Automatic Recognition of Printed Arabic Text Using Neural Network Classifier

Adnan Amin and Mandana Kavianifar
School of Computer Science & Engineering
University of New South Wales
2052 Sydney, Australia

Abstract The main theme of this paper is the automatic recognition of Arabic printed text using an artificial neural networks in addition to conventional techniques. This approach has a number of advantages: it combines rule-based (structural) and classification tests; and feature extraction is inexpensive and execution time is independent of character font and size. The technique can be divided into three major steps: The first step is pre-processing in which the original image is transformed into a binary image utilizing a 300 dpi scanner and then forming the connected component. Second, global features of the input Arabic word are then extracted such as number subwords, number of peaks within the subword, number and position of the complementary character, etc.. Finally, an artificial neural networks is used for character classification. The algorithm was implemented on a powerful MS-DOS based microcomputer and written in C.

1. Introduction

Character recognition systems can contribute tremendously to the advancement of the automation process and can improve the interaction between man and machine in many applications, including office automation, check verification and a large variety of banking, business and data entry applications.

Many papers have been concerned with the recognition of Latin, Chinese and Japanese characters. However, although almost a third of a billion people worldwide, in several different languages, use Arabic characters for writing, little research progress, in both on-line [1–6], and off-line [7–15], has been achieved towards the automatic recognition of Arabic characters. This is a result of the lack of adequate support in term of funding, and other utilities such as Arabic text database, dictionaries, etc.. and of course of the cursive nature of its writing rules.

Arabic writing is similar to English in that it uses letters (which consist of 29 basic letters), numerals, punctuation marks, as well as spaces and special symbols. It differs from English, however, in its representation of vowels since Arabic utilizes various diacritical markings. The presence and absence of vowel diacritics indicates different meanings in what would otherwise be the same word. For example, مدرسة is the Arabic word for both "school" and "teacher". If the word is isolated, diacritics are essential to distinguish between the two possible meanings. If it occurs in a sentence, contextual information inherent in the sentence can be used to infer the appropriate meaning. In this chapter, the issue of vowel diacritics is not treated, since it is more common for

Arabic writing not to employ these diacritics. Diacritics are only found in old manuscripts or in very confined areas.

The Arabic alphabet is represented numerically by a standard communication interchange code approved by the Arab Standard and Metrology Organization (ASMO). Similar to the American Standard Code for Information Interchange (ASCII), each character in the ASMO code is represented by one byte. An English letter has two possible shapes, capital and small. The ASCII code provides separate representations for both of these shapes, whereas an Arabic letter has only one representation in the ASMO table. This is not to say, however, that the Arabic letter has only one shape. On the contrary, an Arabic letter might have up to four different shapes, depending on its relative position in the text. For instance, the letter (ع A'in) has four different shapes: at the beginning of the word (preceded by a space), in the middle of the word (no space around it), at the end of the word (followed by a space), and in isolation (preceded by an unconnected letter and followed by a space).

In addition, different Arabic characters may have exactly the same shape, and are distinguished from each other only by the addition of a complementary character. Complementary characters (A portion of a character that is needed to complement an Arabic character). These are normally a dot, a group of dots or a zigzag (hamza). This may appear on, above, or below the base line are positioned differently, for instance, above, below or within the confines of the character. It is worth noting that any erosion or deletion of these complementary characters results in a misrepresentation of the character. Hence, any thinning algorithm needs to efficiently deal with these dots so as not to change the identity of the character.

Arabic writing is cursive and is such that words are separated by spaces. However, a word can be divided into smaller units called subwords (A portion of a word including one or more connected characters). Some Arabic characters are not connectable with the succeeding character. Therefore, if one of these characters exists in a word, it divides that word into two subwords. These characters appear only at the tail of a subword, and the succeeding character forms the head of the next subword. Figure 1 shows three Arabic words with one, two, and three subwords. The first word consists of one subword which has three letters; the second has two subwords with two and one letter, respectively. The last word contains three subwords, each consisting of only one letter.

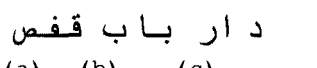

Fig. 1. Arabic words with constituent subwords.

2. Digitization and Preprocessing

Extensive work has been conducted on the binarization proces [16, 17]. The algorithm employed in this work is similar to the method appearing in reference [18]. A 300 dpi scanner is used to digitize the image in this phase and the output is a standard binary formatted image (PBM format).

2.1. Detecting a Loop

One common phenomenon that occurs in both printed and handwritten Arabic text is blotting. For a human, the blobs can be easily recognized as loop from the context. However, these loops are difficult to deal with and cause problems in automatic recognition system.

The basic idea of detecting a loop in a binary image is to move a window through the image from right to left, and top to bottom. If certain conditions are satisfied, then the pixels in the center of the window are whitened. The size of the window is determined by the width of the baseline of the word. The algorithm for detecting a loop can be summarized as follows:

- *Locate the baseline of the image*: A horizontal projection for the image can be performed. The rows which have the maximum number of pixels are taken as part of the baseline of the word. Any other adjacent lines with at least 85% of the pixels of these rows are also taken as part of the baseline. The horizontal projection of the Arabic word is shown in Figure 2 (b).
- *Scan the binary image:* A window of size 2H x 2H is used to scan the image sequentially from right to left and top to bottom. The value of the integer H is determined by the width of the baseline of the word. Furthermore, the window is even; hence, the center of the window consists of four pixels. These pixels are whitened if all of the following conditions are satisfied:
  - (a) All of the white pixels within the window are located on the first two outermost layers of the window;
  - (b) No group of the white pixels is surrounded by black pixels within the window, i.e., no hole exists in the 2H x 2H window;
  - (c) The number of the black pixels within the window are at least equal to 2H -2 pixels.

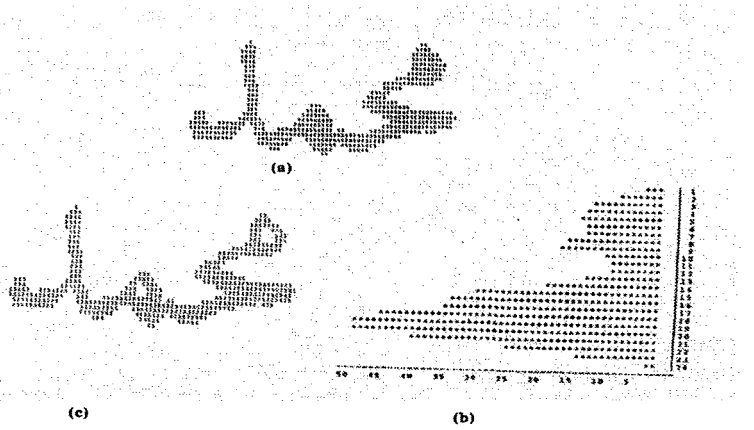

Fig. 2. (a) The input image (b) Horizontal projection of the input word (c) Result of the algorithm.

2.2 Connected components construction

The next stage of preprocessing involves finding the connected components (cc's) for the Arabic words. Connected components are rectangular boxes bounding together regions of connected black pixels. connected black pixels. The objective of the connected component stage is to form rectangles around distinct component on the page, whether they be characters or images. These bounding rectangles then form the skeleton for all future analysis on the page.

The algorithm used to obtain the connected component is a simple iterative procedure which compares successive scanlines of an image to determine whether black pixels in any pair of scanlines are connected together. Bounding rectangles are extended to enclose any grouping of connected black pixels between successive scanlines . Figure 3 demonstrates this procedure

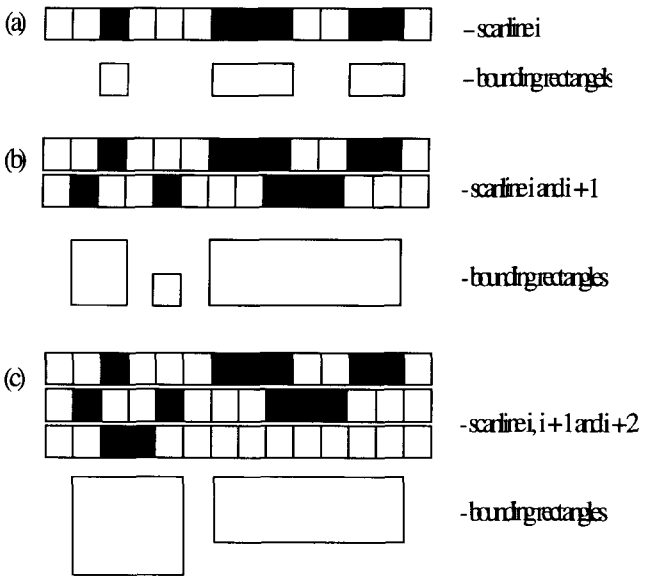

Fig. 3. The process of building connected components from image scanlines.

Each scanline in Figure 3 is 14 pixels in width (note that a pixel is represented by a small rectangular block), the bounding rectangles in Figure 3 (a) just enclose the block pixel of that scanlines, but for each successive scanline the bounding boxes increase to include the block pixels connected to the previous scanlines. Figure 3 (c) also points out that a bounding box stops growing in size only when there are no more black pixels on the current scanline joined onto black pixels of the previous scanline.

3. Global Word Feature

Previous studies typically focuses on the notion of characters in order to recognize Arabic words [15]. Two approaches are generally taken. The first one involves segmenting the input word into individual characters [7, 10, 13] which are then sent to a character recognizer, while the second approach involves scanning the whole word and extracting features to hypothesize the presence of certain character classes. The use of Hidden Markov Models (HMM) [19] falls into this scheme.

In this project, we have used the second approach to exact the proper characteristic of Arabic characters. At the moment we extract seven types of global features: number of subwords, number of peaks of each subwords, number of loops of each peak, number and position of complementary characters, the height and width of each peak. Feature extraction algorithm can be summarized into the following steps:

Step 1: Loop detection: Loops are detected simply as being the inner contours (as shown in Figure 4) obtained by running the contour tracing algorithm. The tracing algorithm traces outer contour of an object and then it traces the inner contours of the object.

Fig. 4. The inner and outer contour of an Arabic word.

Step 2: Determine the number of peaks within the subword: In all printed Arabic characters, the width at a connection point is much less than the width of the beginning character. Therefore, the baseline is a medium line in the Arabic word in which all the connections between the successive characters take place. If a vertical projection of bi-level pixels is performed on the word (Eq.1),

$$v(j) = \sum_{i} w(i, j) \tag{1}$$

where $w(i, j)$ is either zero or one and i, j index the rows and columns, respectively, the peak point will have a sum greater than the average value (AV) (Eq. 2)

$$AV = (1 / Nc) \sum_{j=1}^{Nc} Xj \tag{2}$$

where Nc is the number of columns and X_j is the number of black pixels of the j^{th} column.

Figure 5, illustrates the segmentation of an Arabic word into peaks.

Step 3: Smooth the histogram by using the averaging scheme where each point in the histogram is replaced by the average of itself and the two points on either side of it .

$$X_i = (X_{i-1} + X_i + X_{i-2}) / 3$$

Step 4: Determine the number and position of the complementary character with the peak

Step5: Compute the height and width of each peak whether large or small.

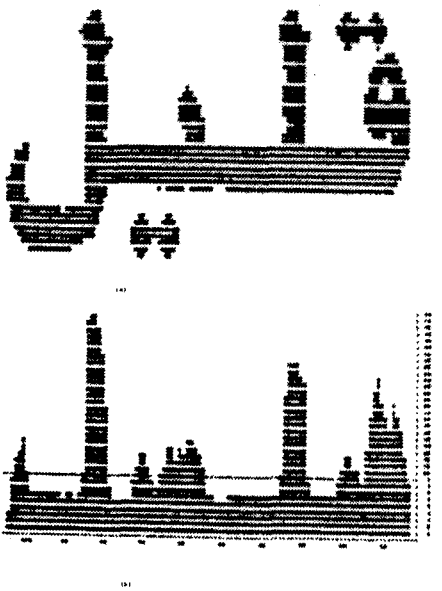

Fig. 5. An example of segmentation of the word into peaks (a) Arabic word (b) histogram after smoothing

4. Character Classification

Having extracted the primitives, it is now required to store them in some form and associate them with a word. Each pattern should uniquely identify a word and each word may be represented by several distinct patterns. These patterns along with the words they identify are required to be fed to a neural network. The neural network is then trained to identify each word. After this, if the pattern for an unknown word is presented to the neural network, it would be able to classify the word by generalizing from the patterns on which it has been trained.

The feature extracted are classified using a feedforward neural network trained by back-propagation [20]. The study uses three layer artificial neural network with one output unit for each possible word output. Every word will be classified in terms of

features such as: number of subwords (up to five), number of peaks of each subwords (up to seven), number of loops, number and position of complementary characters within the peak (up to three), height and width of each peak (Small or Large).

Figure 6 depicts the complete representation to the Arabic word shown in Figure 5. The word contains only one subword and four peaks.

The overall design of the input layer uses a total of 270 neurons. When the network is trained, the output layer, in response to a familiar input pattern or one which resembles a familiar pattern, activates the neurons corresponding to this word classification.

Word: 1 0 0 0 0 (contains only one subword)

Subword: 1 1 1 1 0 0 (subword contains four peaks)

1st peak : 1 1 0 1 (loop, complementary characters, small height, and large width)

2nd Peak: 0 0 1 1 (no loop, no complementary characters, large height, and large width).

3rd Peak : 0 1 0 1 (no loop, complementary characters, small height, and large width).

4th Peak : 0 0 1 1 (no loop, no complementary characters, Large height, and large width)

Fig. 6. The complete representation of word shown in Figure 8 for the neural network input layer.

5. Experimental Results and conclusion

Despite the tremendous research effort in character recognition, the recognition of Arabic characters is still in its infancy.

In this paper, we first discuss preprocessing step whose aim to reduce the noise, recover lost loop due to blotting and construct the connected component. A new method for recognizing printed Arabic text using a global approach was presented to avoid the difficulty of segmentation stage. The computational theory is based on the model of fast reader. Finally, a three layer artificial neural network was used for Arabic words classification.

The algorithm outlined in the previous sections was implemented as C programs on a 486 based PC connected to a 300 dpi scanner. At the moment the system was tested with a set of Arabic words (few hundred words). The average number of characters in these subwords is 2.87. The subwords were written in different fonts and the recognition rate obtained was 98%. More extensive testing and results will be presented in the final paper.

References

1. A. Amin, A. Kaced, J. P. Haton and R. Mohr, Handwritten Arabic characters recognition by the IRAC system, *5th Int. Conf. on Pattern Recognition,* Miami, USA, 1980, 729–731.
2. A. Amin, Machine recognition of handwritten Arabic word by the IRAC II system, *6th Int. Conf. on Pattern Recognition,* Munich, 1982, 34–36.
3. A. Amin, IRAC: Recognition and understanding systems, *Applied Arabic Linguistic and Signal and Information Processing*, ed. R. Descout, Hemisphere, New York, 1987, 159–170.
4. M. S. El-Wakil, On-line Recognition of Handwritten Isolated Arabic characters, Pattern Recognition, **22(2)**, 1989, 97–105.
5. T. S. El-Sheikh and S. G. El-Taweel, Real-time Arabic handwritten character recognition, *Pattern Recogn.* **23**, 12 (1990) 1323–1332.
6. S. Al-Emami and M. Usher, On-line recognition of handwritten Arabic characters, *IEEE Trans. Pattern Anal Machine Intell.* **PAMI-12** (1990) 704–710.
7. A. Amin and G. Masini, Machine recognition of muti-fonts printed Arabic texts, *8th Int. Conf. on Pattern Recognition,* Paris, 1986, 392–395.
8. A. Amin and J. F. Mari, Machine recognition and correction of printed Arabic text, *IEEE Trans. Man Cybern.* **9**, 1 (1989) 1300–1306.
9. A. Amin and S. Al-Fedaghi, Machine recognition of printed Arabic text utilising a natural language morphology, *Int. J. of Man-Machine Studies* **35**, 6 (1991) 769–788.
10. H. Almuallim and S. Yamaguchi, A method of recognition of Arabic cursive handwriting, *IEEE, Trans. Pattern Anal. and Machine Intell.* **PAMI-9** (1987) 715–722.
11. T. El-Sheikh and R. Guindi, Computer recognition of Arabic cursive script, *Pattern Recogn.* **21**, 4 (1988) 293–302.
12. F. El-Khaly and M. Sid-Ahmed, Machine recognition of optically captured machine printed Arabic text, *Pattern Recog.* **23**, 11 (1990) 1207–1214.
13. A. Amin and H. B. Al-Sadoun, A new structural technique for recognizing printed Arabic text, *Int. J. of Pattern Recognition and Artif. Intell.* **9**, 1 (1995) 101–125.
14. B. Al-Badr and S. Mahmoud, Survey and bibliography of Arabic optical text recognition, Signal Processing 41 , 49-77,1995.
15. A. Amin, Arabic Character Recognition. Handbook of Character Recognition and Document Image Analysis edited by H Bunke and P S P Wang, 1996.
16. N. Otsu, A threshold selection method from gray level histogram, *IEEE Trans. Syst. Man Cybernet.* **SMC-9 (1)** (1979), 62 –66.
17. J. S. Weszka and A. Rosenfeld, Histogram modification from threshold selection, *IEEE Trans. Syst. Man Cybernet.* **SMC-9 (1)** (1979), 38 –52.
18. A. Amin and H. Al-Sadoun, Handprinted Arabic character recognition system using an artificial neural network, *Pattern Recognition* **29 (4)** 1996, 663–675.
19. R. Schwartz, C. LaPre, J. Makhoul, C. Raphael, and Y. Zhao, Language independent OCR using a continuous speech recognition system, International Conf. On Pattern Recognition, 99-103, (1996).
20. D. E. Rumelhart, G. E. Hinton and R. J. Williams, Parallel Distributed Processing, Vol. 1, Bradford Books, MIT Press, Massachussetts (1986).