Automatic Recognition Of Typewritten Arabic Characters Using Zernike Moments As A Feature Extractor

Maged Mohamed Mahmoud Fahmy

Computer Science Department
College of Science
University of Bahrain
Isa Town
STATE OF BAHRAIN
Email: Mfahmy@Sci.Uob.Bh

Haytham El-Messiry

Informatics Research Institute
Mubarak City for Scientific Research and Technological Applications
Mansheyat Al-Olama
Alexandria
EGYPT
E-mail: hmessiry@mcsrta.egnet.net

Abstract: This paper introduces a statistical approach for the recognition of Arabic characters. As a first step, the character is segmented into two parts. The first part is dots and Hamza, and the second part is the character body. This reduces the number of character classes from 28 to 18. The second step is the extracting of character features. This is achieved using Zernike moments that are used for mapping character image onto a set of complex orthogonal polynomials. The third step is character classification. A Backpropagation neural network based approach is used for the classification of Arabic characters represented by invariant features (Zernike moments based features). The approach has been evaluated using printed characters, obtained from scanned documents, with different fonts and sizes from a large database collected for this purpose. The overall recognition rate was 99.3 % for this system.

Keywords: Arabic characters, Zernike moments, Backpropagation

Dr **Maged Mohamed Mahmoud Fahmy** is currently working as Assistant Professor, Computer Science Department, College of Science, University of Bahrain. Previously he worked as researcher at Informatics Research Institute, and Mubarak City for Scientific Research and Technological Application, Alexandria, Egypt. He got the BSc. degree in Electronic Engineering, the MSc. degree in Computer Security from the Alexandria University in 1986. He got the Ph.D Engineering in Computer Vision from University of Newcastle-upon-Tyne, the UK, in 1994. He supervised many Master degrees. His fields of interest are in artificial intelligence, computer vision (image processing and neural network), computer networks.

Mr **Haytham El-Messiry** is working as research assistant in Mubarak City for Scientific Research in Alexandria, Egypt. He got his BSc. and MSc. in Computer Science from Alexandria University. Currently he is preparing his PH.D in Computers in Germany. His field of study is artificial intelligence.

1. Introduction

The subject of character recognition has been paid considerable attention in recent years due to the advancement of the automation process. It can improve the interaction between man and machine in many applications such as office automation, check verification, and a large variety of banking, business and data entry. Several methods for recognizing Latin, Chinese characters have been proposed, while the recognition of Arabic characters has been relatively sparse [1]. Techniques developed for classifying characters in other languages cannot be used for recognizing Arabic characters due to the differences in structure. Arabic is cursive in general. Therefore, the recognition rate of Arabic characters is lower than that of disconnected characters, such as printed English. Most of the techniques proposed to date for recognizing Arabic characters have relied on structural and topographic approaches. The problems arising in any OCR system are not only related with data acquisition, but also with the nature of the input.

The feature extraction aspect of image analysis seeks to identify inherent characteristics, or features, found within an image. In Optical Character Recognition problem, these characteristics are used to describe the character, prior to the subsequent task of classification. The extracted character versions may have one, or more, of the variations such as scaling, rotation, and translation. This means that characters might differ in their size, some of them might be rotated, or shifted from the y-axis. In order to recognize many variations of the same character, features that are invariant to certain transformations on the

character need to be used. Invariants are features which have approximately the same values for samples of the same character that are, for example, translated, scaled, rotated, stretched, skewed, or mirrored. However, not all variations among characters from the same character class (e.g. noise or absence or presence of serifs) can be modeled using invariants. Size and translation invariance is easily achieved. The segmentation of individual characters can itself provide estimates of size and location, but the feature extraction method may often provide more accurate estimates. Rotation invariance is important if the characters to be recognized can occur in any orientation. However, if all the characters are expected to have the same rotation, then rotation-variant features should be used to distinguish between such characters as 6 and 9, and n, u. The application of moments, as a feature extractor, provides a method for describing the properties of an object in terms of its area, position, orientation, and other precisely defined parameters.

The algorithm used in this research consists of many steps. It starts with dealing with the character represented by a binary image. The character is then segmented into Primary and Secondary Parts according to character histogram. The Primary Part that represents the body of the character is preprocessed to be ready for the feature extraction phase. The character body is then fit into the unit disk with radius equal to the maximum radius calculated from the preprocessing phase [2]. Features are then extracted by calculating the Zernike moments and getting the 47 features vector that represents each character. The feature vector is entered as an input to one of the feedforward backpropagation neural networks, used as the classifier in this research. These Neural Networks were trained to classify the primary parts of the character. The Recognizer system is composed of four Neural Networks according to the position of the primary part within the word, i.e. if Isolated, Beginning, Middle, and End.

2. Arabic Font Characteristics

The characteristics of Arabic font may be summarized into seven general definitions [3]:

1. The direction of Arabic writing is from right to left, while the Indian numerals are written from left to right.
2. The cross point, branch points inside character, and the connection points between characters always fall near the writing line (the midline of writing); this may provide useful contextual information.
3. Most of the characters are formed by curves and loops. Loops are usually drawn in clockwise direction.
4. Concatenation of isolated Arabic characters is an unacceptable way of Arabic writing. The property of **CURSIVENESS** is a characteristic to all typewritten Arabic fonts and represents the main challenge in the Automatic Recognition of Arabic Text.
5. Most of the Arabic characters have four different shapes depending on the position of the character in a word (beginning, middle, end, and isolated), as shown in Figure 1.
6. Some of the Arabic characters have dots above or below the main body (Figure 2). Different characters may have the same body but distinguish themselves by the number and the positions of dots. These dots could be single, in pairs, or triple, and usually referred to as *Diacritic signs*.
7. Overlapping property for some characters, this appears in certain fonts (e.g. Traditional Arabic) as shown in Figure 3.

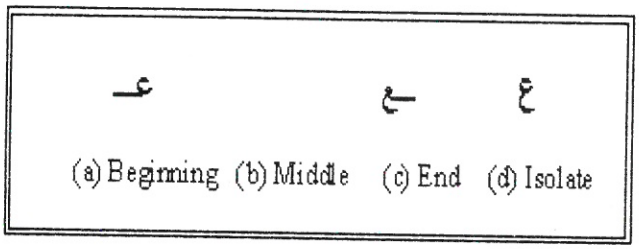


Figure. 1: Different shapes of the Arabic letter *Aeen* (ع)

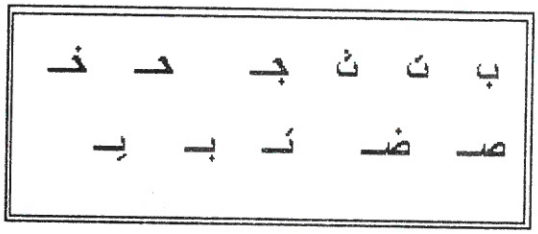


Figure. 2: Arabic characters differing accordi position and number of associate dots

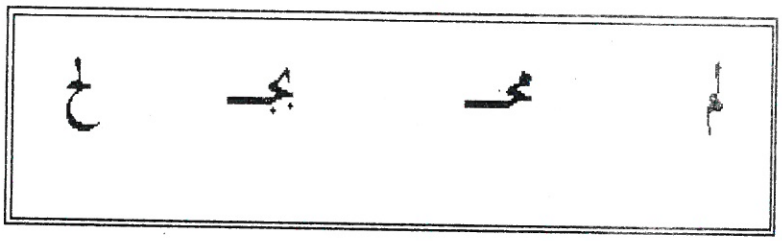


Figure. 3: Examples of Overlapping Arabic characters.

When designing an Arabic optical character recognition system, one should keep in mind problems that may arise. Some of these problems are met with during data acquisition, to be currently generated by random combinations of dilation, translation, skews, and rotation contaminated by superimposed and deleted regions of a varying extent. For a certain font and for the same characters the pixel distribution is not always the same. Other problems may derive from the nature of the input: the input document may not be a properly spaced plain text document. Single spaced documents can cause problems to OCR systems. This problem is often referred to as *line mingling*. Also documents may be composed of multiple text columns. Another problem is the presence of graphics or images within a document. Automatic discrimination of text from graphics and images should be provided by an OCR system.

The second category of problems arises from the variability of Arabic fonts. In an Arabic text, the number of shapes that a certain character can take is relatively greater than in case of a Latin character. The Arabic character shapes within the domain of all fonts are disjoint. This may lead to the fact that a certain character in a specific font can be interpreted as a different character in another font. The need of setting up rules for Arabic font design to ease the process of Arabic text recognition has only recently been recognized. In Typesetted Arabic, certain character pairs may be combined together to form another different character. The resulting character is often referred to as a ligature. The only mandatory ligature is the (Lam Alef). Other ligatures such as (LamHah, BehMeem, etc.) are optional. The presence of ligatures in a typewritten text renders segmentation and recognition even more complex.

3. Preprocessing Segmentation and Feature Extraction

It has been known that the ancient Arabic alphabetic letters were written without dots or diacritics, and later on the dots and diacritics have been used to ease dealing with characters. If this way of thinking is used, the number of classes of characters will decrease from 100 to 64. The explanation for this is that there are characters having the same body but differing in the number or the position of dots, either above or below the body of the character (e.g. beh (ب), teh (ت)). From this moment on our method will apply. First the model uses any segmentation technique that produces the character as two separate parts: The Primary part (i.e. body of the character) and Secondary part (i.e. dots and hamza). Amin's segmentation algorithm [4] has been used, which depends on the histogram of the word. After segmenting the word into characters the Secondary part is also extracted and recognized. This is done using the Yousefi and Udpa algorithm [5].

Feature Extraction aspect of image analysis aims to identify inherent characteristics, or features, of characters found within an image. These characteristics are used to describe the character, prior to the subsequent task of classification. The application of moments provides a method for describing the properties of an object in terms of its area, position, orientation, and other precisely defined parameters.

Choosing a feature extractor that is invariant to Scaling, Translation, and Rotation, helps minimize the preprocessing phase time. The description of moments as a feature extractor method and the use of Zernike moments as feature extractor, a step towards recognizing Arabic characters, were explained in [6]. A vector of 47 elements representing the main features (moments) of the primary part of each character is a result of this step. The elements of this vector are to be normalized before their becoming inputs to neural network.

4. Classification Using Neural Networks

The backpropagation feedforward neural network is used in this research to classify Arabic characters. The backpropagation network operates as a multi-layer, using supervised learning. These layers are input, output, and one or more hidden layer(s). Each layer consists of a number of neurons. Each neuron is connected to all the neurons in the successive layer through weights. The two important phases in using Neural Networks as a classifier are training and testing. During the training, the errors at the output layer are backward propagated through the hidden layers. The weight of each connection is adjusted for all the connections [7]. The process is repeated for either a predefined number of iterations or until the learning starts saturate. Learning starts by using a predefined training set where the input, which represents the Zernike moments in this research, and the desired outputs, is represented. The learning process is summarized as follows.

Present input vector $X_p=x_1,x_2,\ldots..,x_n$ and target output $T_p=t_1,t_2,.,t_m$ where n is the number of input nodes, and m is the number of output nodes. T_p is set to 0 except for one element which is set to 1, that corresponds to the class X_p belongs to. Calculate the actual output y_{pj} for the hidden layer (e.g. Eq (1)):

$$y_{pj} = f[\sum_{i=1}^{n} w_i x_h] \tag{1}$$

Pass it as input to the output layer o_{pj}.

Adapt weights, starting from the output layer, and working backwards (e.g. Eq (2)):

$$w_{ij} = w_{ij}(t) + \eta \delta_{pj} o_{pj} \tag{2}$$

where $w_{ij}(t)$ represents the weights from node i to node j at time t, η is a learning rate, and δ_{pj} is an error term from pattern p on node j .

For output units, the error term is calculated as in Eq (3):

$$\delta_{pj} = o_{pj}(1 - o_{pj})(t_{pj} - o_{pj}) \tag{3}$$

For hidden layer, the error term is calculated as in Eq (4):

$$\delta_{pj} = o_{pj}(1 - o_{pj})\sum \delta_{pk} w_{jk} \tag{4}$$

5. Experiment Results and Discussion

The data set used in this research is a library of segmented Arabic characters. They are gathered from scanned documents with one multi-size font. This is done using a 600 DPI scanner. The scanned documents are segmented into lines, words, and characters. These characters are saved in different records in a database library; each record consists of the font name, segmented character ID, and its position in the word (i.e. Isolated, Beginning, Middle, and End). For each stored character, its Zernike moments were computed and stored together with it.

The algorithm used in this research consists of many steps. The algorithm starts with dealing with the character represented by a binary image. The character is then segmented into Primary and Secondary Parts according to character histogram. The Primary Part, which represents the body of the character, is taken and passed through a preprocessing phase to be ready for the feature extraction phase. The preprocessing phase consists of a number of steps. First, calculate the centroid for each character image, and translate the character to the centroid of the image. Second, choose the best scale factor for scaling the character with almost all of its information. Third, calculate the max radius (i.e. the maximum distance between the centre of the character and the edge of the character). Following these steps the character gets ready for entering the feature extractor phase.

The second step in the algorithm is fitting the character into the unit disk with radius equal to the maximum radius calculated from the preprocessing phase. Calculate the Zernike moments, and get the 47 features that represent the feature vector for each character. The feature vector for each character is entered as an input to one of the backpropagation feedforward neural networks used here as a classifier. The information about the character position, which was deduced from the segmentation phase, helps enter the feature vector for each Primary Part to its appropriate Neural Network. A Neural network is used as a classifier. This Neural Network was trained to classify the primary part of the characters. The training was done off-line using manually fed data collected from different font sizes.

5.1 Recognition of Primary Parts

The data set was partitioned into two parts. The first one is used for training the vision system, while the other one is used for testing. For each stored character, its Zernike moments were computed and stored together with it.

A bank of four backpropagation neural networks is used to recognize Primary Parts of Arabic characters that result from the segmentation, as shown in Figure 4. Each neural network consists of an input layer, one hidden layer, and an output layer. Each neural network is meant to recognize one set of the primary parts according to the position of those primary parts (isolated, start, middle, or end). Thus the classification error rate will diminish. Each neural network has one hidden layer with 25 hidden neurons for Isolated Primary Parts, and 20 for Beginning, Middle, and End Primary Parts. The input layer of each neural network has 47 nodes; each one receives an input that represents one element of the Zernike feature vector. The output layer consists of 18 neurons for both Isolated and End Primary Parts, and 11 neurons for Beginning and Middle Primary Parts, which represent the number of classes for isolated primary characters. Figure 4 shows types of primary parts as input to each bank of the neural network.

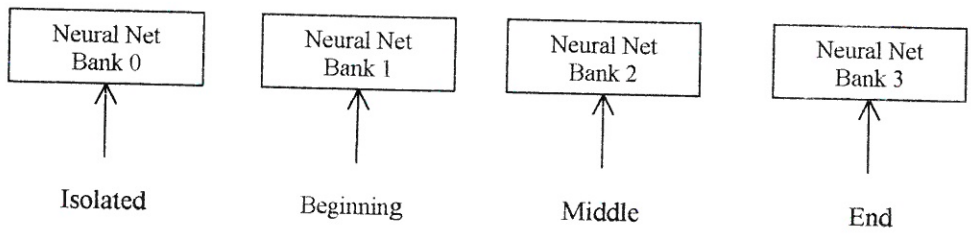


Figure 4. Types of Primary Parts As Input for Each Bank of the Neural Network

Two experiments are carried out. The first was training the network with one-size font, while in the second experiment the training was carried out using multi-size font. Testing was done using scanned characters from text documents with font sizes of 12, 14, and 16, and different versions for each size. Each isolated character is represented in a binary image of 65x 65 pixels. The unit disk is taken for each character and this is computed by finding the maximum radius of the character (i.e. the maximum distance between the centre of the character and the boundary of the character), so that the character could fit on the disk. The unit disk varies from 15 to 35, according to the character shape. The sum square error related to the number of epochs is shown in Figure 8. The description of the font sizes used in testing, the number of tested samples, and the recognition accuracy of the classifiers are also shown in Table 1. Figure 11 shows a character (Middle Ain) that the recognizer failed to recognize due to some dilation results from the segmentation process. Figure 5 shows the backpropagation neural network architecture for isolated primary characters.

Table 1. Description of the Experiment Done for Isolated, Beginning, Middle, and End Characters

Type of characters	Font Type	Font Training Size	Number of Training Samples	Tested sizes	Number of tested samples	Accuracy %
Isolated	Mudir	12	18	12,14,16	74	98.63
Isolated	Mudir	12,14,16	54	12,14,16	216	99.53
Beginning	Mudir	12	11	12,14,16	50	98.0
Beginning	Mudir	12,14,16	33	12,14,16	150	99.33
M	Mudir	12	11	12,14,16	56	94.5
Middle	Mudir	12,14,16	54	12,14,16	200	99.0
End	Mudir	12	18	12,14,16	60	96.7
End	Mudir	12,14,16	54	12,14,16	150	99.33

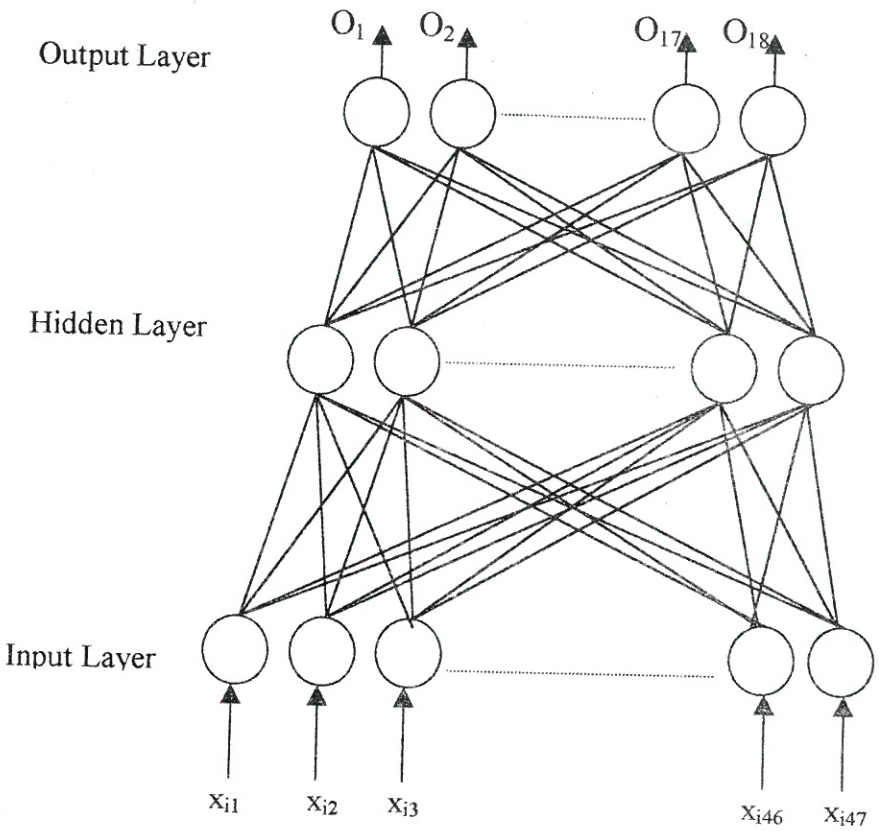


Figure 5. The Backpropagation Neural Network Architecture for Isolated Primary Characters

Figure 6 shows examples of the training set, isolated primary characters. Figure 7 shows 18 classes of Primary characters. Figure 8 shows the sum square error and learning rate for training with: a) one size font, and b) multi size font. Figure 9 shows examples of the training set of the middle primary characters. Figure 10 shows examples of the training set of the middle primary characters. Figure 11 shows that testing sample appears to be noisy.

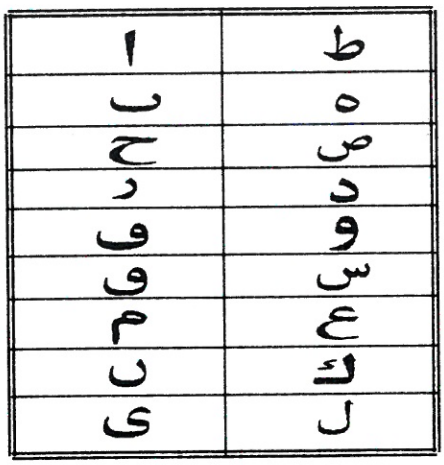


Figure 6. Examples of the Training Set of Isolated Primary Characters

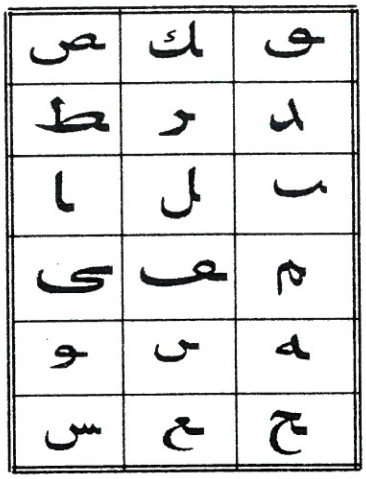
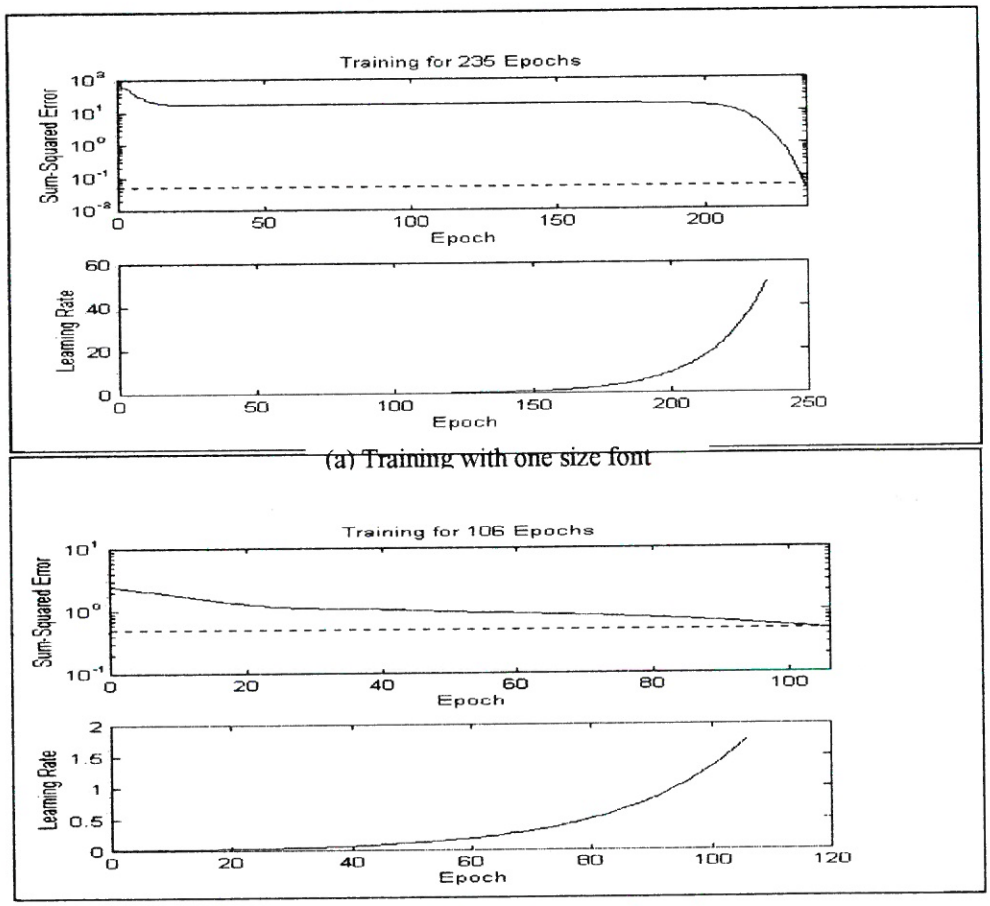


Figure 7. 18 Classes of Primary End Characters

Note: The disk could be represented as a circle that passes by the corners of the drawn rectangle around each character.



(a) Training with one size font

(b) Training with multi- size font

Figure 8. The Sum Square Error and Learning Rate for Training With: a) one size font, and b) multi size font

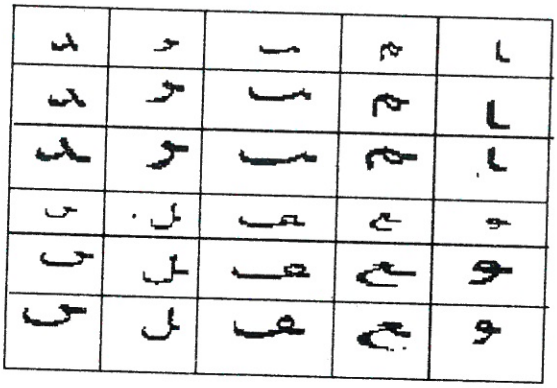


Figure 9. Examples of the Training Set of the Beginning Primary Characters

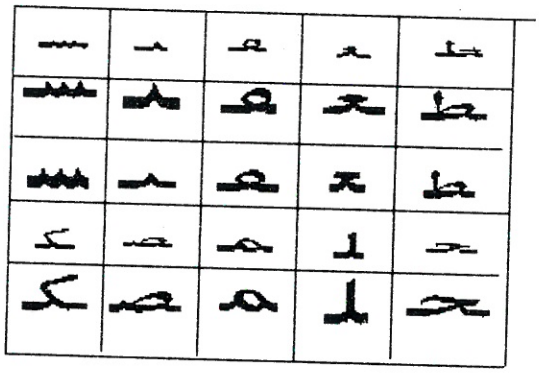


Figure 10. Examples of the Training Set of Middle Primary Characters

Figure 11. Testing Sample Which Appears To Be Noisy

5.2 Recognition of Secondaries

The Secondaries associated with the Arabic characters are only five. These are: one dot, two dots, three dots, Hamza, and Mada. The first four were dealt with in this research. The algorithm used for recognizing the Secondaries is summarized in the following.

If the number of segments from the segmentation and preprocessing stages is three, the secondary is evaluated as *three dots*.

If the number of segments is two, there are two possibilities: *three dots* if and only if any of the segments is wider than 1.5t, where t is the line thickness defined as the length of the column appearing most frequently but less than m, where m is the average column length. If both segments are less than 1.5t, the secondary is evaluated as *two dots*.

If there is only one secondary segment, there are three possibilities: a *hamza* if the segment length is greater than 1.5t. If the number of segments in the horizontal projection is two, it is a *two-dot* secondary. Otherwise, it is an *one-dot* secondary.

6. Conclusion

A new approach for typewritten Arabic character recognition has been presented. The introduced system is trainable. In order to improve its accuracy the user can add more samples to the system. Also, the system could be trained to recognize other font types. The AOCR problem was briefly introduced. This research treated the problem as a feature extraction and classification problem. A number of classifiers have been used to classify the Arabic characters according to their different positions and also because of the similarity of some characters with one another. The proposed recognition system can be applied to both Arabic and Latin characters, so it can be applied to multilingual documents. The idea of segmenting the character into two parts (Primary and Secondary) reduces the number of Arabic character classes, and hence increases the accuracy rate. The design and implementation of the proposed feature extractor took place, and the recognition approach for the Arabic OCR was done, and the performance was evaluated. The category used to measure the performance is the recognition rate, which is defined as the ratio between the correctly recognized characters and the total number of characters expressed as a percentage (ex: 97%, 98%, etc.). The overall recognition rate was 99.3 % for this system. Further work is to investigate the validity of the proposed recognition system for handwritten Arabic characters. Also, the system will be modified to support the recognition of Arabic vowels (TASHKEEL in Arabic text). A comparison study is carried on with a view at comparing the performance of this system with that of other systems.

REFERENCES

1. ADNAN, A., AL-SADOUN, H. and FISCHER, S., **Hand-Printed Arabic Character Recognition System Using An Artificial Network**, PATTERN RECOGNITION, Vol, 29, No. 4, 1996, pp. 663-675.
2. HEW, P. C. , **Fitting the Unit Disk**, Diary, Department of Mathematics, The University of Western Australia, *http:// maths.uwa.edu.au / phew / postgrad / diaries / unitdiskfns.ps.z*, October 1996.
3. SHOUKRY, A., **A Sequential Algorithm for the Segmentation of Typewritten Arabic Digitized Text**, ARABIAN JOURNAL OF .SCIENCE ENGINEERING 16(4), 1991, pp. 543-556.
4. AMIN, A. and MASINI, G., **Machine Recognition of Multi Font Printed Arabic Texts**, Proceedings of ICPR 1986, IEEE, 1986, pp. 392-395.
5. AL-YOUSEFI, H. and UDPA, S. S., **Recognition of Arabic Characters**, IEEE TRANSACTIONS ON PATTERN ANALYSIS, MACHINE INTELLIGENCE, PAMI-14, 1992, pp. 853-857.
6. FAHMY, M. and EL-MESSIRY, H., **Zernike Moments As Feature Extractor for Arabic Character**, the 1st-MINIA International Conference for Advanced Trends in Engineering, March 14-17, 1999.
7. SCHALKOFF, R. J., **Pattern Recognition: Statistical, Structural and Neural Approach**, JOHN WILEY & SONS, INC, 1992.