placed above or below a line of text are not separated (Al-Badr and Mahmoud, 1995; Sami El-Dabi *et al.*, 1990). Thus, a possible reason for this finding may be because of the page decomposition process.

Figure 7 compares the recognition accuracy for zigzag-shaped characters and loop-shaped characters by the evaluated Arabic OCR systems. It is apparent from the results that the performance rates of Arabic OCR systems in accurately recognising loop-shaped characters are higher than in recognising zigzag-shaped characters. It can thus be assumed that recognising characters with a zigzag shape is more challenging than recognising characters that have a loop shape.

Figure 8 presents the recognition accuracy of the evaluated Arabic OCR systems in terms of the digits, punctuation and diacritics groups. Surprisingly, Tesseract and RDI OCR

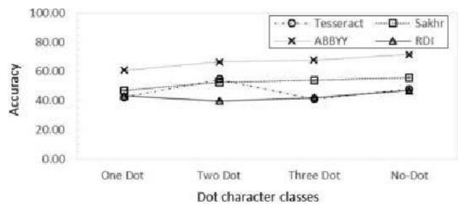


Figure 5. Dot character accuracy versus no-dot character accuracy results from the OCR system evaluation

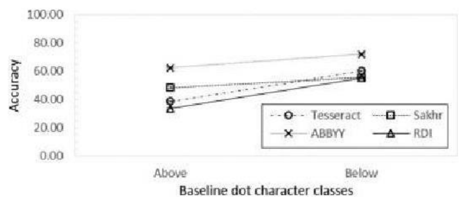


Figure 6. Dot character accuracy based on baseline results from the OCR system evaluation

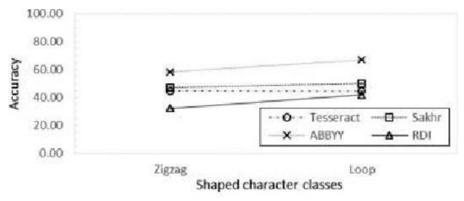


Figure 7. Character accuracy based on shaped character results from the OCR system evaluation

systems were not able to recognise any diacritical marks. However, one possible reason is that these OCR systems were not trained to recognise diacritics. In addition, Figure 8 indicates that the Sakhr OCR system was unable to recognise Arabic digits (Hindu digits). As this result was not expected, we undertook further investigation to find an explanation. After manual inspection of the output of the Sakhr OCR system, we discovered that the system recognises Arabic digits as English digits. In other words, the system replaced the Arabic digits with English digits when producing the output. It is worth mentioning that Sakhr OCR system obtained an 84 per cent digit accuracy rate when allowing English digits as not error. Thus, improved accuracy rates for the Sakhr OCR system could be achieved if the system were to overcome this problem.

The respective accuracy of the Arabic OCR systems in relation to different font pitch sizes is plotted in Figure 9. It can be seen that a drop in character recognition accuracy occurs when reducing the font pitch size. It can thus be concluded that the evaluated systems struggle when the font pitch size is too small, such as a font pitch of 8. Another important finding from this result is that the RDI OCR system performs better for a font pitch of 12 than for a font pitch of 24. The reason for this finding is unclear:

The recognition performance analysis of the evaluated Arabic OCR systems on fonts with an italic style compared to those with a non-italic font style is shown in Figure 10. Our

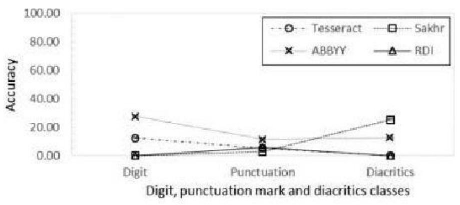


Figure 8.
Digit, punctuation and diacritics accuracy results from the OCR system evaluation

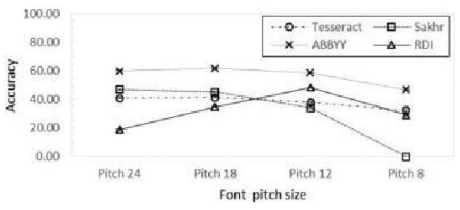


Figure 9.
Arabic OCR accuracy versus font pitch size results from the OCR system evaluation

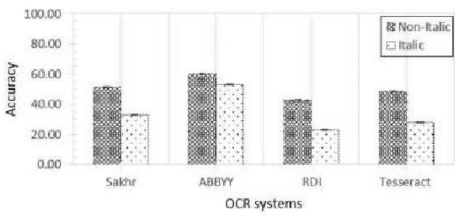


Figure 10.
Italic and non-italic font accuracy comparison results from the OCR system evaluation

experimental results show that, for all evaluated systems, the recognition accuracy rates for a non-italic font style are higher than for an italic font style. The low recognition accuracy percentages for an italic font style are likely to be related to the fact that the italic font style is a cursive font.

Table IV reports on the accuracy of the evaluated Arabic OCR systems in terms of font types. As expected, the low accuracy rates for all evaluated systems in recognising the Diwani Letter font are because of the complexity of this font which contains overlaps.

In summary, most of the performance accuracy rates of the evaluated Arabic OCR systems are below 75 per cent, indicating the continuing need for improvements in the recognition of printed Arabic script. Moreover, the correlation between the performance accuracy of Arabic OCR systems and the features of Arabic script have been highlighted. In particular, the experimental analysis indicates that the characteristics of printed Arabic script, such as the connectivity, and the presence of dots and zigzag shapes, all contribute to the challenge for Arabic OCR systems. Overall, the results indicate that recognition of Arabic script remains an open research problem.

Conclusion and further work

This paper presents the results of a performance evaluation of four well-known Arabic OCR systems: Sakhr, ABBYY, RDI and Tesseract. In addition, this study provides an experimental protocol with a set of objective performance metrics that enables the effectiveness of different Arabic OCR systems to be compared. The results of this study show that all the evaluated Arabic OCR systems have low performance accuracy rates, below 75 per cent, which means that the time which would take to manually correct the OCR output would be a prohibitive. On the other hand, the recognition accuracy rates for isolated characters are higher than for initial, middle and end characters. Moreover, the recognition accuracy rates of no-dot characters are significantly better compared to one, two and three dot characters. The current evaluation study highlights several open areas and major factors that need to be considered when either developing Arabic OCR systems or enhancing the current systems. For instance, the experimental results indicate that recognition of Arabic text with complex font, such as the Diwani Letter font, and text with diacritics are still open research problems. For future work, it will be interesting to assess the effectiveness of applying post-processing methods, such as spelling correction, on the performance accuracy rates of Arabic OCR systems. Furthermore, a methodology should be developed to evaluate the performance of each stage of Arabic OCR which would enable

	OCR System			
Font Type	Sakhr (%)	ABBYY (%)	RDI (%)	Tesseract (%)
Traditional Arabic	48.54	67.66	51.88	47.04
Tahoma	40.52	69.91	26.38	38.37
Simplified Arabic	52.97	67.69	44.94	46.75
M Unicode Sara	36.03	59.40	25.92	33.72
Diwani letter	18.13	18.47	18.13	23.32
DecoType Thuluth	36.12	37.71	24.26	32.48
DecoType Naskh	48.88	50.22	41.63	40.92
Arabic transparent	51.56	75.19	46.00	48.61
Andalus	28.07	37.53	21.68	25.34
AdvertisingBold	57.35	70.26	27.20	39.39

Table IV. Arabic OCR performance accuracy according to 10 different font types

the effects of each stage on the overall performance of Arabic OCR systems to be determined. Further work is also needed to evaluate other Arabic OCR systems. Furthermore, it will be interesting to assess the effectiveness of evaluation experiment of Arabic OCR systems on handwritten Arabic script.

References

Abbyy OCR (Optical Character Recognition) (2017), available at: www.abbyy.com/en-gb/ (accessed 15 April 2017).

Abdelraouf, A., Higgins, C.A. and Khalil, M. (2008), "A database for Arabic printed character recognition", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Berlin Heidelberg, pp. 567-578.

Ahmad, I., Mahmoud, S.A. and Fink, G.A. (2016), "Open-vocabulary recognition of machine-printed Arabic text using hidden Markov models", *Pattern Recognition*, Vol. 51, pp. 97-111, available at: www.sciencedirect.com/science/article/pii/S0031320315003428

Al-Badr, B. and Mahmoud, S.A. (1995), "Survey and bibliography of Arabic optical text recognition", *Signal Processing*, Vol. 41 No. 1, pp. 49-77.

Alghamdi, M.A. and Teahan, W.J. (2017), "A new thinning algorithm for Arabic script", *International Journal of Computer Science and Information Security*, Vol. 15 No. 1, pp. 204-211, available at: http://search.proquest.com/openview/5c29856709508552a5566d2504966d54/1?pq-origsite=gscholar&cbl=616671 (accessed 22 April 2017).

Alghamdi, M.A., Alkhazi, I.S. and Teahan, W.J. (2016), "Arabic OCR evaluation tool", *Proceedings of 7th International Conference on Computer Science and Information Technology (CSIT)*, *IEEE*, *Amman*, pp. 1-6.

Alginahi, Y.M. (2013), "A survey on Arabic character segmentation", *International Journal on Document Analysis and Recognition (IJDAR)*, Vol. 16 No. 2, pp. 105-126.

Al-Muhtaseb, H. and Qahwaji, R. (2011), "Arabic optical character recognition: recent trends and future directions", *Applied Signal and Image Processing: Multidisciplinary Advancements*, IGI Global, pp. 324-346.

Ben Moussa, S., Zahour, A., Benabdelhafid, A. and Alimi, A.M. (2010), "New features using fractal multi-dimensions for generalized Arabic font recognition", *Pattern Recognition Letters*, Vol. 31 No. 5, pp. 361-371.

Dahi, M., Semary, N.A. and Hadhoud, M.M. (2015), "Primitive printed Arabic optical character recognition using statistical features", *the IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, *IEEE*, pp. 567-571.

El-Mahallawy, M.S.M. (2008), "A large scale HMM-based Omni font-written OCR system for cursive scripts", PhD thesis, Faculty of Engineering, Cairo University Giza.

Hosseini, H.M.M. (1997), *Analysis and Recognition of Persian and Arabic Handwritten Characters*, Department of Electrical and Electronic Engineering, University of Adelaide, available at: https://books.google.co.uk/books?id=iJrcNwAACAAJ

Kanai, J., Rice, S., Nagy, G. and Nartker, T. (1993), "Performance metrics for printed document understanding systems", Proceedings of the International Conference on Document Analysis and Recognition (ICDAR-93), *IEEE*, *Tsukuba*, pp. 424-427, available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=395703

Kanungo, T., Marton, G.A. and Bulbul, O. (1999a), "OmniPage vs. Sakhr: paired model evaluation of two Arabic OCR products", *Electronic Imaging'99*, San Jose, CA, pp. 109-120.

Kanungo, T., Marton, G.A. and Bulbul, O. (1999b), "Performance evaluation of two Arabic OCR products", *Proceedings of SPIE*, pp. 76-83, available at: http://link.aip.org/link/?PSI/3584/76/1&Agg=doi

Khorsheed, M.S. (2002), "Off-line Arabic character recognition – a review", *Pattern Analysis & Applications*, Vol. 5 No. 1, pp. 31-45.

Levenshtein, V. (1966), "Binary codes capable of correcting deletions, insertions, and reversals", *Soviet Physics Doklady*, Vol. 10 No. 8, pp. 707-710.

Luqman, H., Mahmoud, S.A. and Awaida, S. (2014), "KAFD Arabic font database", *Pattern Recognition*, Vol. 47 No. 6, pp. 2231-2240.

Margner, V. and El Abed, H. (2009), "Arabic word and text recognition – current developments", *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, MEDAR Consortium.

Mihov, S., Schulz, K.U., Ringlstetter, C., Dojchinova, V. and Nakova, V. (2005), "A corpus for comparative evaluation of OCR software and postcorrection techniques", *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), IEEE*, pp. 162-166.

Parvez, M.T. and Mahmoud, S.A. (2013), "Offline Arabic handwritten text recognition: a survey", *ACM Computing Surveys*, Vol. 45 No. 2, pp. 1-23, available at: http://doi.acm.org/10.1145/2431211.2431222%5Cnhttp://dl.acm.org/ft_gateway.cfm?id=2431222&type=pdf

Sabbour, N. and Shafait, F. (2013), "A segmentation-free approach to Arabic and Urdu OCR", Proceeding of Document Recognition and Retrieval XX Conference, *SPIE*, Vol. 8658, pp. 1-12, available at: http://reviews.spiedigitallibrary.org/data/Conferences/SPIEP/72454/86580N.pdf (accessed 15 April 2017).

Saber, S., Ahmed, A., Elsisi, A. and Hadhoud, M. (2016), "Performance evaluation of Arabic optical character recognition engines for noisy inputs", in Gaber T., Hassanien, A., El-Bendary, N. and Dey, N. (Eds), *1st International Conference on Advanced Intelligent Systems and Informatics (AISI2015)*, November 28-30, 2015, Springer, *Beni Suef, Cham*, Vol. 407, pp. 449-459.

Sakhr Software OCR (Optical Character Recognition) (2017), available at: www.sakhr.com/index.php/en/solutions/ocr (accessed 15 April 2017).

Sami El-Dabi, S., Ramsis, R. and Kamel, A. (1990), "Arabic character recognition system: a statistical approach for recognizing cursive typewritten text", *Pattern Recognition*, Vol. 23 No. 5, pp. 485-495, available at: http://linkinghub.elsevier.com/retrieve/pii/003132039090069W (accessed 22 April 2017).

Slimane, F., Ingold, R., Kanoun, S., Alimi, A.M. and Hennebert, J. (2009), "A new Arabic printed text image database and evaluation protocols", *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), IEEE*, pp. 946-950.

Slimane, F., Kanoun, S., Hennebert, J., Alimi, A.M. and Ingold, R. (2013), "A study on font-family and font-size recognition applied to Arabic word images at ultra-low resolution", *Pattern Recognition Letters*, Vol. 34 No. 2, pp. 209-218, available at: http://dx.doi.org/10.1016/j.patrec.2012.09.012

Teahan, W.J., Inglis, S., Cleary, J.G. and Holmes, G. (1998), "Correcting English text using PPM models", *Proceedings of the Data Compression Conference (DCC'98)*, *IEEE*, pp. 289-298.

Further reading

Saber, S., Ahmed, A. and Hadhoud, M. (2014), "Robust metrics for evaluating Arabic OCR systems", *First International Image Processing, Applications and Systems Conference (IPAS), IEEE*, pp. 1-6.

Corresponding author
Mansoor Alghamdi can be contacted at: malghamdi@ut.edu.sa

(IJCSIS) International Journal of Computer Science and Information Security,
Vol. 15, No. 1, January 2017

A New Thinning Algorithm for Arabic Script

Mansoor A. Alghamdi
Department of Computer Science
Community Collage
Tabuk University
Tabuk, Saudi Arabia
malghamdi@ut.edu.sa

William J. Teahan
School of Computer Science
Bangor University
United Kingdom
w.j.teahan@bangor.ac.uk

***Abstract*—Thinning is one of the critical processes for different applications in image analysis, in particular for Optical Character Recognition (OCR) applications. The accuracy performance of OCR relies on the effectiveness of thinning algorithms. However, previously there has been little attention paid for proposing thinning algorithms for Arabic script. Also, there is a lack of quantitative performance measures of thinning techniques for Arabic script. Consequently, it is unclear which thinning algorithms are more appropriate for Arabic script. In this paper, a new thinning algorithm for Arabic script is proposed with several new performance metrics. An experiment is conducted to evaluate the proposed algorithm against two well established thinning algorithms with respect to the several proposed objective performance metrics. The experimental results show that the new algorithm has the best performance among the other two thinning algorithms.**

Keywords-thinning; skeletonization; Arabic; performance evaluation; Arabic OCR; thinning algorithm

I. INTRODUCTION

Optical Character Recognition (OCR) is a technique that converts a machine-printed or handwritten text image into an editable computer format. OCR is considered a challenging task in the field of pattern recognition. Many OCR approaches have been proposed for Latin and non-Latin scripts. However, Arabic OCR still poses great challenges because of Arabic script characteristics [1].

Generally, the process of developing Arabic OCR systems consists of three main stages: pre-processing, feature extraction and classification. One of the key and initial stages for developing Arabic OCR systems is the pre-prepossessing stage which is a combination of algorithms that are applied to the input pattern images for facilitating the subsequent phases of OCR development process [2].

Producing skeletons is a critical pre-processing operation for OCR in which extracting features from the skeleton of a character is essential [3]. The method for producing the skeleton of a pattern image is called thinning. Thinning “skeletonization” can be defined as the process of unpeeling as many pixels as possible without distorting the general shape of the character [4]. In other words, it involves operations that can be implemented in order to produce the skeleton of object images. Thinning techniques are classified into iterative approaches and non-iterative approaches [5]. The former can be either sequential methods, which perform by peeling the counter pixels individually, or parallel methods which perform simultaneously on all the counter pixels until obtaining a skeleton [6]. The latter utilize other techniques, such as distance transforms, to produce a skeleton without examining all pixels [6].

In general, thinning is usually applied in OCR systems as a method for reducing the amount of data of a pattern that needs to be considered for the next processing stage, thereby saving storage space for the structural information of the pattern [7], [8]. Furthermore, thinning algorithms have contributed to facilitate feature extraction of a pattern which is the core task in OCR to identify the pattern from another, since the relevant information of a pattern is not related to the thickness of the pattern [9], [10]. Therefore, it is claimed that the effectiveness of an OCR performance is heavily relying on how effective the thinning algorithm is [11]. The authors in [4], [12] agree that the main features of a desirable thinning algorithm are: ensuring the peeling is as thin as possible, connected and

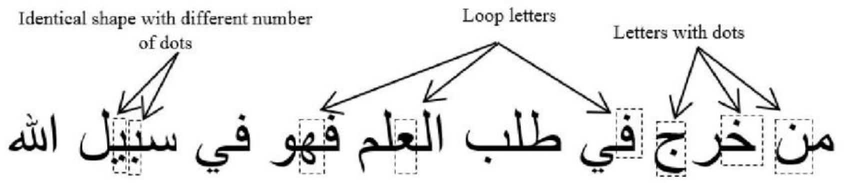

Identical shape with different number of dots
Loop letters
Letters with dots
من خرج في طلب العلم فهو في سبيل الله

Figure 1. Arabic script characteristics.

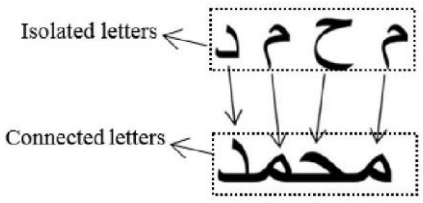


Figure 2. Arabic connectivity characteristic.

preserving the topology of a pattern.

A number of thinning algorithms have been proposed, such as in [13], [14], [15]; for a comprehensive survey refer to [16]. However, these algorithms have not been proposed specifically for Arabic script; rather they are proposed for general purposes. Additionally, direct adoption of such thinning algorithms, which are developed for other languages, may not be applicable to Arabic because of its characteristics [17] [18], [19]. Fig. 1 illustrates the characteristics of Arabic printed script that contribute to the challenge for thinning algorithms of Arabic script. Arabic script contains loop shaped letters, and identical characters that have different number of dots. Also, it is written cursively. In other words, isolated characters are connected to form a word, as shown in Fig. 2.

According to Cowell and Hussain [17], various problems have arisen when applying thinning algorithms to Arabic characters. One of the main obstacles with applying thinning algorithms to Arabic text is recognizing the number of dots for similar characters, where the number of dots can differentiate between them [2], [18]. It is believed that any deletion of character dots will result in misrepresenting these characters [19]. Moreover, another problem of thinning algorithms when considering Arabic script concerns preserving the connectedness of Arabic text. Some thinning approaches may fail in persevering the Arabic text connectivity which will lead to challenges in text recognition [18]. Owing to these reasons, it is claimed that thinning is responsible for many recognition errors in OCR systems [2], [18]. Therefore, thinning algorithms must be capable of both preserving dots and the connectedness of Arabic script.

As stated, there has been relatively few publications on developing thinning algorithms for Arabic [11], [8], [20]. For instance, [17], [21] introduce thinning algorithms for Arabic script. However, the proposed algorithms can only deal with isolated Arabic characters. Furthermore, one study by Ali [11] provides a thinning algorithm for Arabic handwritten script. Unfortunately, none of previous studies have considered the challenge of Arabic script discussed above, such as dots and connectivity preservation, when developing the thinning algorithm.

This paper will provide an efficient thinning algorithm for printed Arabic script. Then evaluation of thinning algorithms for Arabic will be discussed. An evaluation experiment is conducted on the proposed algorithm comparing it with a number of the most common thinning algorithms. Compared to the previous studies [22] and [23], this paper introduces several new objective performance metrics for thinning algorithms. Several performance metrics have been adapted to the evaluation using the evaluation dataset that contains a variety of Arabic font sizes and styles. In the last section, future works and conclusions are addressed.

II. THE PROPOSED METHOD

The proposed algorithm is based on an approach by Kocyigit [24]. However, this algorithm was only investigated for English characters. The method aims to produce a skeleton with one width pixel by detecting the neighborhood connectivity of any given pixel. Each pixel is directly connected with eight neighboring pixels; the pixel is considered as "encased" if only all the black neighbor pixels are interconnected with each other. In other words, if each black neighbour pixel of a considered pixel is connected vertically or diagonally to at least one other black pixel, the given pixel will achieve a state of encasement. For example, if the center pixel (P) is the pixel being processed, it is considered as "encased" in Fig. 3: (a), (b) and (c). In contrast, for (b), (c) and (d), the pixel (P) is not considered as "encased".

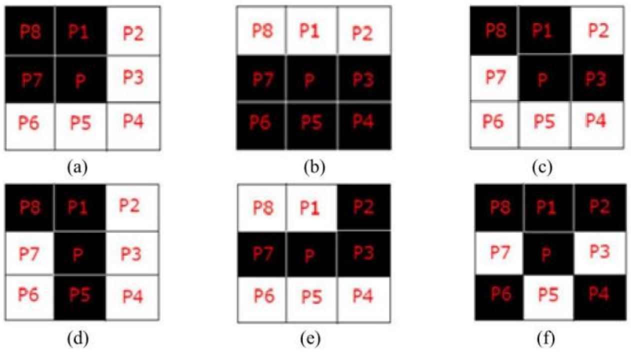


Figure 3. Examples of pixels (a), (b), (c) with an encasement and (d), (e), (f) without an encasement.

The proposed method utilizes a 3x3, 2D array of pixels which is obtained by thresholding the image. The descriptions of the rows and columns are equivalent to the pixel's coordinates and are set to either value "1" (black) or "0" (white) depending on the character. Deletion of a pixel will change the value of the pixel from 1 to 0.

Whenever any pixel is encased, the algorithm cleans up the defined pixel. The algorithm observes sequential and iterative principles in that whenever any pixel is removed, the entire algorithm restarts the iterative process, continuing until a state of consistency is achieved where no further pixel is deleted.

A flowchart of the proposed method is provided in Fig. 4. Initially, the algorithm finds a black pixel. Then, the black pixel is deleted if it has an encasement.

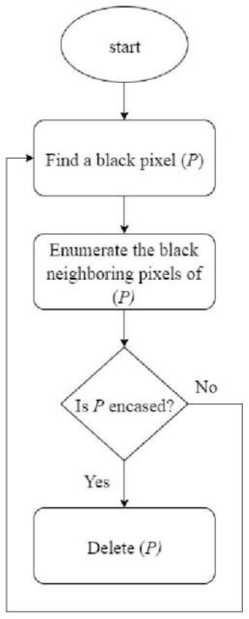


Figure 4. A flowchart of the proposed algorithm

III. Evaluating thinning algorithms for arabic script

In respect to performance evaluation of thinning algorithms, most of studies evaluate their proposed thinning algorithm in terms of computation execution time and compression ratio, such as [7], [25] and [26]. Furthermore, some researchers evaluate thinning algorithms by only illustrating the output images of the algorithms, regardless of application requirements, such as [27], [11], [5], [28]. However, from an OCR perspective, these metrics are not sufficient to provide us with insight into which algorithm performs better for OCR.

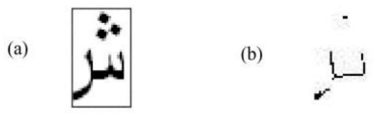


Figure 5. An example of the problem of the connectiviy metric. (a) An Arabic letter with four components and (b) A skeleton for (a) with four components.

The authors in [29] provide other general metrics for assessing thinning algorithms, such as the thinness metric, which assesses the level to which a pattern image is thinned, and the connectivity metric that measures the connectivity of the thinned pattern images.

As mentioned before, Arabic script uses a cursive writing style and therefore, the connectivity measure is considered as a critical metric in order to evaluate the connectivity of the output skeleton of an Arabic text image. For instance, the study in [22] utilizes the connectivity measure to assess the performance of different thinning algorithms for Arabic. The idea of this metric is to consider the number of connected components in the original image and in the output image of the thinning algorithm. Namely, if the number of components is equal in both images, then this is a sign that the thinning algorithm is preserving text connectivity. However, some thinning algorithms may remove some dots of Arabic characters and they may spilt the body of a character into several parts. Consequently, an issue will arise when adopting the connectivity metric to evaluate the connectivity of thinned Arabic text. In particular, when thinning algorithms remove the characters dots or break the connectivity, then a performance metric which is relying on the number of connected components, will not be reliable.

To illustrate this problem, Fig. 5 shows that the number of component of the original image (a) is four and the number of components in the thinned image (b) is also four. Thus, according to the connectivity metric, the algorithm will be assessed as preserving the text connectivity, where in fact it is not. This problem occurs owing to the removing of one dot of the letter and making a gap of the letter's shape. Therefore, it is difficult to obtain statistical evaluation results of connectivity preservation by relying on this measurement.

To overcome this problem, we propose new performance metrics for evaluating thinning algorithms for Arabic text in terms of connectivity and preservation of dots. Moreover, other further objective performance metrics for evaluating thinning algorithms will be discussed below.

A. Connectivity preservation metric

Typically, an image for a pattern is constructed from a set of vertices (nodes) and edges (links) some of which might be connected or disconnected [30]; see Fig. 6 for illustration. For thinning algorithms to preserve text connectivity, they must not divide a pattern in an image into incorrect pieces. A sophisticated thinning algorithm therefore must not delete edges of an image in order to maintain the connectivity characteristic of Arabic script.

Basically, there are two possible cases that will affect preserving the connectivity when producing skeletons for Arabic script. The first case is deletion of edges. For example, considering Figure 7 (a), it is clear that the word in the original image has fourteen (14) vertices and seven (7) edges. In contrast, in the thinned image in Fig. 7 (b), there are fourteen (14) vertices and six (6) edges. The lower number of edges is because of the deletion of one edge. The second case is insertion of a gap which results in the further insertion of edges and vertices. This can be seen in the case shown in Fig. 8 where the thinned image has sixteen (16) vertices and eight (8) edges, compared to the original image in Fig. 6 (a) which has fourteen (14) vertices and seven (7) edges.

Graph Edit Distance (GED) is a distortion measure on graphs that determines the differences between two pattern images [31]. The differences between the two images are obtained with a number of edit operations that are required to convert one image into another image. The authors here will utilize the concept of GED to assess the effectiveness of a thinning algorithm in preserving the connectivity of Arabic text.

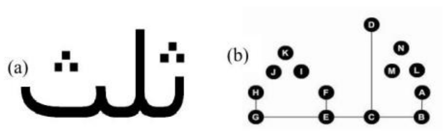


Figure 6. Using a graph to represent a text image: (a) a text image and (b) a graph for (a).

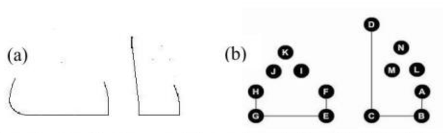


Figure 7. One thinning example for text image in Fig. 6(a). (a) Its skeleton and (b) a graph for (a).

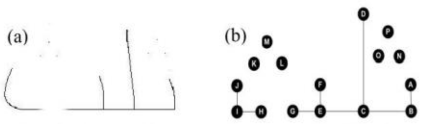


Figure 8. Another thinning example for text in Fig. 6(a). (a) Its skeleton and (b) a graph for (a).

The edit operations are: insertion of edges, insertion of vertices and deletion of edges. In order to measure GED, the minimum number of edit operations (edit distance) needed to convert the thinned image into the original image will be considered. Then, accuracy of thinning algorithms in preserving Arabic connectivity is determined by:

$$\frac{g-e}{g}\times 100 \qquad (1)$$

where g is the number of edges and vertices in the original image, and e is the edit distance. The cost of each edit operation will be defined as 1. For example, if a thinning algorithm inserted a gap, it will result in adding two vertices and delete one edge. Thus, the edit distance cost will be three, as three operations are required to transform the thinned image into the original – two deletions of vertices and one deletion of an edge, each having a cost of 1.

B. Dot preservation metric

In order for thinning algorithms to preserve dots of Arabic script, they must not remove isolated vertices which present the dots of Arabic text (see Fig. 9 for illustration). Thus, to statistically measure the effectiveness of thinning algorithms in preserving Arabic dots, the following equation is utilized:

$$\frac{v-e}{v}\times 100 \qquad (2)$$

where v is the number of isolated vertices in the original image, and e is the edit distance. The cost for each edit operation will be defined to be 1.

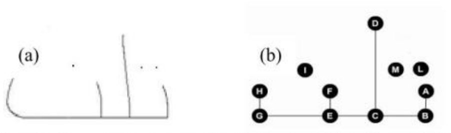


Figure 9. Example of dot perservation for thinning. (a) A skeleton and (b) a graph for (a).

C. Topology preservation metric

The topology preservation metric is the performance measurement utilized to assess the degree of thinning algorithm in preserving the visual information of the original image [12], [32], [33] [34] [35]. As reported by the researchers in [22], [18], producing spurious tails is considered a common problem of thinning algorithms for text images. Spurious tails can change the shape of a pattern, thereby affecting the accuracy of OCR output. Accordingly, it is essential to evaluate the effectiveness of thinning algorithms in terms of producing spurious tails.

The accuracy of thinning algorithms in preserving topology is determined by [12] :

$$1-\left[\frac{1}{2}-\frac{o}{c}\right] \quad (3)$$

where o is the number of object pixels in thinned image and c is the number of counter pixels in the thinned image. Note that in order to state that a thinning algorithm maintains visual information of the original image, the algorithm needs to have a preserving topology value close to 1.

D. Thinning rate (unit pixel width)

It is critical to evaluate the thinning algorithms in term of thinness of the skeleton, since one of the main principles of thinning algorithms is to ensure the peeling is as thin as possible. The thinness of the skeleton can be determined by computing the thinning rate as the following [23]:

$$1-\frac{tc}{tr} \quad (4)$$

where tc and tr refer to the number of triangles in the thinned image and in the original image, respectively. A thinning rate of 1 indicates that the skeleton is thinned to one-pixel wide, whereas a 0 value indicates that the skeleton is not thinned.

IV. Experimental results & discussion

We have used performance metrics described in the previous section for evaluating the proposed thinning algorithm. For comparison purposes, we also produce results for two other commonly used thinning algorithms for Arabic script, namely the Zhan-Suen (Algorithm 1) [36] thinning algorithm and the Hilditch algorithm (Algorithm 2) [37]. Note that all the three thinning algorithms were implemented in the Java programing language. The performance evaluation will provide us with insight into which algorithm performs better for Arabic script. In particular, a good thinning algorithm for Arabic should have a high performance accuracy in preserving connectivity and dots along with a value of preserving topology and thinning rate both close to 1.

Three datasets were used in this experiment. Specifically, the first dataset [38] is the printed Arabic word dataset, which consists of 50 word images, differing in fonts, sizes and styles. Also, two further sets of images were created for this evaluation experiment: images of Arabic characters and images of Arabic digits images.

In this experiment, all images from the three datasets were sent to each thinning algorithm to obtain thinned images. Then, the output images of each thinning algorithm were utilized to compute the quantities of each performance metric, corresponding to the original images for the datasets. Note that the performance metrics are implemented using the Matlab programming language. Samples of the evaluation experiment from the three datasets are visually illustrated in Table I. Table II presents the experimental data on the visual samples shown in Table I.

As discussed previously, connectivity preservation is one of the most essential features expected of thinning algorithms. From the data presented in Table I, it is apparent that the connectivity characteristic of Arabic script is almost maintained by the new algorithm. In particular, Algorithm 1 and Algorithm 2 failed in preserving the connectivity of sample 1 and sample 5, whereas the connectivity is preserved by the new algorithm in both samples.

There were major differences in the preservation of the dots of Arabic script between the three thinning algorithms. However, the new algorithm preserves the dots of Arabic script on all samples images that contain dots, as illustrated in Table I.

Furthermore, it is instructive to note that Algorithm 1 and Algorithm 2 were not able to preserve the loop shape of the sample 5 image, as shown in Table I and Table II, whereas it has been preserved by the new algorithm

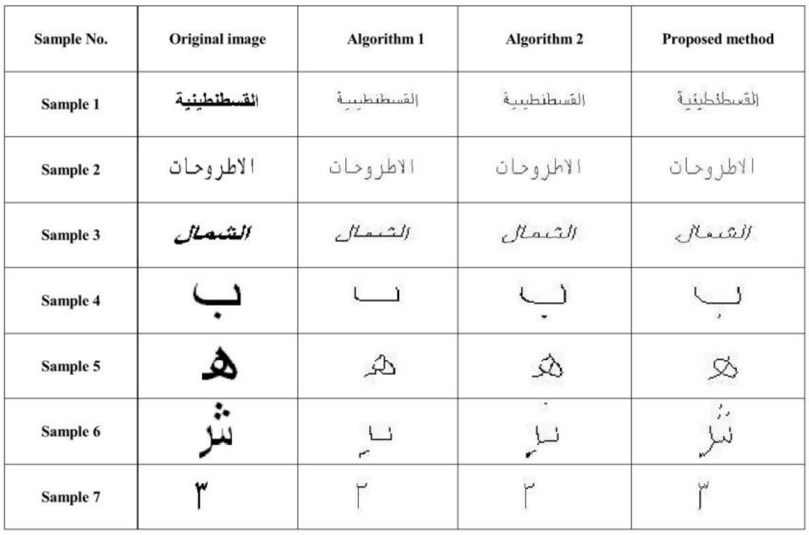
The average scores for each metric for the three thinning algorithms are presented in Table III. The first column compares the three thinning algorithms in terms of the connectivity preservation which shows that the new algorithm obtained the highest accuracy (94.6%) of connectivity preservation among the other two algorithms.

The second column of Table III compares the thinning algorithms in terms of preservation of the dots of Arabic. The results show that there are significant differences between the performance accuracy for dot preservation by Algorithm 1, Algorithm 2 and the new algorithm, 78.2%, 85.2% and 99.4% respectively. Thus, it can be stated that the new algorithm is the most effective at preserving the dots.

The third column of Table III lists the values for topology preservation. As mentioned previously, a value of topology preservation close to 1 indicates that the algorithm is successful in preserving the shape of the original images. The results in the third column of Table III show that the value of topology preservation is significantly better for the new algorithm (0.9599), compared to Algorithm 1 (0.9398) and Algorithm 2 (0.9411).

The results of the thinning rate analysis are presented in the fourth column in Table III. From this column, it clear that the thinning rate value achieved by the new algorithm (0.9887) is superior to the other two algorithms. This indicates that the skeletons produced by the new algorithm is thinner than skeletons produced by Algorithms 1 and 2.

TABLE I. VISUAL SAMPLES USED IN THE EVALUATION.



Sample No.	Original image	Algorithm 1	Algorithm 2	Proposed method
Sample 1	القسطنطينية	القسطنطينية	القسطنطينية	القسطنطينية
Sample 2	الاطروحات	الاطروحات	الاطروحات	الاطروحات
Sample 3	الشمال	الشمال	الشمال	الشمال
Sample 4	ب	ب	ب	ب
Sample 5	ه	ه	ه	ه
Sample 6	شر	شر	شر	شر
Sample 7	٣	٣	٣	٣

TABLE II. COMPARING THE THREE THINNING ALGORITHMS ON THE SAMPLES.

Image	***Connectivity Preservation***			***Dots Preservation***			***Topology Preservation***			***Thinning rate***		
	Algo1	***Algo2***	***Proposed Algorithm***	***Algo1***	***Algo2***	***Proposed Algorithm***	***Algo1***	***Algo2***	***Proposed Algorithm***	***Algo1***	***Algo2***	***Proposed Algorithm***
Sample 1	90.9 %	90.9 %	100 %	90 %	90 %	100 %	0.9955	0.9288	0.9511	0.9473	0.9259	0.9766
Sample 2	100 %	95.8 %	100 %	100 %	100 %	100 %	0.9751	0.9729	0.9875	0.8711	0.8813	0.9796
Sample 3	94.1 %	94.1 %	100 %	100 %	100 %	100 %	0.9630	0.9983	0.9244	0.9695	0.9390	1
Sample 4	100 %	100 %	100 %	0 %	100 %	100 %	0.9247	0.9513	0.9513	0.9716	0.8865	0.9929
Sample 5	83.3 %	83.3 %	100 %	N/A	NIA	N/A	0.8614	0.8614	0.9045	1	0.9752	0.9886
Sample 6	50 %	50 %	83.3 %	0 %	33.3 %	100 %	0.9695	0.9434	0.9782	0.9389	0.9061	0.9859
Sample 7	80 %	80 %	100 %	N/A	N/A	N/A	0.9426	0.9590	0.9918	1	0.9606	0.9859

TABLE III. SUMMARY OF RESULTS COMPARING THE THINNING ALGORITHMS.

Algorithms	Connectivity Preservation	Dot Preservation	Topology Preservation	Thinning Rate
Algorithm 1	84.4 %	78.2 %	0.9398	0.9552
Algorithm 2	83.7 %	85.2 %	0.9411	0.9266
Proposed Algorithm	94.6%	99.4 %	0.9599	0.9887

In summary, the evaluation experiment results show that the new algorithm is more effective at dealing with the challenges of thinning for Arabic script, namely, connectivity and dots preservation. Moreover, the experimental analysis indicates that the skeletons of Arabic text produced by the new algorithm are better than those produced by the two other algorithms in terms of topology preservation and thinning rate. Also, the evaluation experiment of thinning algorithms by utilizing the proposed performance metrics provides a more in-depth analysis of which algorithm will perform better for Arabic OCR systems.

V. CONCLUSION AND FUTURE WORKS

An efficient thinning algorithm for Arabic script is described. Furthermore, the authors describe several objective performance metrics for assessing statistically the effectiveness of thinning algorithms including two new metrics for assessing topology preservation and dots preservation. By utilizing these metrics, a more robust evaluation of the effectiveness of different thinning algorithms can be carried out. Consequently, these metric will simplify the choice of thinning algorithms for Arabic OCR developers.

An evaluation experiment is conducted to evaluate the performance of the new proposed algorithm against other two well established thinning algorithms. In all performance tests, the new algorithm obtains the best results.

For future work, it will be interesting to measure the impact of the new thinning algorithm on the performance accuracy of Arabic OCR systems.

REFERENCES

[1] M. A. Alghamdi, I. S. Alkhazi, and W. J. Teahan, "Arabic OCR evaluation tool," in *2016 7th International Conference on Computer Science and Information Technology (CSIT)*, 2016, pp. 1–6.

[2] B. Al-Badr and S. A. Mahmoud, "Survey and bibliography of Arabic optical text recognition," *Signal Processing*, vol. 41, no. 1, pp. 49–77, 1995.

[3] J. R. Parker, "Algorithms for Image Processing and Computer Vision," *Vasa*, p. 504, 2011.

[4] M. Cheriet, N. Kharma, C. Liu, and C. Suen, *Character recognition systems: a guide for students and practitioners*. John Wiley & Sons, 2007.

[5] W. Abu-Ain, S. N. H. S. Abdullah, B. Bataineh, T. Abu-Ain, and K. Omar, "Skeletonization Algorithm for Binary Images," *Procedia Technol.*, vol. 11, no. 1, pp. 704–709, 2013.

[6] K. Saeed, M. Tabędzki, M. Rybnik, and M. Adamski, "K3M: A universal algorithm for image skeletonization and a review of thinning techniques," *Int. J. Appl. Math. Comput. Sci.*, vol. 20, no. 2, pp. 317–335, 2010.

[7] N. J. Naccache and R. Shinghal, "SPTA: A Proposed Algorithm for Thinning Binary Patterns," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-14, no. 3, pp. 409–418, 1984.

[8] A. M. Al-Shatnawi, F. H. Al-Zawaideh, S. Al-Salameh, and K. Omar, "Offline Arabic Text Recognition – An Overview," *World Comput. Sci. Inf. Technol.*, vol. 1, no. 5, pp. 184–192, 2011.

[9] H. Devi, "Thinning: A Preprocessing Technique for an OCR System for the Brahmi Script," *Anc. Asia*, vol. 1, 2006.

[10] Y. Alginahi, "Preprocessing Techniques in Character Recognition," INTECH, pp. 1–20, 2010.

[11] M. A. Ali, "An efficient thinning algorithm for arabic ocr systems," *Signal Image Process. An Int. J.*, vol. 3, no. 3, pp. 31–38, 2012.

[12] H. Chatbri and K. Kameyama, "Using scale space filtering to make thinning algorithms robust against noise in sketch images," *Pattern Recognit. Lett.*, vol. 42, no. 1, pp. 1–10, 2014.

[13] Z. Guo and R. W. Hall, "Fast fully parallel thinning algorithms," *CVGIP Image Underst.*, vol. 55, no. 3, pp. 317–328, 1992.

[14] C. Hilditch, "Comparison of thinning algorithms on a parallel processor," *Image Vis. Comput.*, vol. 1, no. 3, pp. 115–132, 1983.

[15] P. S. P. Wang and Y. Y. Zhang, "A Fast and Flexible Thinning Algorithm," *IEEE Trans. Comput.*, vol. 38, no. 5, pp. 741–745, 1989.

[16] L. Lam and S. W. Lee, "Thinning methodologies—a comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 9, pp. 869–885, 1992.

[17] J. Cowell and F. Hussain, "Thinning Arabic characters for feature extraction," in *Proceedings of the International Conference on Information Visualisation*, 2001, vol. 2001-Janua, pp. 181–185.

[18] A. M. Al-shatnawi and K. Omar, "The Thinning Problem in Arabic Text Recognition - A Comprehensive Review," *Int. J. Comput. Appl.*, vol. 103, no. 3, pp. 35–42, 2014.

[19] H. M. M. Hosseini, *Analysis and Recognition of Persian and Arabic Handwritten Characters*. University of Adelaide, Department of Electrical and Electronic Engineering, 1997.

[20] H. Al-ani, N. Ban, and H. M. Abass, "Printed Arabic Character Recognition using Neural Network," Journal of Computing, vol. 5, no. 1, pp. 64–66, 2014.

[21] M. Altuwaijri and M. Bayoumi, "A new thinning algorithm for Arabic characters using self-organizing neural network," in *Circuits and Systems, 1995. ISCAS'95., 1995 IEEE International Symposium on*, 1995, vol. 3, pp. 1824–1827.

[22] A. M. Al-Shatnawi, B. M. Alfawwaz, K. Omar, and A. M. Zeki, "Skeleton extraction: Comparison of five methods on the Arabic IFN/ENIT database," in *Computer Science and Information Technology (CSIT), 2014 6th International Conference on*, 2014, pp. 50–59.

[23] P. TARÁBEK, "Performance measurements of thinning algorithms," *J. Information, Control Manag.*, vol. 6, no. 2, pp. 125–132, 2008.

[24] P. Kocyigit, "Agent Based Optical Character Recognition," MSc. thesis, Dept. CS. Bangor University, Bangor, 2012.

[25] A. K. J. Saudagar and H. V. Mohammed, "OpenCV Based Implementation of Zhang-Suen Thinning Algorithm Using Java for Arabic Text Recognition," in *Information Systems Design and Intelligent Applications*, Springer, 2016, pp. 265–271.

[26] Y. Y. Zhang and P. S. P. Wang, "A modified parallel thinning algorithm," *[1988 Proceedings] 9th Int. Conf. Pattern Recognit.*, 1988.

[27] M. Ali and K. Bin Jumari, "Skeletonization algorithm for an Arabic handwriting.," *WSEAS Trans. Comput.*, vol. 2, no. 3, pp. 662–667, 2003.

[28] W. Abu-Ain, S. N. H. Sheikh Abdullah, and K. Omar, "A simple iterative thinning algorithm for text and shape binary images," *J. Theor. Appl. Inf. Technol.*, vol. 63, no. 2, pp. 274–281, 2014.

[29] R. W. Zhou, C. Quek, and G. S. Ng, "A novel single-pass thinning algorithm and an effective set of performance criteria," *Pattern Recognit. Lett.*, vol. 16, no. 12, pp. 1267–1275, 1995.

[30] X. Gao, B. Xiao, D. Tao, and X. Li, "A survey of graph edit distance," *Pattern Anal. Appl.*, vol. 13, no. 1, pp. 113–129, 2010.

[31] M. Neuhaus and H. Bunke, "A Quadratic Programming Approach to the Graph Edit Distance Problem," *Graph-Based Represent. Pattern Recognit.*, pp. 92–102, 2007.

[32] B. K. Jang and R. T. Chin, "One-pass parallel thinning: analysis, properties, and quantitative evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 11, pp. 1129–1140, 1992.

[33] L. Huang, G. Wan, and C. Liu, "An improved parallel thinning algorithm," *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 2003-Janua, no. Icdar, pp. 780–783, 2003.

[34] G. Goyal and M. Dutta, "Design of Pixel Neighborhood Based Offline Handwritten Thinning Framework for Devnagri Numeral Script using Elman Neural Network," *Int. J. Comput. Sci. Inf. Secur.*, vol. 14, no. 7, p. 369, 2016.

[35] G. Goyal and M. Dutta, "Experimental Approach for Performance Analysis of Thinning Algorithms for Offline Handwritten Devnagri Numerals," *Indian J. Sci. Technol.*, vol. 9, no. 30, 2016.

[36] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Commun. ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[37] C. J. Hilitch, "Linear Skeletons From Square Cupboards," in *Machine Intelligence 4*, B. Meltzer and D. Michie, Eds. Edinburgh University Press, 1969, p. 403.

[38] H. Luqman, S. A. Mahmoud, and S. Awaida, "KAFD Arabic font database," *Pattern Recognit.*, vol. 47, no. 6, pp. 2231–2240, 2014.

AUTHORS PROFILE

Mansoor A. Alghamdi I am currently pursuing a Ph.D. in Computer Science at Bangor University, Bangor, UK. In 2011, at Otago University, Dunedin, New Zealand, I completed a Master of Applied Science in Software and Knowledge Engineering with distinguishing. In 2007, at Tabuk University, Tabuk, Saudi Arabia, I completed a B.Sc. Computer Science. I also was a Lecturer in the Department of Computer Science, Community College, Tabuk University, Tabuk, Saudi Arabia from 2012 to 2015.

William J. Teahan I am currently a Lecturer in the School of Computer Science at the University of Wales at Bangor. My work involves research into Artificial Intelligence and Intelligent Agents. Ongoing research has also specifically focused on applying text compression-based language models to natural language processing (NLP), text categorization and text mining. Before I came to Bangor, I was a research fellow with the Information Retrieval Group under Prof. David Harper at The Robert Gordon University in Aberdeen, Scotland from 1999-2000; an invited researcher in the Information Theory Dept. at Lund University in Sweden in 1999; and a Research Assistant in the Machine Learning and Digital Libraries Labs at the University of Waikato in New Zealand in 1998. At Waikato, I completed my Ph.D. in 1998 on applying text compression models to the problem of modelling English text.