Proceedings of the International Conference on Computer Science, Software Engineering, Information Technology, e-Business, and Applications (CSITeA'02). Foz do Iguazu, Brazil. June 6 – 8, 2002.

Segmenting Handwritten Arabic Text

Ramzi Haraty and Alaa Hamid
Lebanese American University
P.O. Box 13-5053 Chouran
Beirut, Lebanon 1120 2801
E-mail: rharaty@lau.edu.lb

Abstract

The segmentation and recognition of Arabic handwritten text has been an area of great interest in the past few years. However, a small number of research papers and reports have been published in this area. There are several major problems with Arabic handwritten text processing: Arabic is written cursively and many external objects are used such as dots, 'Hamza', 'Madda', and diacritic objects. In addition, Arabic characters have more than one shape according to their position inside a word. More than one character can also share the same horizontal space, creating vertically overlapping connected or disconnected blocks of characters. This makes the problem of segmentation of Arabic text into characters, and their classification even more difficult.

In this work a technique is presented that segments difficult handwritten Arabic text. A conventional algorithm is used for the initial segmentation of the text into connected blocks of characters. The algorithm then generates pre-segmentation points for these blocks. A neural network is subsequently used to verify the accuracy of these segmentation points. Another conventional algorithm uses the verified segmentation points and segments the connected blocks of characters. These characters can then be used as input to another neural network for classification.

1. Introduction

In spite of the extensive work done on the recognition of handwritten Latin and Asian languages text and the excellent results obtained in Latin text, a few research papers and reports have been published in the area of handwritten Arabic text recognition. This is because the recognition of handwritten Arabic text is considerably harder than that of Latin text due to a number of reasons:

- Arabic is written cursively, i.e., more than one character can be written connected to each other, forming a block of characters (BC).
- Arabic uses many types of external objects, such as dots, '*Hamza*', '*Madda*', and diacritic objects. These make the task of line separation and segmenting text into BCs more difficult.
- Arabic characters can have more than one shape according to their position inside a BC: initial, middle, final, or standalone, as shown in Figure 1.

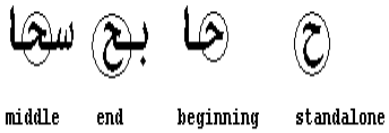

Figure 1. The shapes the character ح takes according to its position inside a word.

- Different writers and the same writer under different conditions will write some Arabic characters in completely different ways, as shown in Figure 2.

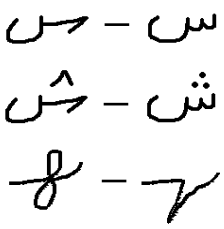

Figure 2. Three characters written in completely different ways.

- Characters that do not touch each other but occupy a shared horizontal space increase the difficulty of BC segmentation.
- Arabic uses many ligatures, especially in handwritten text. Ligatures are characters that occupy a shared horizontal space creating vertically overlapping connected or disconnected BCs.

Other problems of handwritten text recognition include the presence of lines, non-character objects, and noise or 'salt-and-pepper' in the scanned image. Characters can also be written in many different sizes, writing instruments (varying thickness and stroke quality), and slants (causing character shearing along the horizontal axis).

Artificial Neural Networks (ANNs) have been successfully applied to many areas of pattern recognition, especially in the field of character recognition. Some researchers have used conventional methods for segmentation and recognition, while others have used ANN based methods for the character recognition process [3].

This research describes a hybrid method to segment Arabic handwritten text. The method contains two main components. The first is a heuristic algorithm, which is responsible for scanning handwritten text, extracting blocks of connected characters (BCs), and then extracting features to be used in the second component. It is also responsible for generating pre-segmentation points, which are validated by the second component, the ANN. The ANN verifies whether all the segmentation points found are correct or incorrect

The remainder of the paper is divided into 4 sections. Section 2 describes the proposed techniques. Section 3 provides a discussion of current and future work, and a conclusion is presented in Section 4.

2. Proposed Techniques

There are a number of steps that need to be taken before handwritten text on an envelope or a page to be recognized by a computer. These include scanning, binarization, segmentation, and character recognition.

2.1 Scanning

Since there is no standard benchmark database for Arabic handwritten text, samples were acquired randomly from various students and faculty members around the university. They were asked to write down their own mailing address on A4 sized paper. These addresses were then scanned using an Agfa SnapScan 1212p scanner at 150 pixels per inch and saved in monochrome Windows Bitmap (BMP) format. The images had different sizes ranging from 260 x 140 pixels to 1200 x 400 pixels. 360 addresses have been collected consisting of about 4000 words or 9000 BCs.

2.2 Binarization

After the images were acquired, they were converted into monochrome bitmap form. Before any segmentation or processing could take place, it was then necessary to convert the images into binary representations. A heuristic algorithm generated a matrix of ones (1's) and zeros (0's) for each image. Each black pixel was represented with a 1 and each white pixel with a 0. In this form, segmentation and preprocessing could take place more easily.

2.3 Segmentation Using a Heuristic Algorithm

Segmentation deals with the process of attempting to isolate classifiable units from the handwritten text image. It plays an important role in the overall process of character recognition. It has been argued that the overall success rate of any recognition system can be expressed as a product of 2 factors: the success rate of the segmentor and the success rate of the recognizer [1].

2.3.1 Extracting Blocks of Characters

BC extraction is the first step of the segmentation phase. It was necessary to recursively extract connected BCs. Furthermore, dots, diacritics, and other external objects are used heavily in Arabic. This makes the task of linking these external objects to main character objects, to create BCs, a very difficult process. A heuristic algorithm was implemented with 94% accuracy, scanned the whole binary matrix of the image and performed the following steps:

Step 1. Before any processing could occur, invalid isolated black pixels, or '*pepper*' were removed. A black pixel was identified as pepper and discarded if it had a maximum of one black neighbor pixel.

Step 2. Recursively, identify each group of connected black pixels as an object.
Step 3. Classify objects as child or parent ones. If the weight, or black pixel density, of the object is less than half of average weight of all objects, then it is marked as a child. Otherwise, it is marked as a parent.
Step 4. For each child object, determine the distance to all parents.
Step 5. For each child object, determine the distance range, *R,* for all possible parents, which is equal to 150% of the distance to the nearest parent. This formula was found by experimentation and yielded the best results.
Step 6. Merge all children to all parents who are at a maximum distance of *R*. This will lead to child objects being duplicated and merged to more than one parent. This is done to solve the problem of children that are located near objects other than their parent ones.

Higher accuracy couldn't be achieved because of external or child objects that were too far away from their parent objects and too near to other main objects.

2.3.2 Feature Extraction and Possible Segmentation Point Generation

The heuristic BC segmentation algorithm contained two components. The first one scanned the BC looking for important features to identify possible segmentation points. Such features include minimas or ligatures between letters, common in handwritten cursive text. In many cases these ligatures are the ideal segmentation points. In other cases, more features are needed to determine a valid segmentation point like holes, endpoints, corners, and fork points. A complete list of extracted features is shown in Table 1. Skeletonization of the image was required in order to extract most of these features. Skeletonization or thinning is an image-processing step that reduces BCs to their skeletons, i.e., transforming characters into arc segments one pixel thick. The thinning algorithm of [5] produced acceptable results with few enhancements and modifications.

The objective of the second component was to over-segment all the BCs based on the features extracted in the first component. The distribution of the proposed segmentation points is also taken into consideration based on the average character width in a BC. The average character width is determined from the average word height. It is understood that the width of a character in most cases is less than its height. Therefore as an approximate character width estimate, a percentage of the average word height is used to provide a rough solution.

Table 1. Major features extracted for each column of BC matrix.

Feature	Attributes	Description	Attr. Type	Attr. Value
Image width and height		Image width and height in pixels.	Discrete	$(0,\infty)$
Black pixel density	Black pixel density / height	Number of black pixels in the column divided by the image height.	Cont.	[0,1]
	Density minima	Does the column cross a density minimum?	Binary	0 or 1
	Density maxima	Does the column cross a density maximum?	Binary	0 or 1
Transitions	Number of transitions crossed	Scan the image vertically and count the number of foreground-background and background-foreground transitions crossed by column.	Discrete	[0,ht.]
Holes	Number of holes crossed	Count the number of holes (or islands of white pixels completely surrounded by black pixels) crossed by column.	Discrete	[0,ht.]
	Total hole densities / height	Total number of hole pixels crossed by column divided by the image height.	Cont.	[0,1]
Endpoints	Number of endpoints crossed	Number of endpoints crossed by column.	Discrete	[0,ht.]
Corner points	Number of corners crossed	Number of corner points crossed by column.	Discrete	[0,ht.]
Fork points	Number of fork points crossed	Number of fork points crossed by column.	Discrete	[0,ht.]
Relative index of column in image		Index of column divided by image width.	Cont.	[0,1]
Upper and lower contours	Upper and lower contour index / height	Index of upper most and lower most black pixel crossed by column divided by image height.	Cont.	[0,1]
	Upper and lower contour minima or maxima	Does the column cross an upper or lower contour minima or maxima?	Binary	0 or 1
Feature relationships	Index of nearest left and right feature / 100	Index of nearest left and right feature in a 100-pixel width range.	Cont.	[0,1]

2.4 Segmentation Using an Artificial Neural Network

To train the ANN with both accurate and erroneous segmentation points, the output from the heuristic segmentation algorithm was used. It was necessary to manually separate the points generated by the algorithm into valid and invalid segmentation points and save them to a file together with the extracted set of features and desired output for each point.

2.4.1 ANN Architecture

A generalized feedforward neural network was used to validate the accuracy of the proposed segmentation points. This neural network is a generalization of the multi-layer perceptron (MLP) such that each layer feeds forward to all subsequent layers. In theory, a MLP can solve any problem that a generalized feedforward network can solve. In practice, however, generalized feedforward networks often solve the problem much more efficiently [2], [4].

The first criterion is to use an efficient technique for training. Training of the network was done using the error back-propagation technique. This technique gets its name from the fact that the network is presented with an input pattern, for which an output pattern is calculated. Then the error between the desired and actual output can be determined, and passed backwards through the network. Based on these errors, weight adaptations are calculated, and errors are passed to a previous layer, continuing until the first layer is reached. The error is thus propagated back through the network. A training set of 48,000 exemplars was used.

The second criterion is the network size. The number of PEs in a hidden layer is associated with the mapping ability of the network. The larger the number, the more powerful the network is. However, if one

continues to increase the network size, there is a point where the generalization gets worse. This is due to the fact that we may be over-fitting the training set, so when the network works with patterns that it has never seen before the response is unpredictable.

There is no rule of thumb to determine good network architecture just from the number of inputs and outputs. It depends critically on the number of training cases, the amount of noise, and the complexity of the function or classification the network is supposed to learn. There are cases with one input and one output that require thousands of hidden units, and cases with a thousand inputs and a thousand outputs that require only one hidden unit, or none at all. To solve this issue many different networks with different number of hidden units were tried at first, starting with the smallest possible number of PEs. The generalization error for each one was estimated, and the network with the minimum estimated generalization error that learned best to identify correct segmentation points was chosen.

The best ANN architecture reached consisted of 52 inputs, 1 output, and 4 hidden layers. The 52 inputs were feature attributes of a pre-segmentation point and the output was the validity of the point. The third criterion is the termination of the training process. Cross validation was used, which is a highly recommended method for stopping network training. It monitors the mean square error on an independent set of data and stops training when this error begins to increase. This is considered to be the point of best generalization. The cross-validation set consisted of 10,000 exemplars. The best ANN architecture reached consisted of 52 inputs (extracted features), 1 output, and 4 hidden layers.

2.4.2 Experimental Results

The output range of the ANN was between –0.9 and +0.9. A positive value indicated that a point is a valid segmentation point; a negative value indicated that a point should be ignored.

A heuristic algorithm checked the results of the segmentation ANN on a test set of 10,000 exemplars. The algorithm defined the segmentation of a BC as correct when each known segmentation point was covered by an approved segmentation point by the ANN. Adequate coverage of a segmentation point is achieved when the distance from the known segmentation point to the closest approved point is less than 15% of the average character size.

There were some objects that were impossible to segment in handwritten Arabic text. Every 100 BCs of the collected data contain 10.16 un-segmentable ligatures and 13.02 characters with miss-located external objects. In addition, 9.24 س and ش characters occur in every 100 BCs, which are almost always un-segmentable. Other miscellaneous un-segmentable BCs include characters like the letter ض, which is always segmented into the letters ع or م, and ن.

Table 2 shows the results of the segmentation ANNs trained on the 48,000 training exemplars and tested on 10,000 exemplars. Many experiments were performed varying settings such as the network type, the number of hidden layers and the number of processing elements in each layer. For each experiment the number of inputs remained the same: 52 input features for each column.

Table 2. Segmentation ANN results using 48,000 training exemplars, tested on 10,000 exemplars.

ANN Architecture			MSE	Correct Points		Incorrect Points	
Network Type	Hidden Layers			Invalid points	Valid points	Invalid points marked as valid	Valid points marked as invalid
	No	PEs					
Feedforward MLP	2	41-27	0.81	1354 (13.54%)		8646 (86.46%)	
Feedforward MLP	3	41-27-20	0.72	2684 (26.84%		7316 (73.16%)	
Feedforward MLP	4	41-27-20-16	0.41	5311 (53.11%)		4689 (46.89%)	
				3767 (37.67%)	1544 (15.44%)	3326 (33.26%)	1363 (13.63%)
Feedforward MLP	5	41-27-20-16-13	0.56	4225 (42.25%)		5775 (57.75%)	
Feedforward MLP	6	41-27-20-16-13-11	0.50	4633 (46.33%)		5367 (53.67%)	
MLP	5	55-31-19-15-11	0.82	1332 (13.32%)		8668 (86.68%)	
MLP	7	55-31-19-15-13-11-9	0.59	3854 (38.54%)		6146 (61.46%)	
MLP	9	55-31-19-15-13-11-9-7-6	0.73	2336 (23.36%)		7664 (76.64%)	

The ANN architecture performed best in identifying correct segmentation points and discarding incorrect ones. The minimum MSE achieved was 0.41. The ANN was able to identify the accuracy of 5,311 points out of the 10,000-point testing set. Of the correctly identified points, 3, 767 were invalid segmentation points and 1,544 were valid segmentation points.

The ANN incorrectly identified 4,689 points. 3,326 of these points were invalid segmentation points marked as valid, and 1,363 were valid points marked as invalid. It should be noted that the majority of incorrectly identified points were invalid segmentation points marked as valid, which implies that the ANN over-segmented the handwritten BCs.

After studying the distribution of the ANN results over the range [-0.9,+0.9], it was noted that the majority of these 3,326 points were found in the [0,+0.5] range, as illustrated in Figure 3.

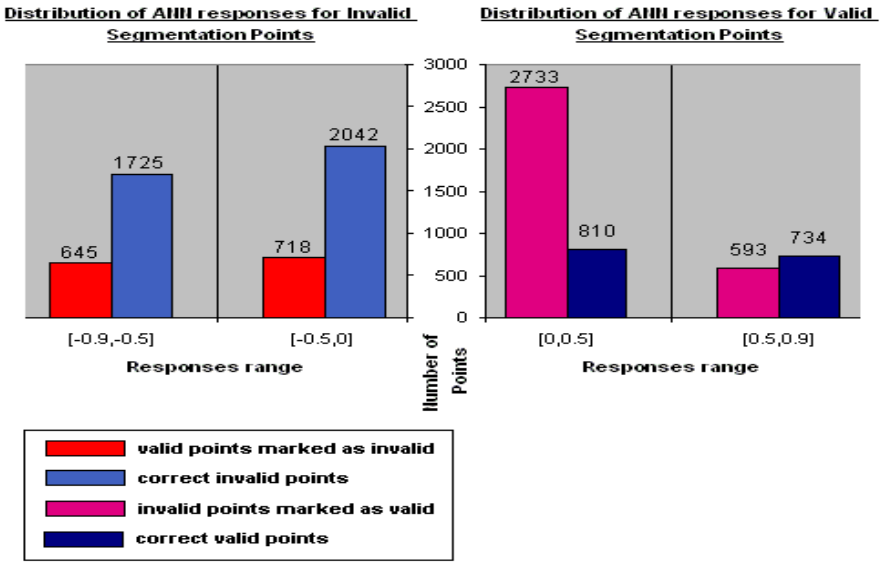

Figure 3. Distribution of ANN responses.

To decrease the number of incorrectly identified segmentation points, a threshold value was applied to the ANN output. A module was implemented which checked the ANN responses against a 0.5 threshold. Responses between 0 and +0.5 were therefore rejected. It should be noted that these rejected patterns also include 810 correctly identified valid segmentation points. However, the number of incorrectly identified points, 2,733, in the (0,0.5) range is much greater than the correctly identified points. Table 3 shows the results after rejecting these patterns.

Table 3. ANN results after rejecting patterns with responses in the (0,0.5) range.

Correct Points		**Incorrect Points**		**Rejected Points**	
Invalid points	**Valid points**	**Invalid points marked as valid**	**Valid points marked as invalid**	**Invalid points marked as valid**	**Valid points**
4501 (45.01%)		1956 (19.56%)		3543 (35.43%)	
3767 (37.67%)	734 (7.34%)	593 (5.93%)	1363 (13.63%)	2733 (27.33%)	810 (8.10%)

3. Conclusion and Future Work

A heuristic segmentation technique used in conjunction with a generalized feed-forward multi-layer neural network has been presented in this paper. It was used to segment difficult handwritten Arabic text, producing promising results.

The segmentation program over-segmented the BCs it was presented with. This allowed the segmentation ANN to discard improper segmentation points and leave accurate ones. Overall the whole process was very successful, however some limitations still exist. The segmentation phase proved to be successful in vertical segmentation of connected blocks of characters. However, in Arabic handwritten text, a lot of characters share the same horizontal space. A major limitation of the presented technique is that it couldn't segment horizontally these overlapping characters.

Another problem is that there are a lot of handwritten characters that can be segmented and classified into two or more different classes depending on whether you look at them separately, or in a word, or even in a sentence. In other words, character segmentation and classification, especially handwritten Arabic characters, depends largely on contextual information, and not only on the topographic features extracted from these characters. Arabic handwriting recognition is a difficult problem and it is not yet realistic to expect systems to achieve an acceptable accuracy in large vocabularies or where contextual information is of little use.

In future work, the segmentation technique will be improved in a number of ways. Firstly, the heuristic component of the segmentation system will need to be enhanced further. Originally, one of the main aims of the heuristic algorithm was to keep the number of incorrect segmentation points to a minimum, so that errors and processing time could be reduced. As a result, under-segmentation was noticeable in some BCs. Looking for more features or possibly enhancing the current feature extraction methods can solve this problem.

References

[1] B. Al-Badr and R. Haralick, "A Segmentation-Free Approach to Text Recognition with Application to Arabic Text", *International Journal on Document Analysis and Recognition*, vol. 1, no. 3, pp. 147-166, Dec. 1998.

[2] L. Almeida, "*Multilayer Perceptrons, Handbook of Neural Computation*", IOP Publishing Ltd. and Oxford University Press, 1997.

[3] M. Blumenstein and B.Verma, "Recent Achievements in Off-Line Handwriting Recognition Systems", in Proc. *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA '98)*, Melbourne, Australia, 1998, pp. 27-33.

[4] M. Blumenstein and B. Verma, "Neural Based Solution for the Segmentation and Recognition of Difficult Handwritten Words", in Proc. *Fifth International Conference on Document Analysis and Recognition*, 1998.

[5] A. Rosenfeld, "A Simple Parallel Algorithm for Skeletonization", http://www.cs.mcgill.ca/~laleh/rosen_alg.html. 2002.