Pattern Analysis & Applications (2002)5:31–45

Off-Line Arabic Character Recognition – A Review

M. S. Khorsheed

Computer Laboratory, University of Cambridge, Cambridge, UK

Abstract: Off-line recognition requires transferring the text under consideration into an image file. This represents the only available solution to bring the printed materials to the electronic media. However, the transferring process causes the system to lose the temporal information of that text. Other complexities that an off-line recognition system has to deal with are the lower resolution of the document and the poor binarisation, which can contribute to readability when essential features of the characters are deleted or obscured. Recognising Arabic script presents two additional challenges: orthography is cursive and letter shape is context sensitive. Certain character combinations form new ligature shapes, which are often font-dependent. Some ligatures involve vertical stacking of characters. Since not all letters connect, word boundary location becomes an interesting problem, as spacing may separate not only words, but also certain characters within a word. Various techniques have been implemented to achieve high recognition rates. These techniques have tackled different aspects of the recognition system. This review is organised into five major sections, covering a general overview, Arabic writing characteristics, Arabic text recognition system, Arabic OCR software and conclusions.

Keywords: Arabic OCR; Feature extraction; Fourier Transform; Hidden Markov Models; Horizontal projection; Hough Transform; Neural Networks; Off-line recognition; Preprocessing segmentation; Vertical projection

1. INTRODUCTION

Among the branches of pattern recognition is the automatic reading of a text, namely, text recognition. The objective is to imitate the human ability to read printed text with human accuracy, but at a higher speed. A useful target performance is 5 characters/second with a 99.9% recognition rate, with all errors being rejections [1].

There are three factors pushing towards text recognition. The first two are the easy use of electronic media and its growth at the expense of conventional media. The third is the necessity of converting the data from the conventional media into the new electronic media. This factor motivates the vast range of applications of off-line text recognition, which includes automatic mail routing [2,3], machine processing of forms [4], bank cheques [5,6] printed newspapers [7] and signature verification [8].

Most optical character recognition methods assume that individual characters can be isolated, and such techniques, although successful when presented with Latin typewritten or typeset text, cannot be applied reliably to cursive script, such as Arabic, where the shape of the character is context sensitive. This feature, besides other characteristics of the Arabic language, has obliged researchers to examine some obstacles which have only recently been addressed by researchers of other languages. These obstacles have played an important role in delaying character recognition systems for the Arabic language compared to other languages such as Latin and Chinese. Additional factors involve: the early start of these systems for Latin (1940s) and Chinese (1960s); the absence of scientific journals and conferences specialised in the field, which caused a lack of communication between researchers; and the absence of support utilities such as a language corpus and electronic dictionaries.

Recently, a number of papers which analyse the work done on Arabic character/word recognition have appeared. Some of these papers deal with both on- and off-line recognition [9–12]. Others focus only on off-line recognition [13–16]. This survey presents an up-to-date review of the work done in the field of off-line Arabic recognition. This paper differs from previous work in that results of different techniques presented here are illustrated by implementing them on Arabic word/text samples. These samples were extracted from Arabic manuscripts that can be found in the University Library (UL) of Cambridge.

Received: 24 July 2000
Received in revised form: 13 February 2001
Accepted: 9 March 2001

2. ARABIC WRITING CHARACTERISTICS

The Arabic language has a very rich vocabulary. More than 200 million people speak Arabic as their native language, and over 1 billion people use the Arabic language in several religion-related activities. The alphabet set used to write this language is the Arabic alphabet, (see Table 1). There are also a number of languages that use the Arabic alphabet, such as Persian [17], Kurdi [18] and Jawi [19].

Arabic script is written from right to left. As opposed to starting from the left-most position of the page as for Latin, Arabic script starts from the right-most position of the page towards the left in a cursive way, even in machine-printed form 'الكتابة العربية متصلة'.

Table 1. The Arabic alphabet set. Each character may have up to four different shapes. The transliterated form of each character is illustrated in the right column

Character	Isolated	Initial	Middle	End	Transliteration
Alif	أ	أ	ـا	ـا	*a*
Ba'	ب	بـ	ـبـ	ـب	*b*
Ta'	ت	تـ	ـتـ	ـت	*t*
Tha'	ث	ثـ	ـثـ	ـث	*t̲*
Jeem	ج	جـ	ـجـ	ـج	*ǧ*
Ḥa'	ح	حـ	ـحـ	ـح	*ḥ*
Kha'	خ	خـ	ـخـ	ـخ	*ḫ*
Dal	د	د	ـد	ـد	*d*
Thal	ذ	ذ	ـذ	ـذ	*d̲*
Ra'	ر	ر	ـر	ـر	*r*
Zy	ز	ز	ـز	ـز	*z*
Seen	س	سـ	ـسـ	ـس	*s*
Sheen	ش	شـ	ـشـ	ـش	*š*
Sad	ص	صـ	ـصـ	ـص	*ṣ*
Dhad	ض	ضـ	ـضـ	ـض	*ḍ*
T'ah	ط	طـ	ـطـ	ـط	*ṭ*
The'ah	ظ	ظـ	ـظـ	ـظ	*ẓ*
Ain	ع	عـ	ـعـ	ـع	ʿ
Gain	غ	غـ	ـغـ	ـغ	*ġ*
Fa	ف	فـ	ـفـ	ـف	*f*
Qaf	ق	قـ	ـقـ	ـق	*q*
Kaf	ك	كـ	ـكـ	ـك	*k*
Lam	ل	لـ	ـلـ	ـل	*l*
Meem	م	مـ	ـمـ	ـم	*m*
Noon	ن	نـ	ـنـ	ـن	*n*
Ha'	ه	هـ	ـهـ	ـه	*h*
Waw	و	و	ـو	ـو	*w*
Ya	ي	يـ	ـيـ	ـي	*y*

The Arabic alphabet consists of 28 basic letters, which consist of strokes and dots. Ten of them have one dot, three have two dots, two have three dots. Dots can be above; 'ن ض', below; 'ب ي' or in the middle; 'ج ظ' of the letter. The shape of the letter is context sensitive, depending on its location within a word, as shown in Table 1. A letter can have up to four different shapes: isolated, beginning connection from the left, middle connection from the left and right, and end connection from the right. Most of the letters can be connected from both sides, the right and the left. However, there are six letters which can be connected from one side only; the right. These letters are grouped in the following two words: 'ذر وزاد'. This characteristic implies that each word may form one or more sub-words, and each sub-word may contain more than one letter. As an example, consider the two word 'محمد – Mohammad' and 'عربي – Arab'. While the word 'محمد' forms a single connected component, the word 'عربي' forms two connected components. The reason is that all letters in the first word can be connected from both sides, while the letter 'ر' in the second word may only be connected from the right side, which causes a discontinuity of cursiveness.

Some Arabic letters may have a zig-zag-like stroke called *Hamza*. This additional character can be on *Alif* 'أ', below *Alif*, 'إ', on *Waw* 'ؤ', on a *Alif-Maqsura* 'ئ', or isolated on the line 'ء'. Another non-basic character is *Ta-Marbuta* 'ة'. It is a special form of the letter 'ت', and it is always at the end of the word. It can be connected as in 'سرعة' or isolated as in 'هرة'.

Writing styles may be classified according to their complexity into three categories:

1. *Typewritten or machine-printed:* this is a computer-generated style, and it is the simplest among all styles because of the uniformity in writing a word.
2. *Typeset:* this is normally used to print newspapers and books. Typeset style is generally more difficult than the machine-printed style, because of the existence of overlaps and ligatures, which poses a challenging problem, (see Fig 1). Ligatures occur when two or more letters overlap vertically and touch. By contrast, overlaps occur when two or more letters overlap vertically without touching. Recently, some computer-generated fonts have imitated the typeset style in providing ligatures and overlaps.
3. *Handwritten:* this is assumed to be the most difficult style because of the variations in character shape even if it is rewritten by the same person. Handwritten style may be further classified into: scribe, personal and decorative. Scribe is more carefully written than the personal handwriting style that represents the daily usage of Arabic

Fig. 1. A sample of Arabic writing. The two word are 'الخليفة – The caliph' and 'أبوبكر – Abu Bakr'.

alphabet by individuals. Few people are able to perform an exquisite scribe handwritten script. Scribe handwriting is certainly different from decorative handwriting, which is normally used for adornment purposes.

A *Baseline* is an important characteristic of Arabic writing. This is a horizontal line that runs through the connected primitives of a text. In a binary image test, the baseline is assumed to have the maximum number of black pixels. This assumption is no longer valid if the script is skewed, (see Section 3.2.2). If the script is handwritten, the baseline is not usually straight, and may only be estimated.

Arabic characters may have diacritics which are written as strokes and can change the pronunciation and meaning of the word. Diacritics may appear as strokes above the character as *Fat-ha* in زَرَعَ, *Dhamma* in جُزُر, *Sukun* in يَكْتُبْ, *Shadda* in نوّر, or *Maddah* in آه, or below the character as *Kasra* in إبِل. *Tanween* is a form of diacritising Arabic script but with double *Fat-ha* as in شيئًا, double *Dhamma* as in شيءٌ, or double *Kasra* as in شيءٍ. These diacritics perform an essential function in transliterating an Arabic script, as described below. In spite of this importance, text may be un-diacritic, and readers of Arabic are accustomed to inferring the meaning from the context.

Arabic transliteration is concerned with rewriting Arabic words in the Latin alphabet. One reason for transliteration is that it is not always possible to render an Arabic text into English. This is because at times there is no corresponding equivalent for an Arabic word in English. As a result, transliteration can serve two purposes: acting as an intermediate step toward translation; and transforming the Arabic script into Roman letters so it can be easily read by non-Arabic readers.

To illustrate the effect of diacritics on transliteration, consider the following word 'كلية', which can be pronounced as either 'كُلِيّة, *kuliyyt* – college', or 'كِلية *kilyat* – kidney'. Thus, for an un-diacritic Arabic word there may be a vast number of transliterations, whereas for the diacritic Arabic word it is usually a one-to-one table matching process. To remedy the problem of undiacritised words, a two-way Arabic morphological system (analysis/generation) has been developed to deal with voweled, semi-voweled, and non-voweled Arabic text [20].

3. ARABIC TEXT RECOGNITION SYSTEM

The Arabic text recognition system can be broken down into a number of stages: image acquisition, preprocessing, segmentation, feature extraction and classification and recognition. Figure 2 illustrates these stages according to their order of occurrence.

The techniques described in this section are illustrated by applying my implementation of these techniques to Arabic manuscript samples obtained from Cambridge University Library (UL) [21].

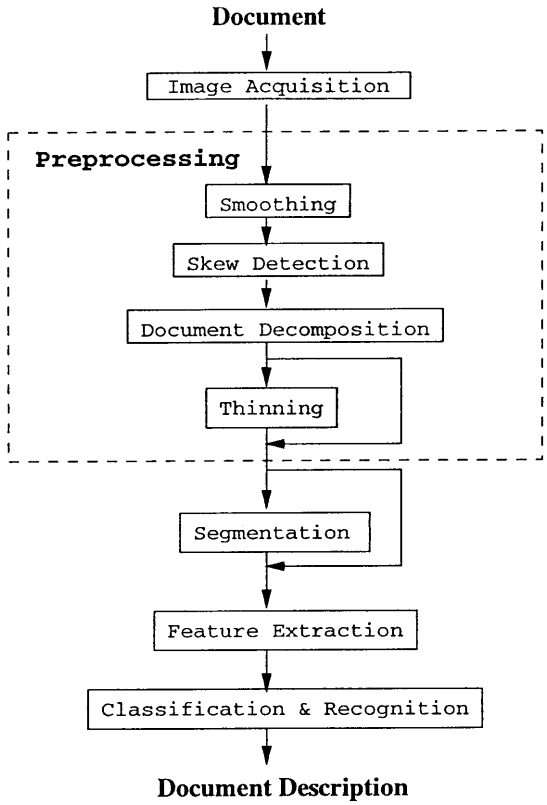

Fig. 2. A general diagram for the Arabic text recognition system.

3.1. Image Acquisition

This is the first step in the recognition system. The objective is to acquire the text and transform it into a digitised raster image. Figure 3 shows two types of character recognition systems in terms of acquiring their input: on-line and off-line recognition systems.

The on-line recognition system recognises the text as it is being written [22,23]. The preferred input device is an

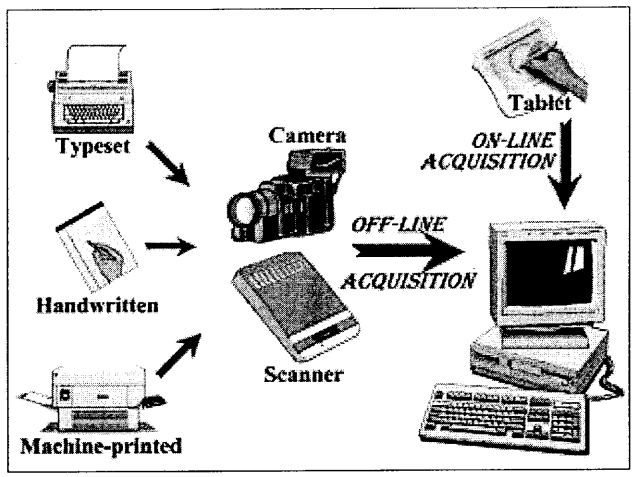

Fig. 3. Different text acquisition methods.

electronic tablet with a stylus pen. The electronic tablet captures the (x,y) co-ordinate data of pen-tip movement, which typically has a resolution of 10 points/mm, a sampling rate of 100 points/s, and an indication of pen-up and pen-down [24]. The on-line recognition system has two major advantages: the high-recognition accuracy and the interaction. The first advantage is that on-line recognition captures a character as a set of strokes, which are represented by a series of co-ordinate points. The second advantage is that it is very natural for the user to detect and correct unrecognised characters immediately by verifying the recognition results as they appear. On the other hand, on-line recognition is limited to recognising handwritten text.

The off-line recognition system recognises the text after it has been written or typed. The system may acquire the text using a video camera [25–29] or a scanner [30–33]. The latter is commonly used because it is more convenient, it introduces less noise into the imaging process, extra features such as automatic binarisation and image enhancement can be coupled with the scanning process to enhance the resulting image text and, most importantly, it is more relevant to the problem of recognising written script.

For document management applications the aim is to speed-up the scanning process to maximum speed. The scanner, here, can run at 300 dots per inch (dpi), and are designed with a high volume document feeder and high-throughput Small Computer System Interface (SCSI) that can process 85 pages per minute (ppm).

Lower resolution and poor binarisation can contribute to readability when essential features of characters are deleted or obscured. The resulting image can also be affected by the presence of marking or stains, or if the document has been faxed or copied several times. The latter causes a diminishing of contrast, the appearance of 'salt and pepper' noise, and the false appearance of text by becoming either thinner or thicker than the original document.

Binarisation, or thresholding, is a conversion from a grey-level image to a bilevel image, (see Fig 4). A bilevel image contains all of the essential information concerning the number, position and shape of objects while containing less information. The simple and straightforward method is to select a threshold value, then all pixels with a grey-level below this threshold are classified as black, and those above as white.

The threshold must be determined from the pixel values found in the image, for example the use of the mean grey level in the image as a threshold. Another method is by using the histogram of the grey levels in the image. Given a histogram and the percentage of black pixels desired, one can determine the number of black pixels by multiplying the percentage by the total number of pixels, then simply count the pixels in histogram bins, starting at bin 0 until the count is greater than or equal to the desired number of black pixels. The threshold is the grey level associated with the last bin counted. Other approaches such as: using edge pixels, iterative selection and using entropy, can also be applied [34–36]. In the edge pixel method, the threshold is found by first computing the Laplacian of the input image, then selecting those pixels with large Laplacian values. In

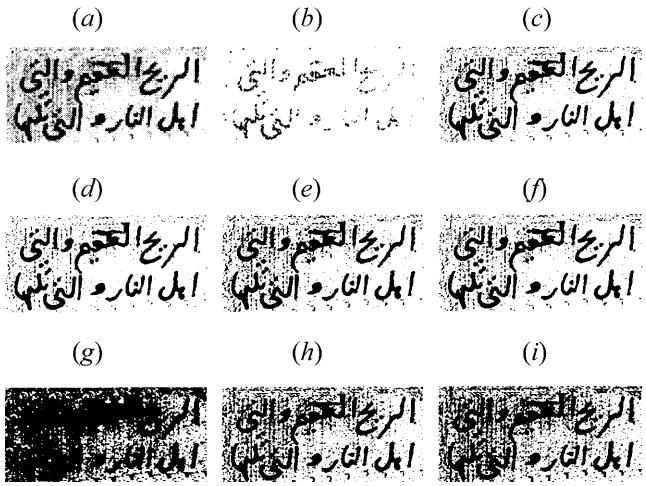

Fig. 4. Sample results from eight single-threshold selection methods. (a) Original, (b) Two Peaks = 10, (c) G.L.H. = 146, (d) Entropy = 125, (e) Yager =153, (f) I.S. = 144, (g) Min. Err. = 253, (h) Mean = 193, (i) Pun = 215.

the iterative selection method, the threshold is first calculated as the average value. The average values of the object and the background classes are iteratively calculated, and the mean of these two values represents the new threshold. The entropy method treats the image as a source of information. Each of the entropys associated with the black pixels and the white pixels is weighted with a calculated probability. The threshold is found by maximising a predefined equation based on the implemented algorithm.

3.2. Preprocessing

The OCR system depends upon both the original document quality and the registered image quality. The preprocessing stage attempts to compensate for poor quality originals and/or poor quality scanning. This is achieved by reducing both noise and data variations. All image acquisition processes are subject to noise of some type, therefore there is no ideal situation in which no noise is present. Noise can neither be predicted nor measured accurately from a noisy image. Instead, noise may be characterised by its effect on the image. There are two defined types of noise: signal-independent noise and signal-dependent noise. Signal-independent noise adds a random set of grey levels, statistically independent of the image data, to the pixels in the image. In signal-dependent noise, the value at each point in the image is a function of the grey level there.

3.2.1 Smoothing. This reduces the noise in an image using mathematical morphology operations. Two operations are mainly used, Opening and Closing. Opening opens small gaps or spaces between touching objects in an image; this will break narrow isthmuses and eliminate small islands. In contrast, Closing fills small gaps in an image; this will eliminate small holes on the contour. Both Opening and Closing apply the same basic morphology operations, namely, Dilation and Erosion, but in the opposite order.

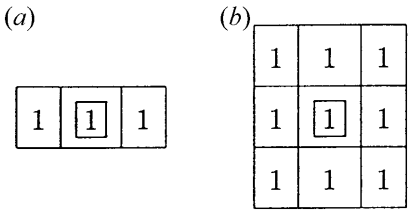

Fig. 5. Two structuring elements. The origin is surrounded by a frame. (a) SE_1, (b) SE_2

Fig. 6. Results of opening and closing using structuring elements in Fig. 5. (a) The original text image, (b) opening (a) with SE_1, (c) closing (b) with SE_1, (d) opening (a) with SE_2, and (e) closing (d) with SE_2.

Opening applies an erosion operation immediately followed by a dilation operation using the same Structuring Element, (see Fig 5). Closing applies a dilation operation immediately followed by an erosion operation, again, using the same structuring element.

In Dilation a small area around a pixel is set to a given pattern. This can be mathematically represented as

$$A \oplus B = \{c | c = a + b, \quad a \in A, b \in B\} \tag{1}$$

A represents the image being dilated, and B is a second set of pixels called a structuring element.

The erosion of image A by structuring element B can be defined as

$$A \ominus B = \{c | (B)_c \subseteq A\} \tag{2}$$

It is the set of all pixels c such that the structuring element B translated by c corresponds to a set of black pixels in A. Figure 6 illustrates the results of opening and closing using the two structuring elements shown in Fig. 5. A drawback of this method is that there is no standard approach to obtain the best structuring element to be implemented.

Another approach to reducing salt-and-pepper noise is by applying the median filter, (see Fig. 7). This is a small window which passes through all pixels in the image. The pixel in the centre is replaced by the median value of all

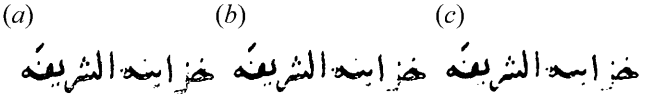

Fig. 7. Results of removing noise from the original text image in Fig. 6 using median filter; window size is (a) 3 × 3, (b) 5 × 5, and (c) 7 × 7.

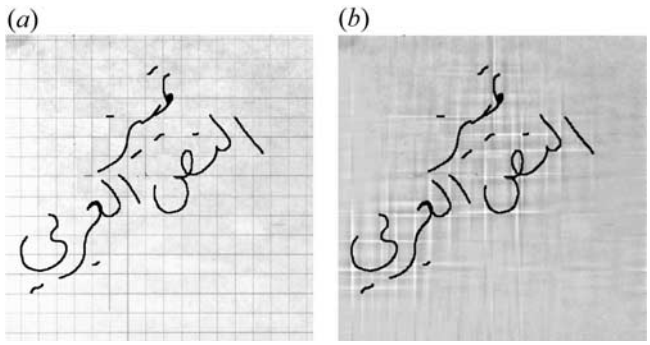

Fig. 8. Removal of grid lines using a notch filter. (a) original image, (b) restored image.

the pixels in the region. The median filter is slow, requiring not only a pass through the image of the window, but also needing the pixels' values in that window to find the median. Furthermore, it may reduce the contrast of the edges and close small gaps between different objects in the image.

At times, writing can be on a graph paper, thus scanning the original paper will produce an image that is disturbed by structured or periodical noise. In Fig. 8, The Fourier Transform [35] is used to determine where the peak frequencies are. The noise corresponds to one of these peaks, and can be virtually eliminated by clearing those regions in the frequency domain image that correspond to the noise frequencies, and then back-transforming it into a regular image.

Mahmoud [32] implemented a statistically based smoothing algorithm to eliminate small areas and fill little holes. The algorithm modified each pixel according to its initial value and to the values of its 8-neighbours.

3.2.2. Skew Detection and Correction. Scanning a document so that text lines are within about three degrees of the true horizontal is acceptable. This is feasible if the document is aligned manually on the object glass of the scanner. Recent scanners are equipped with automatic feeders which may cause the document to rotate up to 20 degrees of the true horizontal. One of the first steps in attempting to read this document is to estimate the orientation angle, the skew angle, of the text lines. This process is called *skew detection*, and the process of rotating the document with the skew angle, in the opposite direction, is called *skew correction.*

The common, and perhaps the most efficient, approach to estimate the skew angle is to use the Hough Transform [36–38] (Fig 9). The Hough Transform is a method for

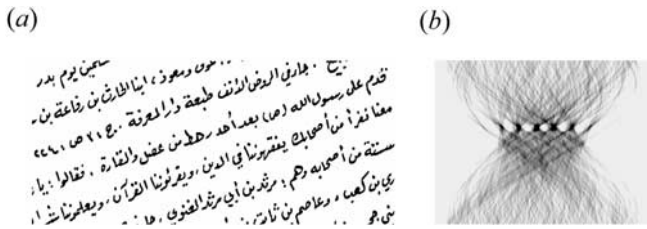

Fig. 9. Skew detection. (a) Document image rotated 18 degrees to the left, (b) the Hough Transform image of (a).

detecting straight lines in a raster image. Infinite straight lines that pass through this pixel and satisfy the following equation may represent each black pixel in the image:

$$y = mx + c \quad (3)$$

where (x, y) are the co-ordinates of the pixel, m is the slope of the line and c is the intersection of the line with the Y axis.

Another approach to estimating a skew angle is based on using bounding boxes of *Connected Components* [39]. This is a two step process. In the first step, all 8-neighbour connected pixels are grouped as distinct components, and the centre of gravity for each component is calculated. In the second step, a virtual line is drawn between various centres. The angle between this line and the horizontal represents the skew angle. However, implementing this approach to Arabic text image may be misled by dots and diacritics located above and below the word characters.

3.2.3. Document Decomposition. A document image is a visual representation of a printed page. Typically, this page consists of blocks of text that are interspersed with tables and figures. Methods of deriving the blocks can take advantage of the fact that the structural elements of a document are generally laid down in rectangular blocks aligned parallel to the horizontal and vertical axes of the page. The document decomposition and structural analysis task can be divided into three phases [40]: Phase one consists of block segmentation where the document is decomposed into several rectangular blocks. Each block is a homogeneous entity containing one of the following: a text, an image, a diagram or a table. Phase two consists of block classification. Each block is assigned a label (title, regular text, picture, table, etc.) using properties of individual blocks from phase one. Phase three consists of a logical grouping and ordering of the blocks. For OCR the concentration is focused on text blocks.

Work on Arabic has been limited to text documents, thus the notation of *document decomposition* means the separation of text lines and the segmentation of words and sub-words. The classical method for identifying text lines in an Arabic text image is to use a fixed threshold to separate the pairs of consecutive lines [41,42]. This threshold is obtained using the distances between various baselines of the text. The median of different distance values is an appropriate selection.

An alternative approach [43] is to use the horizontal projection and look for the pixel lines that have a density of zero, then consider that every text line is situated between two blocks of zero density pixel lines, (see Fig 10). This method is enhanced by identifying the lines of pixels that have the largest density in the text [44]. The upper and lower parts are then analysed with respect to these lines.

The next phase is to segment a text line into words and sub-words. Words and sub-words are determined by inspecting the vertical projection. An average threshold value computed from all vertical gaps is used to determine whether a spacing is an inter-word spacing or an intra-word spacing [41,42,45], as shown in Fig. 10.

Another attempt at decomposing the Arabic script into words is based on the connected components of that script,

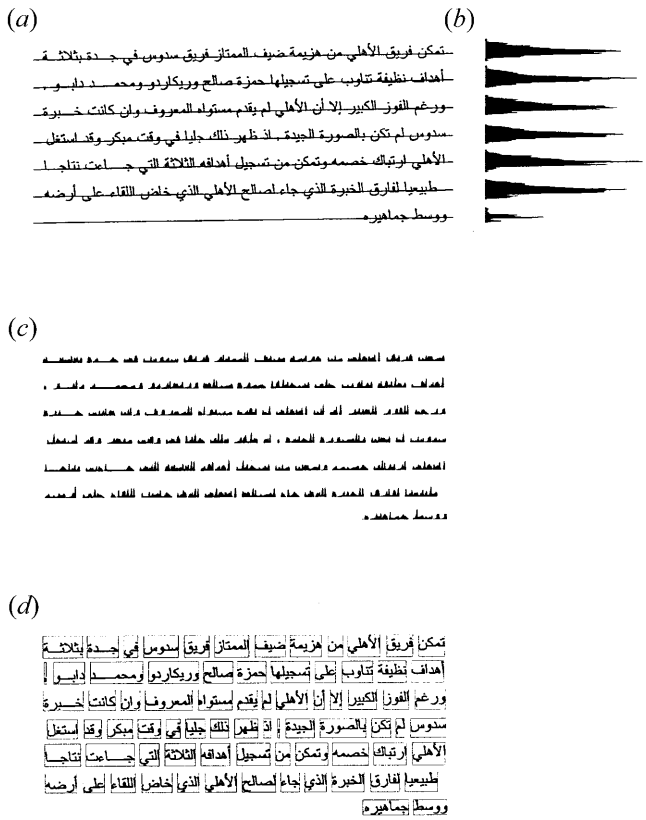

Fig. 10. Document decomposition. (a) Original document image, (b) the horizontal projection, (c) the vertical projection of each text line, and (d) the document image after segmenting it into words.

as shown in Fig. 11. This method can solve the difficulty of segmenting handwritten script, where a clear cut between two consecutive text line may be not possible to find. Abuhaiba et al [46] automated the process of combining secondary strokes with appropriate main strokes. This process depends on finding a perfect bipartite graph with minimum cost, and it is known as the *assignment problem.*

3.2.4. Slant Normalisation. This problem may be clearly seen in handwritten words, although machine-printed words with italic fonts suffer from the same problem. Kim and Govindaraju [47,48] proposed an algorithm to correct slant angle in which vertical and near vertical lines are extracted by tracing chain code components using a pair of one dimensional filters, each being a five element array of different weights. A convolution operation between the filter and five consecutive components was applied by sliding the filter one component at each iteration.

3.2.5. Thinning and Skeletonisation. These are the operations that produce the skeleton. A skeleton is presumed to represent the shape of the object in a relatively small number of pixels, all of which are structural and have semi-equal distance from two or more contour points.

Thinning algorithms may be classified into parallel and sequential. The parallel algorithms [49,50] operate on all pixels simultaneously. In contrast, the sequential algorithms

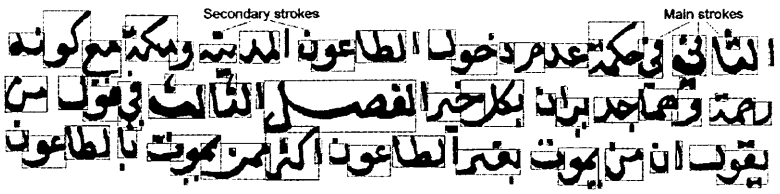

Fig. 11. Document connected components.

[51,52] examine pixels and transform them, depending on the preceding processed results. The approach in both cases is to remove the boundary pixels of the character that are neither essential for preserving the connectivity of the pattern, nor for representing any significant geometrical feature of the pattern. The process converges when the connected skeleton does not change or vanish, even if the iteration continues. The literature contains many thinning algorithms most of which are susceptible to noise in that the generated skeletons are sensitive to even small variations in the input pattern [35,36,53–55].

Most of the existing thinning algorithms operate by iteratively stripping contour pixels. The main problem associated with this approach is that the computation speed is very low, due to its layer-to-layer stripping nature. An alternative is to implement a vectorisation-based algorithm [56] which operates directly on the run length encoding of a binary image. Mahmoud et al [30] proposed a new algorithm for skeletonisation of isolated Arabic characters that was based on clustering the character image. The skeleton was then generated after finding the adjacent matrix of different clusters. Finally, they refined the skeleton by removing insignificant vertices. Altuwaijri and Bayoumi [57] implemented a self-organising neural network to cluster the Arabic character. Plotting the cluster centres and connecting adjacent clusters generated a straight-line sequence, which formed the skeleton.

Another alternative is driven by Euclidean distance mapping [58], which guarantees the invariance under isometric transformation of the results. The computational complexity of these algorithms has been considerably reduced with respect to other thinning algorithms. Moreover, the results generated by these algorithms are better than those generated by traditional thinning algorithms. However, these algorithms were only applied to Latin script.

Figure 12 shows the implementations of two different thinning algorithms. While the first algorithm preserved the inner tooth of س in the word سَماعة, the second algorithm failed to do the same. In contrast, while the first algorithm generated a surprising branch for ع in the word ٱلكعبة, the second algorithm managed to produce the correct skeleton in spite of the varying stroke width. The algorithm results may even vary when implemented to different fonts. Consider the skeletons shown in Figs 12(b) and 12(e); while the algorithm managed to separate the two dots of ة in ٱلكعبة it failed to do the same when processed the word سَماعة. This is due to different font styles.

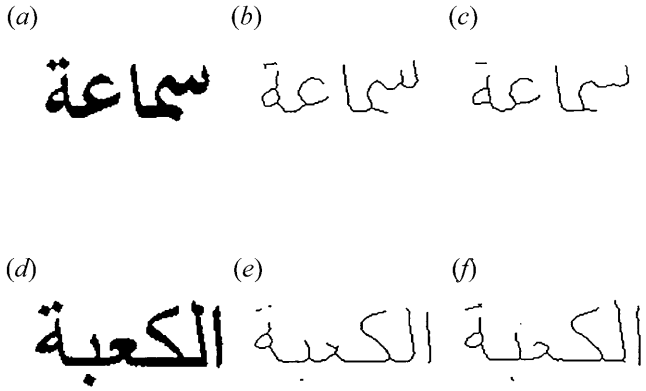

Fig. 12. Sample results of a number of thinning algorithms.

3.3. Segmentation

By now the document image has been enhanced, decomposed into blocks and text lines. In addition, words and sub-words have been extracted from text blocks. At this stage, the system focuses on how it prepares the words and/or the sub-words for the feature extraction. As mentioned previously, cursiveness is the main obstacle facing any Arabic text recognition system whether it is on- or off-line.

Segmenting on-line Arabic handwriting is much simpler than segmenting off-line machine-printed Arabic words [59–64]. This simplicity motivated the work carried out in developing an algorithm to restore the temporal information in off-line Arabic handwriting, so that on-line recognition systems may be used [65].

Arabic text recognition systems can be categorised, relative to their approach in tackling word segmentation, into segmentation-based systems and segmentation-free systems.

3.3.1. Segmentation-based Systems. These can be divided into four categories:

1. *Isolated/pre-segmented characters:* researchers here recognise numerals [66,67], isolated characters [68,69], or assume that these characters result from a reliable segmentation algorithm [70,63,71,32]. These systems are not practical, except if we consider mathematical formulas or the indexing of diagrams.
2. *Segmenting a word into characters:* this is the first approach used for segmentation [72–74]. The system attempts to segment a word into its characters, then recognise each character separately. The reason behind the emergence of

this approach is the simplicity of the recognition afterwards, since the cursiveness obstacle is not present and the problem is now similar to Latin OCR.

Some research [31,42,74] has proposed segmentation algorithms that are based on the vertical projection of the word image. The connectivity points show the least sum of the average summation over all columns. This results in a number of segments which are then connected together to form the basic shape of the character, (see Fig. 13).

Another technique uses a two level segmentation scheme [75]. After segmenting a word into its characters using horizontal and vertical projections, a lower level of segmentation is applied to isolate the dots and zigzag-like shapes.

In Yarman-Vural and Atici [33], the segmentation stage partitions a sub-word into its character segments. The algorithm is based on extracting a subset of key feature segments in the sub-word, and identifying cut points in each segment. The subset of the key feature segments is obtained by tracing the contour in the clockwise direction. Dots and diacritics are not considered during segmentation. This type of segmentation is a cause of recognition errors, and hence a low recognition rate.

3. *Segmenting a word into primitives:* this segments a sub-word or connected component into symbols where each symbol may represent a character, a ligature, or possibly a fraction of a character, (see Fig. 14). Abdelazim and Hashish [76] calculated the vertical projection for each column, then obtained significant primitives by traversing the resulting curve with a selected threshold value.

The baseline represents an important feature for Arabic writing. Parhami and Taraghi [17] identified a sequence of connection points on the baseline of Farsi script, which uses the Arabic alphabet. The connection point is where the baseline changes from/to its normal thickness. The alphabet was then divided into three major groups, accordingly. This method may not be efficient when applied to handwritten script, where stroke size varies relative to the pen pressure on the page. Tolba and

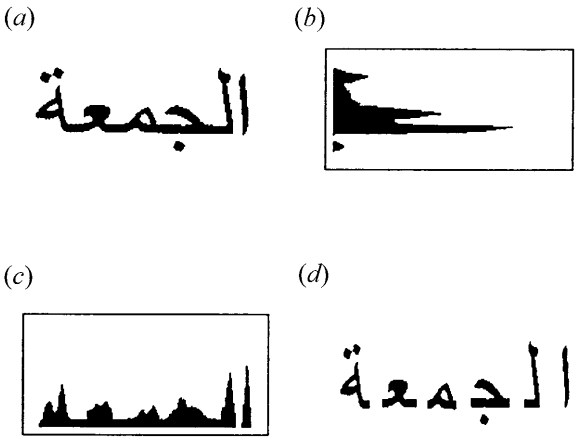

Fig. 13. Segmenting a word into its characters. (a) Original, (b) Horiz. projec., (c) Vertical projec., (d) Segmented word.

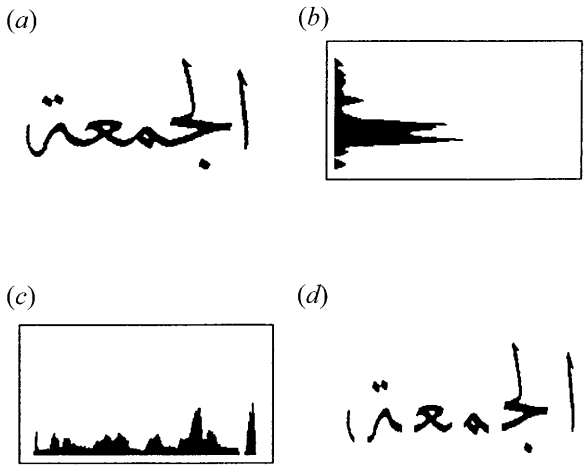

Fig. 14. Segmenting a word into primitives. (a) Original, (b) Horiz. projec., (c) Vertical projec., (d) Segmented word.

Shaddad [77] also utilised the baseline. They slid a window along the direction of writing, and at each instant a segmentation parameter was calculated to match the content of the window with a predefined set of primitives. If the segmentation was less than a certain threshold, the region was marked as a 'Silence Region'.

Motawa et al [78] applied morphological operations, *opening* and *closing*, to a word image to allocate *singularities* and *regularities*. Singularities represent the start, the end, or the transition to another character. Regularities contain the information required for connecting a character to its successor. Accordingly, these regularities are excellent candidates for segmentation.

Some researchers extracted the dots and the diacritics in advance. The connected components were then segmented into meta character glyphs [79], principal strokes [29], or character segments [33,80].

The set of boundary pixels or the contour includes important information of an object which can be segmented into primitives [81], (see Fig. 15). This may be done by finding points on the contour where there is a transition from a column, which has all its black pixels within the baseline boundaries, to another column, which does not [82]. Another approach is completely based on tracing the outer contour [14,83] of a given word, and calculating the distance between the extreme points of the intersection of the contour with a vertical line.

Kavianifar and Amin [84] divided the contour into three classes: main body or stroke, complementary character, and noise. They set two thresholds to help find the equivalent contour class based on the contour length.

A skeleton is a compact representation of a word image. It can be traced in the same way that Arabic

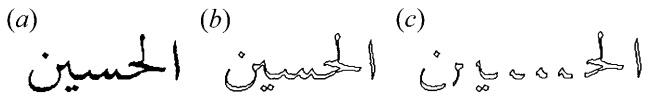

Fig. 15. Segmenting the contour of a word. (a) Original, (b) The contour, (c) Segmented contour.

writing is taught to segment the word into a stroke sequence [25,85], structural features such as loops and branches [86], or principal and secondary strokes [29]. Alternatively, a character skeleton can be converted to a tree structure, and each character is then represented by a single fuzzy constrained character graph model [87].

Al-Muallim and Yamagushi [25] segmented a word into strokes. The extraction of a stroke was made by finding out its start point, and then following the curve to a point which was inferred to be the stroke endpoint.

Khorsheed and Clocksin [88,89] decomposed the word skeleton into a sequence of links in orthographic order. Each link commences and terminates with a feature point. A feature point is a black pixel in the skeleton which has a number of transitions from the foreground to the background within its 8-neighbour pixels equals one, three or four, and that pixel is referred to as an *End point*, *Branch point* or *Cross point*, respectively.

4. *Integration of recognition and segmentation:* this claims that the procedure resembles, to a great extent, the human recognition process. The segmentation here is performed after recognition. The approach is to scan the word starting from the far right, and at each step either cluster a column to one of the codebook entries [90] or calculate accumulative moment invariants [91]. The system is not always able to recognise all characters, which implied that all succeeding characters in that sub-word would not be processed. To remedy this drawback, El-Dabi et al [92] developed a backup scanning algorithm that was triggered when such a blockage happened.

   The previous approach may produce a long segment sequence. Mosfeq [93] presented a segmentation algorithm as a centring operaton involving a focus of attention. A character was correctly segmented if it appeared in the centre of a large window, regardless of what else appeared in that window.

3.3.2. Segmentation-free Systems. This scheme of text recognition is motivated by discoveries in psychological studies of the human reading process [94]. It attempts to recognise the whole representation of a word without trying to segment and recognise characters or primitives individually. This approach was originally introduced for speech recognition [95].

One approach of the word level Arabic recognition was to analyse the word shape with a unique vector of features, then this feature vector might be matched against a database of analogous feature vectors [96], or represented in attribute/value form to an inductive learning system [97].

Another approach implemented the morphological Hit-or-Miss Transform [98], which was based on marking the location at which a structuring element fits within a pixel set corresponding to a shape of interest and another structuring element lies outside the pixel set. Shape primitives located on the whole page were then combined into characters [99].

A third approach was based on choosing a text line as the major unit for training and recognition [100–102]. When a page was decomposed into text lines, the horizontal position along each line was selected as an independent variable. Hence, a text line was scanned from right-to-left, and at each horizontal position a set of features was extracted from a narrow vertical strip. The system was based on hidden Markov models, where each character was represented by a separate model. The output was a sequence of characters that had the highest likelihood.

A new technique for recognising typewritten and handwritten Arabic cursive words has also been presented [103,104]. The technique treated the word as a whole. Each word was represented by a set of Fourier coefficients extracted from the word image.

3.4. Feature Extraction

A feature is a measurement made on a glyph, and combining it into a vector is a simple way of collating multiple measurements. Ideally, the features extracted from an image capture the essential characteristics of the character or the word by filtering out all attributes which make a character/word in one font different from the same character/word in another. At the same time, they preserve the properties that make one character/word different from another character/word. Feature types can be categorised as follows.

3.4.1. Structural Features. Structural features are the most popular features investigated by researchers [1]. Structural features describe geometrical and topological characteristics of a pattern by representing its global and local properties [35,36,105].

The extracted structural features depend on the category of the pattern to be classified. For Arabic character, word and text recognition, the features include strokes and bays in various directions, endpoints, intersection of line segments, loops, stroke positions relative to the baseline, dots and their positions relative to the baseline, and zigzag [66,85,89,97,106].

Feature space can be divided into more than one independent division, and the extracted features can be different in each division [33]. Sometimes, based on the preliminary features, a character/word image may be assigned to a certain group where further feature extraction is carried out [25,86,107].

Structural features can tolerate distortion and variations in writing (multifonts and handwriting), however, they are not easy to extract. A combination of several structural features could enhance the overall recognition rate [88,89].

At times, it is very important to locate pixels with certain properties on the skeleton of a word to act as delimiters helping extracting strokes [69,85]. Alternatively, the word image may be scanned vertically, column by column, in order to produce a structural feature vector for each column [80].

3.4.2. Statistical Features. Statistical features are derived from the statistical distribution of pixels and describe the characteristic measurements of a pattern. These include zoning [102,108], which features the density distribution of character pixels, characteristic loci [90], which counts the one and the zero segments and the length of each segment,

the ratio of pixel distribution between two parts of the image [44,109,110] and moments [28,45,91,111].

The text line was partitioned by some researchers [100,101,102] into a sequence of narrow vertical overlapping frames, with a width that was a small fraction of the height of the line. Each frame was divided into 20 equal overlapping cells. Features extracted from each cell were: the intensity of a cell, the vertical and horizontal derivative of intensity, and the local slope and correlation across a window of two cells. The frame could have a one-pixel width, as in Abdelazim et al [90]. The system extracted a feature vector from each column of pixels in the word image, where each feature represented the length of black/white pixel run.

Moment invariants refer to certain functions of moments [53], which are invariant to geometric transformations such as translation, scaling and rotation [35,112]. Moment invariants are sensitive to any change and multi-font recognition. To remedy this, Al-Khatib et al [81] integrated three feature extraction methods: moment invariants, border transitions and perimeter-area ratio.

Generally speaking, statistical features are easy to extract; nonetheless, they may be misleading, due to a fraction of noise brought forth haphazardly according to the binarisation process.

3.4.3. Global Transformation. Global transformation techniques transform the pixel representation to a more compact form. This reduces the dimensionality of the feature vector, and provides feature invariants to global deformation like translation, dilation and rotation. Zaki et al [113] used the projection transform to convert a character image of order $M \times N$ into a projection vector of order $M+N$. Amin et al [31,41] implemented projection transform to represent the character image as a string of primitives.

Fourier Descriptors (FDs) [114] were successfully implemented for Arabic OCR [32,83]. FDs use the coordinates of the contour pixels, so the character's closed boundary can be represented by a periodic function. Mahmoud [32] integrated another global transformation technique with FDs, which is the boundary line encoding technique. This technique is based on tracing the contour of the character to generate directions, direction lengths and curvature features.

Another implementation transformed each word into a normalised polar image, then applied the two-dimensional Fourier transform to the polar image. The resulting spectrum tolerated variations in size, rotation or displacement. Each word was represented by a template that includes a set of Fourier coefficients. The recognition of input word, wether typewritten [115] or handwritten [103], was based on the normalised Euclidean distance from those templates.

Other researchers applied a chain-code transformation. This represents the boundary pixels of the character using Freeman code [116]. Starting from an initial pixel, a sequence of codes tracing the directions of consecutive pixels. Features extracted from the Freeman code may be described as directional vectors [42,32], and they may be combined with other features [73].

3.5. Classification and Recognition

A major task after feature extraction is to classify the object into one of several categories. There are a number of various classification techniques applied in text recognition.

3.5.1. Minimum Distance Classifier. Given K different classes, where each class is characterised by a feature vector prototype, the problem is to assign an input feature vector to one of these classes according to a predefined discriminant function. The features can be geometrical [17], statistical [32,92] or structural [73].

Abuhaiba and Mahmoud [70] designed a set of fuzzy constrained character graph models, then an input character, which has been converted into a tree structure, was assigned to the character model which had the maximum degree of acceptance. This process can be repeated on more than one level [107], where a character was first assigned to the nearest group among the 96 native groups. The character was then recognised as one of the characters in the native group, based on the minimum distance between the input character and the template character. The common distance measure is the Euclidean distance [34], but Hamming distance and Ascher et al scores were also applied to recognise Arabic characters [117]. The K-means clustering algorithm [95] was implemented successfully in Arabic OCR for one-dimensional [113] and two-dimensional [76] feature vectors.

3.5.2. Decision Tree Classifier. This classifier splits the N-dimensional feature space into unique regions by means of a sequential method. The algorithm is such that every class need not be tested to arrive at a decision. This becomes advantageous when the number of classes is very large [28,31,42,74], where the dictionary was composed of a tree and the nodes were labelled with character names. Each node of the dictionary was associated with a Boolean variable indicating if the path joining the root to the terminal node corresponded effectively to an existing word. If during the ongoing sequential identification process several models are candidates, then the last mentioned attribute is calculated to make the final decision [86]. As in the previous category, classification here can be a two-step process [25,29,63,85]. In the first step, an input character is assigned to one of the main groups according to some syntactic rules. Then, and relative to a more detailed feature vector, the input character is matched with one of the group members.

Abdelazim and Hashish [76] implemented a template correlation matching and a tree classifier to discriminate among primitives in the same cluster group. The recognition process of an unknown pattern was propagated sequentially by following a path through the decision tree from the root node to the leaf node. The leaf node was either labelled with an identified primitive or could be a reject node. It was found that the tree method was faster than a template matching against idealised reference patterns.

Amin [97] used the $C4.5$ learning algorithm to create decision trees to represent classification rules. A node in a tree represented a test on a particular attribute, thus when

an object reached a leaf node, it was classified according to the name of that leaf node.

3.5.3. Statistical Classifier. This classifier assumes that different classes and the feature vector have an underlying joint probability. One approach is to use the Bayes classifier [75]. The Bayes classifier minimises the total average loss in assigning an unknown pattern to one of the possible classes. The probability density function can be cumulative [109], therefore at the end, the assignment is to that class with majority samples.

Al-Badr and Haralick [99,118] implemented a three-step recognition process: first they found instances of a set of shape primitives on a text image; then they took the detected primitives of a word and hypothesised a number of alternative strings as the recognition of the word. The choice would be on the one with the maximum *a posteriori* probability. Finally, the probability of a match was computed between the symbol model and the word image.

Hidden Markov Models are statistical models which have been found extremely efficient for a wide spectrum of applications, especially speech processing [119,120]. This success has motivated researchers to implement HMMs in character recognition. Abdelazim and Hashish [66] first applied HMMs to recognise Hindu numerals; '٠١٢٣٤٥٦٧٨٩'. Each numeral was represented with a separate HMM. The observation sequence was passed to all the ten models, and assigned to the numeral with the highest model probability of the observation sequence $P(O|\lambda)$.

Recently, HMMs have been also implemented to recognise Arabic text. In Allam [82], the contour of the word image was segmented after baseline estimation. This resulted in a sequence of labels, the latter of which was classified by finding the HMM that gave the highest probability. To reduce the computation time and enhance the recognition rate, a segment was not compared to the whole set of models, but compared to a selected group according to the position of that segment within the word. Other ongoing research is by Makhoul et al [100,101,102]. Their system depends upon the estimation of character models, a lexicon, and grammar from training samples. The training phase takes scanned lines of text coupled with the ground truth, the text equivalent of the text image, as input. Then, each line is divided into narrow overlapping vertical windows from which feature vectors are extracted. The character modelling component takes the feature vectors and the corresponding ground truth and estimates the character models. The recognition phase follows the same step to extract the feature vectors, which are used with different knowledge sources estimated in the training phase to find the character sequence with the highest likelihood $P(O|\lambda)$.

In Khorsheed and Clocksin [88,89], a single HMM was built for all words in the lexicon. Each word was represented by one path through the model. The model was composed of multiple character models each represented a letter in the alphabet. The observation sequence of the input word was thrown to the HMM, and the Viterbi algorithm was applied to issue an order list of the system output paths. The same approach was implemented to recognise handwritten cursive Arabic words [21]. This time the system was lexicon-free. The training of each character model was performed separately using Baum–Welch method [121].

3.5.4. Neural Network Classifier. OCR is one of the most successful applications that has been proposed for neural networks. A Neural Network (NN) [122] is a non-linear system which may be characterised according to a particular network topology. This topology is decided by the characteristics of the neurons and the learning methodology. There are three main advantages behind implementing NNs in OCR: NNs have faster development times; they have an ability to automatically take into account the peculiarities of different writing/printing styles; and they can be run on parallel processors. On the other hand, introducing a new shape to the NN requires that the network be retrained, or even worse, that the network be trained to a different architecture. Intensive work can be found on the subject of Arabic OCR using neural networks [79,110,111,123,124]. NNs can simply cluster the feature vectors in the feature space [69,111,125], or they can integrate feature extraction and classification stages by classifying characters directly from images [93,123,126]. NNs were also applied to recognise Arabic words on-line [127].

Generally speaking, the common architecture of NNs used in Arabic OCR is a network with three layers: input, hidden and output. The number of nodes in the input layer varies according to the dimensionality of the feature vector or the segment image size. The number of nodes in the hidden layer govern the variance of samples that can be correctly recognised by this NN [67,81].

In Sanossian [111], two different NN architectures were employed: architecture one was a network of 13 input nodes, 20 hidden nodes and 28 output nodes; in architecture two, the letters were divided into six groups according to their similarities, e.g. ح، خ، ج، ع، غ all fall in one group. The problem was solved in two stages. In stage one, the letter was classified into one of the six groups using a single NN with 13 input nodes and six output nodes. In stage two, the letter was classified by one of six different NNs, giving one network per group.

4. ARABIC OCR SOFTWARE

Currently, there are four different software packages for recognising Arabic script. These packages are: TextPert 3.7 Arabic produced by CTA; ICRA 4.0 produced by Arab Scientific Software & Engineering Technologies; OmniPage produced by Caere Corporation; and Al-Qari' al-Ali 2.0 produced by the alAlamiah Software Company. All these packages are only used for recognising typeset and typewritten Arabic script. A number of critical evaluations for these packages can be either obtained from the Internet [128–132], or found on the proceedings of ICEMCO'96 [133,134]. Here they are recounted briefly:

1. *TextPert:* this runs on the Macintosh Arabic system, and it is easy to use. However, training new fonts is not possible. The recognition rate was approaching acceptable

standards when the program was tested on very good simple texts, but it virtually recognised nothing with more complicated fonts [128,130]. As a consequence, until the training feature is introduced to the software, its usage will be limited to those who only want to scan certain kinds of computer-generated documents.

2. *ICRA:* this runs under Microsoft Windows Arabic, and every typeface needs to be learned. The training process takes about one hour for each typeface. An experiment training this software with a number of Arabic magazines [129] showed that using these texts in their ordinary size gave disappointing results. Enlarging the text about 20% improved the recognition rate, which ranged between 90% and 99.7%.
3. *OmniPage:* this runs under standard Arabic Windows 3.1 and Arabic Windows for Workgroups 3.11, without customising. It integrates with standard Arabic word processors, including Microsoft Word, Microsoft Write and Accent. OmniPage does not require any training of fonts [131]. This is not essentially an advantage, since the program commits the same systematic error repeatedly without being able to learn from its failure.
4. *Al-Qari' al-Ali:* this program was first developed by Dr Rezvan of the Russian Academy of Science at the beginning of the 1990s [133]. It is a segmentation-based system which combines vector and bitmap analysis. The program is delivered with a standard set of modern computer fonts which can be recognised automatically. The main problem with this program is the considerable amount of time it takes to train for new fonts, especially typeset fonts with many ligatures. Versions 1.0 and 1.1 of this program run on al-Nawafidh al-Arabiya, which is an equivalent environment to Microsoft Windows but in Arabic, and version 2.0 runs under Microsoft Arabic Windows.

5. CONCLUSION

This paper has reviewed the previous work done in the field of Arabic text recognition. After discussing the characteristics of Arabic writing that influence the process of recognition, a general model for an Arabic text recognition system has been presented. This model was divided into five stages: image acquisition, preprocessing, segmentation, feature extraction, classification and recognition. Research methods at each stage were discussed and analysed using my implementation of various techniques to Arabic samples extracted from different sources.

The performance of four Arabic OCRs has been briefly reviewed. Although researchers has suggested various high-quality techniques to recognise multi-typewritten and unconstrained handwritten fonts, commercial Arabic OCR systems suffer from the lack of these two features. One reason behind this is that researchers do not implement the proposed techniques into real working environments. Researchers blame this on the absence of an Arabic text database, which can be used for evaluation purposes.

The various constraints imposed by OCR systems are not limited to Arabic. In English, OCRs impose that individual characters must not touch each other, text must be clearly printed, and the machine must be trained with any new font.

References

1. Govindan V, Shivaprasad A. Character recognition – a review. Pattern Recognition 1990; 23(7): 671–683
2. Downton A, Tregidgo R, Leedham C, Hendrawan. Recognition of handwritten British postal addresses. In: S Impedovo, J Simon (eds) From Pixels To Features III: Frontiers In Handwriting Recognition, Elsevier, 1992; 129–144
3. Dzuba G, Filatov A, Volgunin A. Handwritten zip code recognition. Proceeding of the Fourth International Conference on Document Analysis and Recognition, ICDAR'97, Ulm, Germany, 1997; 766–770
4. Cracknell C, Dowton A, Du L. An object oriented form description language and approach to handwritten form processing. Proceedings of the Fourth International Conference on Document Analysis and Recognition, ICDAR'97, Ulm, Germany, 1997; 180–184
5. Guillevic D, Suen C. Hmm-knn word recognition engine for bank cheque processing. Proceedings of the International Conference on Pattern Recognition, ICPR'98, Brisbane, Austrailia, 1998; 1526–1529
6. Guillevic D, Suen C. Recognition of legal amounts on bank cheques. Pattern Analysis and Applications 1998; 1(1): 28–41
7. Gatos B, Mantzaris S, Chandrinos K, Tsigris A, Perantonis S. Recognition of printed arabic text using neural networks. Proceedings of the Fourth International Conference on Document Analysis and Recognition, ICDAR'97, Ulm, Germany, 1997
8. Fadhel E, Bhattacharyya P. Application of steerable wavelet transform using neural network for signature verification. Pattern Analysis and Applications 1999; 2(2): 184–195
9. Shoukry A. Arabic character recognition state of the art. Proceedings of the 11th National Computer Conference, Dhahran, Saudi Arabia, 1989; 382–390
10. Amin A, Alsadoun H. Issues on arabic character recognition. Arabian Journal for Science and Engineering 1993; 18(3): 320–341
11. Albadr B, Mahmoud S. Survey and bibliography of arabic optical text recognition. Signal Processing 1995; 41: 49–77
12. Eldin AS, Nouh A. Arabic character recognition: A survey. Proceedings of the International Society for Optical Engineers, SPIE, 1998; 3386: 331–340
13. Ahmed P, Khan M. Computer recognition of arabic script based text: The state of art. The 4th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, 1994
14. Alohali Y, Ahmed P. A software environment for the development and evaluation of arabic character recognition systems. The 5th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, 1996
15. Amin A. Off-line arabic character recognition – a survey. Proceedings of the Fourth International Conference On Document Analysis and Recognition, Ulm, Germany, 1997; 596–599
16. Amin A. Off-line arabic character recognition: The state of the art. Pattern Recognition 1998; 31(5): 517–530
17. Parhami B, Taraghi M. Automatic recognition of printed farsi texts. Pattern Recognition 1981; 14(6): 395–403
18. Gérard G. Building a kurdish language corpus: An overview of the technical problems. Proceedings of the 6th International

Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, 1998; 5.3.1–5.3.11
19. Manaf M, Hamdan A, Yusof M. Development of a system to recognise handwritten jawi scripts and conversion of jawi scripts to rumi scripts. Proceedings of the 6th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, 1998; 5.6.1–5.6.10
20. Elsadany T, Hashish M. An arabic morphological system. IBM Systems Journal 1989; 28(4): 600–612
21. Khorsheed MS. Automatic Recognition Of Words In Arabic Manuscripts. PhD thesis, University Of Cambridge, June 2000. (Also available as University of Cambridge, Computer Laboratory Technical Report No. 495.)
22. Alemami S, Usher M. On-line recognition of handwritten arabic characters. IEEE Trans Pattern Analysis and Machine Intelligence 1990; 12(7): 704–710
23. Alimi A. An evolutionary neuro-fuzzy approach to recognise on-line arabic handwriting. Proceedings of the Fourth International Conference on Document Analysis and Recognition, ICDAR'97, Ulm, Germany, August 1997; 382–386
24. Wakahara T, Murase H, Odaka K. On-line handwritten recognition. Proc IEEE 1992; 80(7): 1181–1194
25. Almuallim H, Yamaguchi S. A method of recognition of arabic cursive handwriting. IEEE Trans Pattern Analysis and Machine Intelligence 1987; 9(5): 715–722
26. Nurul-Ula A, Nouh A. Automatic recognition of arabic character using logic statements. Journal of Engineering Sciences, King Saud University 1988; 14(2): 343–352
27. Nurul-Ula A, Nouh A. Automatic recognition of arabic character using logic statements. Journal of Engineering Sciences, King Saud University 1988; 14(2): 353–362
28. Elkhaly F, Ahmed MS. Machine recognition of optically captured machine printed arabic text. Pattern Recognition 1990; 23(11): 1207–1214
29. Goraine H, Usher M, Alemami S. Off-line arabic character recognition. IEEE Computer Journal July 1992; 71–74
30. Mahmoud S, Abuhaiba I, Green R. Skeletonization of arabic characters using clustering based skeletonization algorithm. Pattern Recognition 1991; 24(5): 453–464
31. Amin A, Alfedaghi S. Machine recognition of printed arabic text utilizing natural language morphology. Int Journal Man-Machine Studies 1991; 35: 769–788
32. Mahmoud S. Arabic characters recognition using fourier descriptors and character contour encoding. Pattern Recognition 1994; 27(6): 815–824
33. Yarman-Vural F, Atici A. A heuristic algorithm for optical character recognition of arabic script. Proceedings of the Visual Communications and Image Processing 1996; 2727: 725–736
34. Pavlidis T. Algorithms For Graphics And Image Processing. Computer Science Press, 1982
35. Gonzalez R, Woods R. Digital Image Processing. Addison-Wesley, 1992
36. Parker JR. Algorithms For Image Processing and Computer Vision. John Wiley & Sons, 1997
37. Fakir M, Sodeyama C. Machine recognition of arabic printed scripts by dynamic programming matching method. IEICE Trans Information and Systems 1993; 76(2): 235–245
38. Jiang H, Han C, Fan K. A fast approach to the detection and correction of skew documents. Pattern Recognition Letters 1997; 18(7): 675–686
39. Baird H. Anatomy of a versatile page reader. Proc IEEE 1992; 80(7): 1059–1065
40. Srihari S, Lam S, Govindaraju V, Srihari R, Hull J. Document understanding: Research direction. Technical Report, Center of Excellence For Document Analysis and Recognition, May 1992
41. Amin A, Masini G. Machine recognition of multi font printed arabic texts. Proceedings of the 8th International Joint Conference on Pattern Recognition, Paris, France, 1986; 392–395
42. Amin A, Mari J. Machine recognition and correction of printed arabic text. IEEE Trans Systems, Man and Cybernetics 1989; 19(5): 1300–1306
43. Fehri M, Ahmed MB. A new approach to arabic character recognition in multi-font document. The 4th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, 1994
44. Fehri M, Ahmed MB. An optical font recognising method for arabic texts. Proceedings of the 6th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, 1998; 5.15.1–5.15.7
45. Altuwaijri M, Bayoumi M. Arabic text recognition using neural networks. Proceedings of IEEE International Symposium on Circuits and Systems, London, UK, 1994; 415–418
46. Abuhaiba I, Holt M, Datta S. Recognition of off-line cursive handwriting. Computer Vision and Image Understanding 1998; 64(1): 19–38
47. Kim G, Govindaraju V. Efficient chain code based image manipulation for handwritten word recognition. Proceedings of the International Society for Optical Engineers, SPIE, 1996; vol. 2660: 262–272
48. Kim G, Govindaraju V. A lexicon driven approach to handwritten word recognition for real-time applications. IEEE Trans Pattern Analysis and Machine Intelligence, 1997; 19(4): 367–379
49. Guo Z, Hall R. Fast fully parallel thinning algorithms. Computer Vision, Graphics and Image Processing, 1992; 55(3): 317–328
50. Jang B, Chin R. One-pass parallel thinning analysis, properties, and quantitative evaluation. IEEE Trans Pattern Analysis and Machine Intelligence 1992; 14(11): 1129–1140
51. Yu S, Tsai W. A new thinning algorithm for grayscale images by the relaxation technique. Pattern Recognition 1990; 23(10): 1067–1076
52. Larmagnac J. Thinning and line segmentation by line following technique. Proceedings of the International Society for Optical Engineers, SPIE 1998; 3305: 210–219
53. Jain A. Fundamentals Of Digital Image Processing. Prentice Hall, 1989
54. Lam L, Lee S, Suen C. Thinning methodologies – a comprehensive survey. IEEE Trans Pattern Analysis and Machine Intelligence 1992; 14(9): 869–885
55. Ritter G, Wilson J. Handbook Of Computer Vision Algorithms in Image Algebra. CRC Press, 1996
56. Fan K, Chen D, Wen M. Skeletonization of binary images with nonuniform width via bloack decomposition and contour vector matching. Pattern Recognition 1998; 31(7): 823–838
57. Altuwaijri M, Bayoumi M. Thinning algorithm for arabic characters using art2 neural network. IEEE Trans Circuits and Systems 1998; 45(2): 260–264
58. Malandain G, Fernandez-Vidal S. Euclidean skeletons. Image and Vision Computing 1998; 16(2): 317–327
59. Amin A, Masini G. Machine recognition of cursive arabic words. Proceedings of the Applications of Digital Image Processing IV, SPIE 1982; 359: 286–292
60. Elwakil M, Shoukry A. On-line recognition of handwritten isolated arabic characters. Proceedings of the Symposium on Computer Arabiazation, Riyadh, Saudi Arabia, 1987; 109–120
61. Elwakil M, Shoukry A. On-line recognition of isolated arabic characters. Pattern Recognition 1989; 22(2): 97–105
62. Elsheikh T, Eltaweel S. Real-time arabic handwritten character recognition. The Third International Conference on Image Processing and it's Applications, IEE, 1989; 212–216
63. Elsheikh T, Eltaweel G. Real-time arabic handwritten character recognition. Pattern Recognition 1990; 23(12): 1323–1332
64. Elsheikh T, Eltaweel S. Segmentation of handwritten arabic words. Proceedings of the 12th National Computer Conference, Riyadh, Saudi Arabia, 1990; 389–402

65. Abuhaiba I, Ahmed P. Restoration of temporal information in off-line arabic handwriting. Pattern Recognition 1993; 20(7): 1009–1017
66. Abdelazim H, Hashish M. Automatic recognition of handwritten hindi numerals. Proceedings of the 11th National Computer Conference, Dhahran, Saudi Arabia, 1989; 287–298
67. Said F, Yacoub R, Suen C. Recognition of english and arabic numerals using a dynamic number of hidden neurons. Proceedings of the Fifth International Conference on Document Analysis and Recognition, ICDAR'99, Bangalore, India, IAPR, September 1999
68. Nouh A, Sultan A, Tolba R. On feature extraction and selection for arabic character recognition. Arab Gulg Journal For Scientific Research 1984; 2(1): 329–346
69. Amin A, Alsadoun H, Fischer S. Hand-printed arabic character recognition system using an artificial network. Pattern Recognition 1996; 29(4): 663–675
70. Abuhaiba I, Mahmoud S. Fuzzy graphs to recognise handwritten arabic characters: Arabian Journal for Science and Engineering 1995; 20(1): 77–93
71. Amin A, Kaced A, Haton J, Mohr R. Handwritten arabic character recognition by the irac system. Proceedings of the 5th International Conference on Pattern Recognition, Florida, US, 1980; 729–731
72. Amin A. Arabic handwriting recognition and understanding. Proceedings of the Computer Processing and Transmission of the Arabic Language Workshop, Kuwait, 1985; 1–37.
73. Saadallah S, Yagu S. Design of an arabic character reading machine. Workshop on Computer Processing and Transmission of the Arabic Language, Kuwait, 1985
74. Amin A. OCR of arabic text. The 4th International Conference on Pattern Recognition, Cambridge, UK, 1988; 616–625
75. Alyousefi H, Upda S. Recognition of arabic characters. IEEE Trans Pattern Analysis and Machine Intelligence 1992; 14(8): 853–857
76. Abdelazim H, Hashish M. Arabic reading machine. The 10th National Computer Conference, King Abdul Aziz University, 1988; 733–744
77. Tolba M, Shaddad E. On the automatic reading of printed arabic characters. Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Los Angeles, CA, 1990; 496–498
78. Motawa D, Amin A, Sabourin R. Segmentation of arabic cursive script. Proceeding of the Fifth International Conference on Document Analysis and Recognition, ICDAR'99, Bangalore, India, September 1999; 625–628
79. Hassibi K. Machine-printed arabic ocr using neural networks. The 4th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1994
80. Allam M. Arabic character recognition. Proceeding of the International Society for Optical Engineers, SPIE, 1994; 2181: 351–359
81. Emam A, Alkhatib H, Ismail M, Korany E. Character recognition of Arabic Script. The 4th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1994
82. Allam M. Segmentation vs. segmentation-free for recognising Arabic text. Proceedings of the International Society for Optical Engineers, SPIE 1995; 2422: 228–235
83. Elsheikh T, Guindi R. Computer recognition of arabic cursive scripts. Pattern Recognition 1988; 21(4): 293–302
84. Kavianifar M, Amin A. Preprocessing and structural feature extraction for a multi-fonts Arabic/Persian OCR. Proceedings of the Fifth International Conference on Document Analysis and Recognition, ICDAR'99, Bangalore, India, September 1999
85. Goraine H, Usher M. Printed arabic text recognition. The 4th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1994
86. Zahour A, Taconet B, Faure A. A new method for recognition of arabic cursive scripts. The 1st International Conference on Document Analysis and Pattern Recognition 1991; 454–462
87. Abuhaiba I, Mahmoud S, Green R. Recognition of handwritten cursive arabic characters. IEEE Trans on Pattern Analysis and Machine Intelligence 1994; 16(6): 664–671
88. Khorsheed MS, Clocksin WF. Off-line arabic word recognition using a hidden markov model. Statistical Methods for Image Processing – A Satellite Conference of the 52nd ISI Session in Helsinki, Uppsala, Sweden, August 1999
89. Khorsheed MS, Clocksin WF. Structural features of cursive Arabic script.' Proceedings of the Tenth British Machine Vision Conference, Nottingham, UK, 1999; 2: 422–431
90. Abdelazim H, Mousa A, Saleh Y, Hashish M. Arabic text recognition using partial observation approach. Proceedings of the 12th National Computer Conference, Riyadh, Saudi Arabia, King Saud University, 1990; 427–437
91. Ramsis R, Eldabi S, Kamel A. Arabic character recognition system. Technical Report, IBM Kuwait Scientific Cenetr, 1988
92. Eldabi S, Ramsis R, Kamel A. Arabic character recognition system: A statistical approach for recognizing cursive typewritten text. Pattern Recognition 1990; 23(5): 485–495
93. Mosfeq R. Arabic character recognition using integrated segmentation and recognition. The 5th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1996
94. Zhao X, Srihari S. Word recognition using ideal word patterns. Proceedings of the International Society for Optical Engineers, SPIE, 1994; 2181: 24–34
95. Rabiner L, Juang B. Fundamentals of Speech Recognition. Prentice Hall, 1993
96. Erlandson E, Trenkle J, Vogt R. Word-level recognition of multifont arabic text using a feature vector matching approach. Proceedings of the International Society for Optical Engineers, SPIE, 1996; 2660: 63–70
97. Amin A. Recognition of printed arabic text using machine learning. Proceedings of the International Society for Optical Engineers, SPIE, 1998; 3305: 63–70
98. Kraus E, Dougherty E. Segmentation-free morphological character recognition. Proceedings of the International Society for Optical Engineers, SPIE, 1994; 2181: 14–23
99. Albadr B, Haralick R. Segmentation-free recognition of Arabic text. The 5th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1996
100. Makhoul J, LáPré C, Raphael C, Schwartz R, Zahao Y. Towards language-independent character recognition using speech recognition methods. The 5th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1996
101. Makhoul J, Schwartz R, Lapre C, Bazzi I. A script-independent methodology for optical character recognition. Pattern Recognition 1998; 31(9): 1285–1294
102. Bazzi I, Schwartz R, Makhoul J. An omnifont open-vocabulary ocr system for English and Arabic. IEEE Trans Pattern Analysis and Machine Intelligence 1999; 21(6): 495–504
103. Clocksin WF, Khorsheed MS. Word recognition in Arabic handwriting. Proceedings of International Conference on Artificial Intelligence Applications ICAIA'2000, Cairo, Egypt, February 2000
104. Khorsheed MS, Clocksin WF. Spectral features for Arabic word recognition. The IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP'2000, Istanbul, Turkey, June 2000
105. Simon J. Off-line cursive word recognition. Proc IEEE 1992; 80(7): 1150–1161
106. Amin A, Alsadoun H. Hand printed arabic character recognition system. Proceedings of the 12th International Conference on Pattern Recognition, IAPR, 1994; 536–539

107. Eldesouky A, Salem M, Elgwad AA, Arafat H. A handwritten Arabic character recognition technique for machine reader. Proceedings of the Third International Conference on Software Engineering for Real Time Systems, Cirencester, UK 1991; 212–216
108. Kondybaev N. Maximum entropy approach in automatic classification of symbolic images. The 4th International Conference and Exhibition on Multi-Lingual Computing, (Cambridge, UK), Cambridge University Press, 1994
109. Mahmoud M, Elhamalaway M, Fahmi A. A statistical approach for arabic character recognition. Proceedings of the 12th International Conference For Statistics and Computer Science, Cairo, Egypt, 1987; 243–250
110. Bouslama F. Neuro-fuzzy techniques in the recognition of written arabic characters. Proceedings of North American Fuzzy Information Process, Berkeley, CA, IEEE, 1996; 142–146
111. Sanossian H. An Arabic character recognition system using neural network. Proceedings of 1996 IEEE Signal Processing Society Workshop, Kyoto, Japan, IEEE, 1996; 340–348
112. Hu M. Visual pattern recognition by moment invariants. IRE Trans Information Theory 1962; 8: 179–187
113. Zaki F, Elkonyaly S, Elfattah AA, Enab Y. A new technique for arabic handwriting recognition. Proceedings of the 11th International Conference for Statistics and Computer Science, Cairo, Egypt, 1986; 171–180
114. Zahn C, Roskies R. Fourier descriptors for plane closed curves. IEEE Trans Computers 1972; 21(3): 269–282
115. Khorsheed MS, Clocksin WF. Segmentation-free word recognition for Arabic handwriting. The International Conference on Pattern Recognition ICPR'2000, Barcelona, Spain, September 2000
116. Freeman H. On the encoding of arbitrary geometric configurations. IRE Trans Electronic Computers 1961; EC-10: 260–268
117. Nouh A, Ula A, ElDin S. Algorithms for feature extraction: A case study for the arabic character recognition. The 10th National Computer Conference, King Abdul Aziz University, 1988; 653–666
118. Albadr B, Haralick R. Symbol recognition without prior segmentation. Proceedings of the International Society for Optical Engineers 1994; 2181: 303–314
119. Juang B, Rabiner L. Mixture autoregressive hidden markov models for speech signals. IEEE Trans Acoustics, Speech and Signal Processing 1985; 33: 1404–1413
120. Emam O, Hashish M. Application of hmm of isolated Arabic words. The 10th National Computer Conference, King Abdul Aziz University, 1988; 761–767
121. Rabiner L. A tutorial on HMM and selected applications in speech recognition. Proc IEEE 1989; 77: 257–286
122. Pao Y. Adaptive Pattern Recognition and Neural Networks. Addison-Wesley, 1989
123. Hosseini H, Bouzerdoum A. A system for arabic character recognition. Proceedings of Austrailian New Zealand International Information Systems Conference ANZIIS'94, Brisbane, Australia, 1994; 120–124
124. Amen R, Jambi K, Albideiwi I, Alaidarous A. Arabic character recognition. Technical Report, Computer Science Department, Faculty of Science, King Abdul Aziz University, 1996
125. Amin A, Mansoor W. Recognition of printed arabic text using neural networks. Proceedings of the Fourth International Conference on Document Analysis and Recognition, ICDAR'97, Ulm, Germany, IAPR, August 1997; 612–615
126. Auda H. Nearest neighbor classification of handwritten numerals using class probability matrices. Proceedings of the Seventh International Conference on Artificial Intelligence Applications, Cairo, Egypt, 1999; 175–183
127. Alazim HA. A hybrid fuzzy-neural approach to the recognition of arabic script. The 5th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1996
128. Bell J, Zemanek P. Test of two Arabic OCR programs. Arabic and Computer on The Internet, http:www.hffak.uib.no/smi/ksv/arabocr.html, 1994
129. Hoogland J. Arabic OCR. Reader on The Internet, http:leb.net/archives/reader, 1995
130. Bell J. Arabic OCR. Reader on The Internet, http:leb.net/archives/reader, 1995
131. Zemanek P. Arabic OCR – OmniPage. Reader on The Internet, http:leb.net/archives/reader, 1996
132. Hoogland J. Icemco and the day after. Reader on The Internet, http:leb.net/archives/reader, 1996
133. Bell J, Matveev A, Zemánek P. Arabic OCR with Al Alamiah's A-Qari' Al-Ali. The 5th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK), Cambridge University Press, 1996
134. Hoogland J. The use of OCR software of Arabic in order to create a text corpus of modern standard Arabic for lexicographic purposes. The 5th International Conference and Exhibition on Multi-Lingual Computing, Cambridge, UK, Cambridge University Press, 1996

Mohammad Khorsheed received the BSc and the MSc degrees from the Computer Engineering Dept. at King Saud University, Saudi Arabia in 1989 and 1994. In 1997, he joined the PhD program at the Computer Laboratory, University of Cambridge, UK. He was researching in recognising historical Arabic manuscripts using Hidden Marov Models. His research interests include speaker-independent Arabic speech recognition and intelligent control, especially, Neural Network and Fuzzy Logic based control systems.

Correspondence and offprint requests to: M.S. Khorsheed, P.O. Box 6917, Riyadh 11452, Saudi Arabia. Email: msmk2@cl.cam.ac.uk