AUTOMATIC SEGMENTATION FOR ARABIC CHARACTERS IN HANDWRITING DOCUMENTS

A. Lawgali, A. Bouridane, M. Angelova, Z. Ghassemlooy

School of Computing, Engineering and Information Sciences
Northumbria University, Newcastle upon Tyne, UK
ahmed.lawgali@northumbria.ac.uk

ABSTRACT

The cursive and ligature nature of the Arabic script make the segmentation of words into individual characters a difficult task. Despite attempts to apply methods for cursive Latin and other scripts to Arabic script, it is generally insufficient to segment the Arabic text. This paper proposes a new segmentation algorithm for the handwritten Arabic text and the main idea consists of segmenting the word into sub-words and then computing the baseline of each sub-word. Using the descenders of sub-words and the baseline, candidate points are then calculated using a vertical projection. The algorithm has been tested using 800 handwritten Arabic words taken from the IFN/ENIT database and a comparison made against some existing methods and promising results have been obtained.

Index Terms— Arabic character segmentation.

1. INTRODUCTION

Automatic off-line recognition of text, which is the ability of the computer to distinguish characters and words, can be divided into the recognition of printed and handwritten characters. Printed characters have one style and a size for any given font. However, handwritten characters have styles and sizes, which vary both for the same writer and between different writers. Many languages use Arabic characters such as Persian, Urdu and Jawi [5]. Little research has been carried out in the field of Arabic handwritten character recognition compared to research in Latin and Chinese [8] counterparts. The cursive (the way successive characters are connected together depending on their positions) nature and its peculiar ligatures (a prevalent glyph which replaces two or more characters) of the Arabic script makes the segmentation of words into individual characters a difficult task [1]. Despite attempts to apply methods for cursive Latin to Arabic, it is generally insufficient to segment Arabic text [8]. In addition, the ligature increases the difficulty of segmentation, which does not allow algorithms developed for other scripts to be applied to the Arabic script [1]. Several methods were presented by researchers based on recognition of the whole word without segmentation and others assumed that the characters are already segmented in order to avoid the segmentation process [5]. There are algorithms which split the words into characters [4]; however, most of the proposed segmentation algorithms currently do not solve the problem of overlapping characters in Arabic handwriting. The segmentation stage is the most difficult task and is the main source of errors in the recognition. Segmentation still represents a challenge in text recognition and needs to be improved [8]. Sari et. al. [7] introduced a method based on the morphological rule analysis of word to extract segmentation points. In [4] a technique for over-segmenting the word is proposed which uses knowledge of character shapes to reject extra segmentation points. Husam et. al. [2] presented a technique for segmenting a word into its primitives where a neural network is used for validating the segmentation points based on some features such as directions. In this paper, a new segmentation algorithm for Arabic characters in handwritten documents is proposed. The main idea is to detect the baseline for each subword in order to extract the candidate segmentation points from the skeleton of the sub-word. The organization of the paper is as follows. Section 2 discusses the challenges and motivation for the proposed segmentation technique. Section 3 describes in detail the proposed approach while Section 4 discusses results and their analysis. Section 5 concludes the paper including some future work.

2. CHALLENGES AND MOTIVATION

Arabic script is written from right to left and is composed of 28 characters with no capital or lower cases. Unlike Latin text, each character has two or four shapes where the shape of each character depends on its position in the word. 'Diacritics' play a significant role in Arabic characters. The shape of some characters is similar but the difference arises with the position and the number of diacritics such as (ب, ت, ث), which can take place either above or below the characters. Arabic handwriting is cursive meaning that characters of a word are connected through an imaginary horizontal line, known as baseline. Also, there are lines, which appear above and below the baseline, called ascenders and descenders as shown in Fig. 1 [5]. In addition, six characters do not connect to a subsequent character in a word as it causes a separation of the word into parts. These parts are called the sub-words. The spaces separate the words and short spaces separate sub-words. Two or more characters in the Arabic script can be combined vertically through different shapes. Overlap between neighboring characters is called the ligature. This may result in the second character appearing before the first one in some cases [5]. Ligature might occur when characters such as ح, خ, ج, م, ه appear after other particular characters. Fig. 2(a) shows an example of ligatures. In some cases, two characters may touch un-intentionally as shown in Fig. 2(b) for (و, ر) characters. Some characters may appear to be similar although they are different and it is difficult for the human eye to spot the difference [8]. The length and width of characters can be different (e.g., (ا, ب). Finally, the same character can be written differently in various forms; for example (ع, عـ) [8].

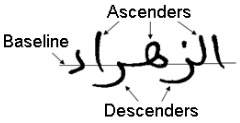


Fig. 1. Baseline, ascenders and descenders are shown in a Arabic word

(a) Overlapping in Arabic characters (b) Un-intentional touching in Arabic characters

Fig. 2. Some examples of Arabic handwriting

3. PROPOSED APPROACH

This work exploits the fact that the segmentation points, which occur at the end of a character and the beginning of the next, are usually located in the region surrounding the baseline. To achieve this, we propose to determine first the baseline for use to find the candidate segmentation points efficiently. Our approach consists of the following four tasks: data acquisition and pre-processing, word to sub-word segmentation, baseline detection, and extraction of segmentation points.

3.1. Data Acquisition and Pre-processing

Words are read from the IFN/ENIT database [6] containing 26459 handwritten Tunisian town/village names and written by 411 different writers consisting of about 212,000 characters and 115,000 pieces of Arabic words. Noise removal and binarization process have been carried out within the development of the database. The IFN/ENIT database is used for testing the algorithm. Pre-processing is applied to remove the details that have no discriminative power in the process of recognition (i.e. redundant). The diacritics and marks such as 'hamza' are removed from the words since they can affect the baseline extraction [3]. Since the diacritics can occur above or below the characters and their sizes may vary, care must be taken when they are removed especially that they may also be distinct or touching.

3.2. Words Sub-words Segmentation

A sub-word is a group of pixels joint together without any space. Therefore, the process of segmenting a word into sub-words depends on the space between the characters. This method assumes that there is no space in any single character. However, Fig. 3 shows that this assumption is not always correct in some cases such as Fig. 3(a). The order of the sub-words is important. The assumption is that it starts from right to left. Each sub word is stored as an image to deal with it as a word to detect its baseline and segmentation points. The small parts/blobs which appear after the segmentation process are considered as noise and are removed. The foregrounds (white areas) were also removed from sub-words where rectangular borders with two pixels around the sub-words are kept. Fig. 4 shows how the small parts appear as noise.

(a) Incorrect case (b) Correct case

Fig. 3. Correct and incorrect cases for character ط

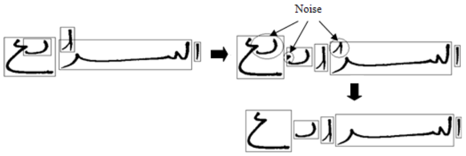


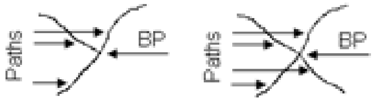
Fig. 4. Removing noise from the sub-word

3.3. Baseline Detection

Baseline (BL) extraction stage is an important step for character segmentation especially that most of the connection points between characters lie on it. Therefore, detecting the BL for sub-words rather than the words containing more than one sub-word is more efficient and will result in more successful outcomes [3]. This is due to the fact that the sub-words may not be located on the same line. The skeleton of a sub-word is extracted by reducing the width of the character to a single pixel. The horizontal and vertical projection of the skeleton of a sub-word is computed by:

$$HP(i) = \sum_{i} P(i,j) \ , \qquad VP(j) = \sum_{j} P(i,j) \qquad (1)$$

where HP(i) and VP(j) are the value of the horizontal and vertical projection respectively and P(i,j) is the pixel value of the binary image at the location (i,j). The highest peak of horizontal projection is calculated as the first estimation of the baseline (FEBL). Branch points (BP) of the word are often located around the BL. The BP having at least three path branches is shown in Fig. 5. BPs for skeletons of sub-word are also calculated. The average distance between BPs are calculated as the second estimation baseline (SEBL) of a sub-word. In the case where the sub-word does not have a BP, as in characters (ا, ب), the SEBL is not required. The BL is calculated from the average distance between the FEBL and the SEBL as shown in Fig. 6.



(a) Three paths (b) Four paths

Fig. 5. Branch point with paths

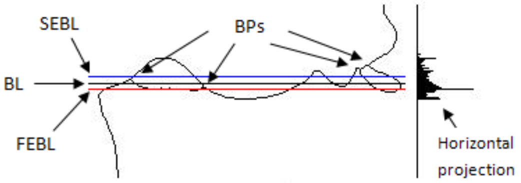
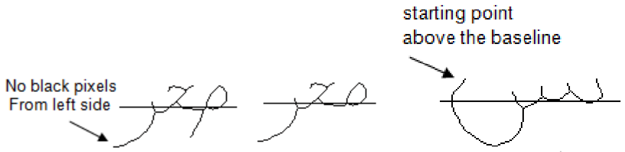


Fig. 6. Detect baseline process

3.4. Extraction of Segmentation Points

The segmentation points (SPs) occur between the end of a character and the beginning of the next one [4]. Most of the connection points lie in the region surrounding the BL. In some cases, a sub-word can be another character with no BP such as (ا, ب). In this case, SPs cannot be detected. Our approach uses a vertical projection to determine candidate points as SPs. The vertical projection for the skeleton of sub-word is computed by Equation 1. This method is successful with printed words [8]. Our approach has improved by deleting the descenders of sub-word that has a starting point below the BL. A starting point is defined as a pixel having one neighbor. It is worth noting that a descender will not be deleted if there are not black pixels on the left side of the starting point as shown in Fig. 7(a) or its starting point lies above the BL as depicted in Fig. 7(c). Fig. 7 illustrates the process of deleting the descenders. The algorithm starts by accessing image pixels from bottom up and from right to left in order to calculate the number of black pixels for each column using the vertical projection equation given previously. The number of black pixels for each column are then stored in a one-dimensional array. Therefore, any array element of value 1 corresponds to a thickness of one pixel. The method proceeds by using the FESP if the pixel lies in an area close to the BL and its length is of at least three pixels such as points P1, P2, P3 and P5 in Fig. 8. If the pixel is far from the BL threshold level, it is ignored as shown by point P6. If the length of the character is large, the SPs should be near the end of the left side of the stroke since there could be diacritics lying above or below the character. In some cases, there exists no thickness of one pixel in a sub-word such as (ا), therefore a SP in this sub-word does not apply. For each SP, the algorithm tests the left side of each SP to determine the number of branches available. If there is not a BP such as P4 in Fig. 8 this SP is removed and ignored except if the last character is Alif (ا) such as P2, otherwise the SP is accepted. SPs are applied to the original word without diacritics to extract the shapes of characters as outlined in Fig. 9.


starting point above the baseline
No black pixels From left side

(a) Before delete the descender (b) After delete the descender (c) No descender deleted

Fig. 7. Deleting the descenders

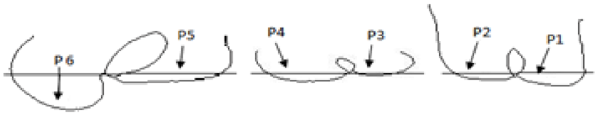


Fig. 8. The first estimation of segmentation point

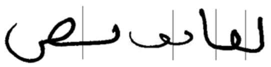


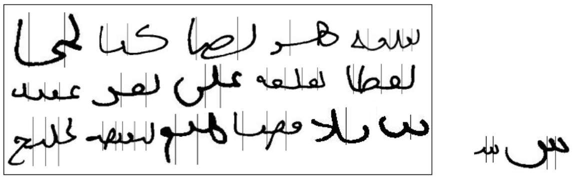
Fig. 9. Segmentation points on the original word

4. RESULTS AND ANALYSIS

Experiments have been carried out using 500 images containing 824 Arabic handwritten words taken from the IFN/ENIT database and 3964 characters. The choice of the words have been selected carefully to cover all shapes of the Arabic characters. The correct SPs were evaluated manually by observing the result of the algorithm and cross checked by two independent Arabic speaking people. The results obtained show that 87.9% of the SPs were extracted correctly. Characters that have overlapping segment as one (segment) and a further investigation of the overlapping problem of the characters will be carried out as part of a future work. Table 1 shows our results compared with previous works. Results of [7] have tested their algorithm on 100 words and achieved 86% accuracy. Their algorithm was based on morphological rules analysis of word to extract the segmentation point. Husam et. al. [2] used a technique of over-segment based on the neural network for validating segmentation points based on some features such as directions. They have achieved 82.98% character accuracy. Although some handwritten words are difficult to read and most/all existing methods fail to efficiently segment them, our algorithm has shown that it is capable to achieve accurate segmentation as demonstrated by the results. For example, all characters have been segmented correctly to one character except the characters س or ش which have three strokes segmented into three segments based on strokes. These segments can be combined as one character in the classification stage. Fig. 10 shows some results obtained using our algorithm. The vertical lines explain the positions of the SPs. Fig. 10(a) shows some results correctly segmented and Fig. 10(b) depicts segmentation of the character س into three segments.

Authors	Experiment data	Accuracy	Method
Sari et al. [7]	Local database (100 words)	86%	Based on the morphological analysis
Husam et al. [2]	Local database (500 words)	82.98%	Over-segment based on the ANN
Our algorithm	IFN/ENIT database (800 words)	87.9%	Based on the extracting BL

Table 1. Comparison our results with previous works



(a) Some correct results

(b) segmenting character س

Fig. 10. Some results of the algorithm

5. CONCLUSION AND FUTURE WORK

This paper has presented a new segmentation algorithm of handwritten Arabic words. The algorithm starts with segmenting the word into sub-words and then the baseline of each sub-word is computed. The descenders of sub-words which have a starting point below the baseline are then deleted. The vertical projection is used to find the candidate points for the segmentation. Segmenting character to a small parts could be expensive due to the fact that each part will need to extract features and classify them. However, our algorithm has been able to segment all characters correctly to one character except the characters س or ش which have three strokes segmented into three segments based on strokes. The algorithm has been tested using 800 handwritten Arabic words taken from IFN/ENIT database and promising results have been obtained. As future work, the segmentation algorithm will be improved by further investigating the more complex problem of overlapping characters.

6. REFERENCES

[1] A. A. Aburas and M. E. Gumah. Arabic handwriting recognition: Challenges and solutions. In *Intern. Symposium on Information Technology, ITSim.*, volume 2, pages 1 –6, 2008.

[2] H. A. Al-Hamad and R. Abu Zitar. Development of an efficient neural-based segmentation technique for arabic handwriting recognition. *Pattern Recogn.*, 43(8):2773–2798, 2010.

[3] A. AL-Shatnawi and K. Omar. A comparative study between methods of arabic baseline detection. In *Intern. Conference on Electrical Engineering and Informatics, ICEEI.*, volume 01, pages 73 –77, 2009.

[4] L. Lorigo and V. Govindaraju. Segmentation and pre-recognition of arabic handwriting. In *Proc. 8th Intern. Conference on Document Analysis and Recognition.*, pages 605 – 609 Vol. 2, 2005.

[5] L. M. Lorigo and V. Govindaraju. Offline arabic handwriting recognition: a survey. *IEEE Trans. on Pattern. Anal. and Mach. Intelligence,*, 28(5):712 –724, 2006.

[6] M. Pechwitz, S. S. Maddouri, V. Mrgner, N. Ellouze, and H. Amiri. Ifn/enit - database of handwritten arabic words. In *In Proc. of CIFED*, pages 129–136, 2002.

[7] T. Sari, L. Souici, and M. Sellami. Off-line handwritten arabic character segmentation algorithm: Acsa. In *Proc. 8th Intern. Workshop on Frontiers in Handwriting Recogn, 2002.*, pages 452 – 457, 2002.

[8] A. M. Zeki. The segmentation problem in arabic character recognition the state of the art. In *First Intern. Conference on Information and Communication Technologies, ICICT.*, pages 11 – 26, 2005.