Military Technical College
Kobry El-Kobbah
Cairo, Egypt

10th International Conference
On Aerospace Sciences&
Aviation Technology

On Line Recognition System for Arabic Handwritten Text

M. Shaarawy*, Aly Fahmy **, and M. M .Fouad *

Abstract
This paper presents the design and the implementation of an On Line Arabic text Recognition system "OLAR" that is used for cursive handwritten recognition. In addition to Arabic characters, OLAR can recognize numerical characters, and special symbols. The direction and style of writing are used to compose the main components of the feature vector of the characters to be recognized. OLAR uses Euclidean distance approach and artificial neural networks for classification. The obtained results showed that OLAR can compete well with other handwriting recognition systems. The recognition rate ranges from 90% to 100%.

Keywords: Arabic Text Recognition, Artificial Neural Networks, Handwritten Text Recognition.

* Egyptian Armed Forces.
** Faculty of Information & Computers, Cairo University, Egypt.

1. Introduction

Pen based computing technology is one of the most rapidly expanding areas of current research, attracting people from a wide variety of disciplines. The reason behind rapid development of that technology is its ability to deal with free.hand written text instead of the usual keyboard and mouse, binary scanned images of handwriting text. However, the new technology suffers from the high cost and the tangible recognition delay felt by the users. This technology is usually used to deal with signature and handwriting text recognition systems to achieve speeding up processes, reliability, minimum effort, minimum waste of time, and low cost [1].

Although there are some systems developed for on.line English handwritten recognition, this kind of systems has not been developed in a reliable manner for on.line Arabic handwritten recognition. For instance, Calligrapher system uses a fuzzy logic matching algorithm to match written characters, both cursive and a mixture of cursive and printed, with "prototypes" of letters. Also, Apple Newton Print Recognizer (ANPR) uses segmentation, classification by neural networks, and context driven search. As well, IBM's ThinkWrite system uses algorithms like ANPR. Finally, Graffiti System uses a character system in which all letters of the alphabet and all symbols are represented by one stroke [2].

Moreover, due to the fact that Arabic texts are written cursively from right to left, an efficient algorithm is required for separating Arabic words into characters. This type of algorithms faces many problems, due to the features of Arabic handwritten texts, which can be summarized as follows [3]:

Arabic texts are written using a total of 28 characters.

Arabic words may constitute one or more connected portions (sub words), for example the word حيث constitutes one portion of three characters ث, ـيـ, and حـ, the word قال constitutes two portions, one characterل and قا of two characters ق and ا, and the word عـــاملون constitutes three portions; one character ن ، ملـو of three characters و , ـلـ and مـ, and عا of two characters ا and عـ.

Some characters are not connectable from the left side, for example the characters ا ، د and ر.

Some characters have more than one shape depending on its position within a word, for example ع ، ـعـ and عـ represent the same character. The first shape when the character is being at the end of a word, the second shape when the character is being connected from left and right sides and the third shape when the character is being at the beginning of the word.

Some characters have one, two or three dots up or down to a main stroke, for example ب ، ن ، ث ، ت and ـيـ..

The style of writing differs from person to another.

The character size may differ within the same text, even, within the same line or word.

Line spacing is not fixed or regular within the same text.

Word spacing is not fixed or regular within the same text.

Sub word spacing is not fixed or regular within the same text.

Some characters have loops, e.g., ط ، ق ، و ، ـهـ that are not fixed or regular in size or shape within the same text.

In this work the minimum distance approach is used with an artificial neural network to implement an online Arabic handwritten text recognition. Section 2, presents the hardware and the software configurations of the proposed system. Analysis of the obtained results is introduced in section 3. Finally, the conclusion and the future work are introduced in section 4.

2. Proposed System

A handwriting kit is used for entering Arabic texts. This kit consists of a pressure sensitive pen. The writing pen is cordless, with no batteries and achieves up to 1024 pressure levels. The writing tablet is transparent overlay technology and achieves resolution 1000 dpi.
The first step is to write with the pen to obtain the handwriting direction. The word boundaries have to be detected. The resulted spaces have to be classified either a separator between sub.words, or a segment between words. In turn, each character boundaries have to be detected to get its feature vector. Then, the character is classified to one of the basic characters. So, if there exists dot(s) or a segment around the character, it becomes one of the derived characters. Finally, the character is displayed, or saved into a documented file. These processes operate till the user closes the system or finishes writing.

2.1 Grouping of Arabic characters

To simplify the process of Arabic characters recognition, the Arabic character set is divided into two groups; basic character group, and derived character group Basic character group includes all characters that can not be derived from another character. They are 18 characters, taking into consideration that each character may have more than one shape giving 57 shapes, Table (1). Derived character group includes all characters that can be derived from one of the basic characters . by adding dot(s) or a segment. They are 22 characters. Also, each character has different shapes giving 68 shapes, Table (2).

2.2 Functional Block Diagram

The functional block diagram of the system is depicted in Figure(1). These functions include acquisition, detection of word boundaries, and detection of character boundaries, feature extraction, dots representation, and classification.

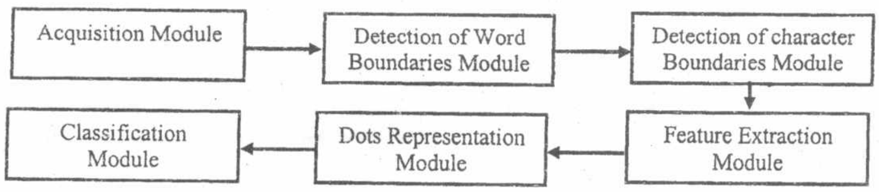
Acquisition Module
Detection of Word Boundaries Module
Detection of character Boundaries Module
Feature Extraction Module
Dots Representation Module
Classification Module

Figure (1) Functional block diagram of the software part

Table (1) Basic characters shapes

Character	Different Shapes				Total
	1	2	3	4	
ا	ا	ـا			2
ب	بـ	ـبـ	ـب	ب	4
ح	حـ	ـحـ	ـح	ح	4
د	د	ـد			2
ر	ر	ـر			2
س	سـ	ـسـ	ـس	س	4
ص	صـ	ـصـ	ـص	ص	4
ط	طـ	ـطـ	ـط	ط	4
ع	عـ	ـعـ	ـع	ع	4
ف	فـ	ـفـ	ـف	ف	4
ك	كـ	ـكـ	ـك	ك	4
ل	لـ	ـلـ	ـل	ل	4
م	مـ	ـمـ	ـم	م	4
لا	لا	ـلا			2
هـ	هـ	ـهـ	ـه	ه	4
و	و	ـو			2
ى	ى	ـى			2
ء	ء				1
Total					57

Table (2) Derived characters shapes

Character	Different Shapes				Total
	1	2	3	4	
أ	أ	ـأ			2
إ	إ	ـإ			2
آ	آ	ـآ			2
ت	تـ	ـتـ	ـت	ت	4
ث	ثـ	ـثـ	ـث	ث	4
ج	جـ	ـجـ	ـج	ج	4
خ	خـ	ـخـ	ـخ	خ	4
ذ	ذ	ـذ			2
ز	ز	ـز			2
ش	شـ	ـشـ	ـش	ش	4
ض	ضـ	ـضـ	ـض	ض	4
ظ	ظـ	ـظـ	ـظ	ظ	4
غ	غـ	ـغـ	ـغ	غ	4
ق	قـ	ـقـ	ـق	ق	4
ن	نـ	ـنـ	ـن	ن	4
ي	يـ	ـيـ	ـي	ي	4
لأ	لأ	ـلأ			2
لإ	لإ	ـلإ			2
لآ	لآ	ـلآ			2
ؤ	ؤ	ـؤ			2
ئ	ئـ	ـئـ	ـئ	ئ	4
ة	ة	ـة			2
Total					68

2.2.1 Acquisition Module

As soon as a person uses the pen for writing on the Tablet, the acquisition module gets the coordinates of the touched pixels. By these coordinates, the directions (TURNS) of each pixel are calculated, Table (3).

2.2.2 Detection of Word Boundaries Module

Arabic words are written connected along a virtual horizontal row. Some Arabic words may constitute one or more sub words. These sub words are separated by small spaces relative to those between independent words. In handwriting, these spaces, usually, are not regular. For example, in Figure (2), the horizontal spaces defined by wl, w2, and w3 are interword spaces, while those, defined by s1, s2 and s3 are sub word spaces. In order to differentiate interword spaces from that of sub-words, we measure all horizontal spaces (horizontal segments with no object pixels) along the mid row to determine a threshold value (d), such that, [4]:
All spaces $< d$ are considered as a sub-word space.
All spaces $\geq d$ are considered as a word space.

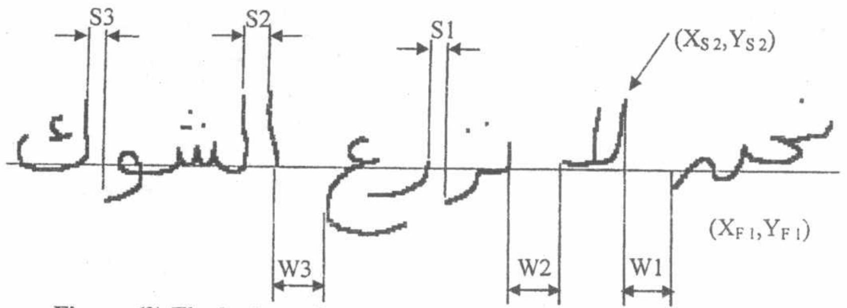

Figure (2) The horizontal spaces between words and sub.words

2.2.3 Detection of Character Boundaries Module

Finding the boundaries of characters is the most complex problem in this application. So, we should agree that sub-word means one stroke written by the pen. The following algorithm is considered to separate Arabic words into isolated characters. This algorithm is performed on basic characters; before applying dots module to the sub-word. The algorithm for separating Arabic sub-word into isolated characters is illustrated as follows:
Set X1, Y1 to the coordinates of the starting pixel as soon handwriting on the tablet and the pen is down. While writing, the pen touches the Table and the generated pixels are stored in {(X2, Y2), (X3, Y3)...(Xn, Yn), see Figure (3). Angle α is generated between the line segment – that connects each two successive pixels – and the horizontal direction. Each angle α corresponds to a TURN number, where :

$$\alpha n = \text{arc tan } [(Yn . Yn.1) / (Xn - Xn.1)]$$

As soon as the pen is moved up the tablet, the stroke (word, sub word, or letter) has been already written; the pen acts as a mouse up event.
Get the coordinate (Xf, Yf) of the pixel that is the most left in that stroke. When the pen continues writing (pen down) the starting pixel of the new stroke is then captured.
If the distance between the starting point of the new stroke and (Xf, Yf) of the previous stroke is greater than WORDSPACE then the new stroke is considered to be related to a new word, otherwise it is considered to be a sub word in the previous word.

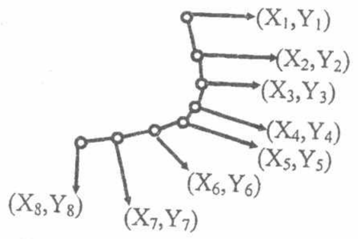

Figure (3) The stroke of character ' د ' consists of eight pixels

Table (3) Turn number corresponds to an angle range

Angel Range	Turn No.	Angel Range	Turn No.
.22.5o : +22.5o	4	+157.5o : .157.5o	8
+22.5o : +67.5o	5	.157.5o : .112.5o	1
+67.5o : +112.5o	6	.112.5o : .67.5o	2
+112.5o : +157.5o	7	.67.5o : .22.5o	3

2.2.4 Feature Extraction Module

As mentioned before, Arabic Basic characters are 28 characters, but due to the character Hamza (ء) is considered as a separate character in writing, we adapted our work to deal with 29 characters. Due to the various shapes that a character may have, we considered 120 character shapes for Naskh writing. The following algorithm is introduced to extract the feature vector of each character (without dots). The algorithm for extracting a feature vector of each segmented character is illustrated as follows, (see Figure (4)).

For each segmented character, a set of features is extracted. The feature vector has to start with letter "D" which means a mouse down action. Whereas, it ends with letter "U" this means a mouse up action.

The turns are put between the two letters "D" and "U".

In Arabic language writing, a stroke may constitute more than one character. For example, the stroke of حمد consists of three characters; حـ ، ـمـ ، ـد. The connection between two successive characters in a stroke is usually achieved along a horizontal connecting segment "ـ". So, separation of characters of the same stroke is achieved at this horizontal segment which corresponds to TURN "8".

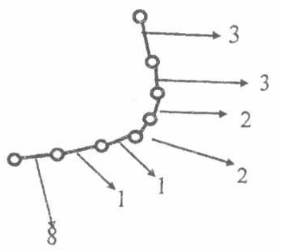

Figure (4) The feature vector of letter ' د ' in Turns will be (D 3 3 2 2 1 1 8 U)

4. If the feature vector of the stroke constitutes two or more "8" turns, these turns are considered to be a connection segment between two successive characters. Then separation between these two characters is done at these turns by inserting the character '@' between these "8" turns to isolate the feature vectors of the two characters. Each stroke of Figure (5) has its feature vector. In the feature vector of the first stoke, the separator "@" is inserted if two or more successive "8" turns appear. Thus, the stroke is divided into two characters "عـ", and "ـا". The feature vector of the first stroke is (D 8 1 1 2 2 3 4 4 1 8 @ 8 7 6 6 U). Thus, the letter 'عـ' has the feature vector (8 1 1 2 2 3 4 4 1 8), and the letter 'ـا' has the feature vectlor (8 7 6 6), which will be used to learn the ANN classifier.

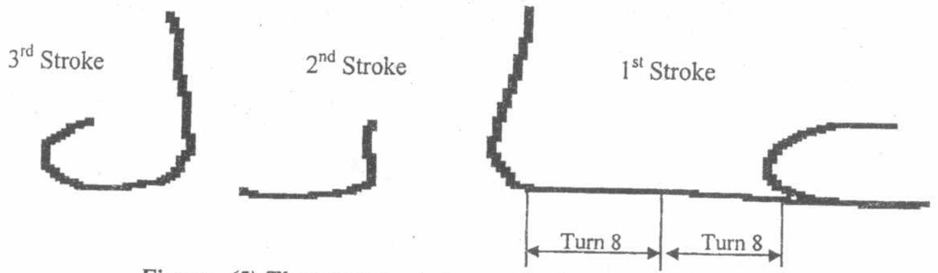

Figure (5) The separator between two characters'عـ' and 'ـا' is at two successive turns of '8'.

2.2.5 Dots Representation Module

In order to enhance the recognition performance, the characters are classified into three groups according to certain measures:

1- Characters without dots (e.g. ك . ط – ص – ع – ح – د).
2- Characters with dots (e.g. ج.خ.غ.ف.ق.ث.ض.ش.ي.ب.ت.ن.ز.ظ.ذ.ة).
3- Characters with down dots (e.g., ب . ي . ج)
4- Characters with up dots (e.g. خ – ف.غ.ض.ظ.ز.ت.ة . ث.ش.ذ).
5- Characters with one dot (e.g., ن.ظ . غ . ف.ض . ج . خ . ب.ز.ذ).
6- Characters with two dots (e.g., ت . ي . ق.ة).
7- Characters with three dots (e.g. ث . ش).

For each segmented character, a vector of three digits is generated to describe the existence, location and number of dots. For example, a character with no dots, the vector will be (0, 0, 0.), while the character ب (a character with one dot down), the vector will be (1, 0, 1), the character ق (a character with two dots up), the vector will be (1, 1, 2), the character ث (a character with three dots up), the vector will be (1, 1, 3). Then, each feature vector has 17 elements for Turns, and 3 elements for Dots. When writing a word or sub word having dots, it's a must to write the dots of the stroke that has been just written (e.g. if we write 'قانون ' , we have to write the two dots of the letter 'ق ' before starting writing the new stroke 'نو ') [2]. The stroke may consist of one letter. It's mentioned earlier, there is a separator ' @ ' between any two successive cursive letters in one stroke. The pixel that causes the turn 8 in the feature vector of the letter is considered as (Xf, Yf) [where (Xf, Yf) is the most left pixel in that character], also the most right pixel is calculated while writing, so if a dot or more up to three, has been written that range, and up or down the mid row, then we get the dot vector for that character. Let us see the next example, illustrated in Figure (6). The first stroke 'يذ ' will be classified to 'بد ' as default because tow successive turns of 8 are generated. So, we have got the boundaries of the first character (XS1, YS1) as a starting point and (Xf1, Yf1). When writing the first dot under the mid row within the range [XS1, Xf1], the feature vector is generated to be (1, 0,1). Hence, the letter 'ب' is produced. If the second dot is written within the same range [XS1, Xf1] under the mid row, then the letter 'ي' is produced instead of 'ب'. Also, the point (Xf1, Yf1) can be considered as the starting point of the second character 'ذ' in the same stroke 'يذ '. As soon as the pen is up, the point of (Xf2, Yf2) is generated. So, when we write a dot within [XS2, Xf2] above the mid row, then the letter 'ذ ' is generated instead of 'د '. Thus, the strategy is to recognize the written strokes as basic characters. Then, each basic character is changed or not due to putting dots or not.

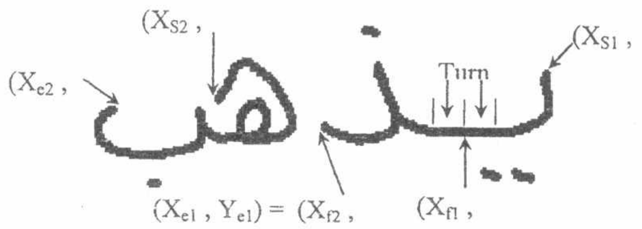

Figure (6) The word يذهب is represented in two strokes

But there are 9 letters that have not been explored yet like: ا , إ , آ , لآ , لإ , لأ , ك , ـذ and ؤ. The letter Hamza 'ء' is a basic character. It can be easily classified in the word 'مـاء '. The first stroke is ended with point (Xe1, Ye1) as shown in Figure (7). The second stroke starts with the point (XS2, YS2) and ends with point (Xe2, Ye2). So, if the (Xe1 < Xe2) then the second stroke is far horizontally from the first one (i.e. in this case the letter 'ء' is to be a basic character).

2.2.6 Classification Module

Till now the classification stage has not yet been explained. Each word is separated into characters or even some sub-words that also, can be segmented to characters. Hence, each character has to be well classified to constitute a readable sentence. Firstly, ANN has been used as classifier.

2.2.6.1 Artificial Neural Network as a Classifier

Hence, we have to classify the 57 shapes that correspond to 14 basic characters, which called ones. Once the written character is recognized, it's easily to make sure of it uses dots representation module. The feature vector of a character constitutes 20 elements; 17 elements for the generated TURNs, and 3 for dots representation. The number of targets is 18 classes [for basic characters]. We have used a package called 'NeroSoft' [7]. Using MLP module to represent the Backprobagation algorithm. The system has the ability to add any new shapes of characters, symbols and digits. So, we have added 14 symbols and 10 digits, see Table (4).

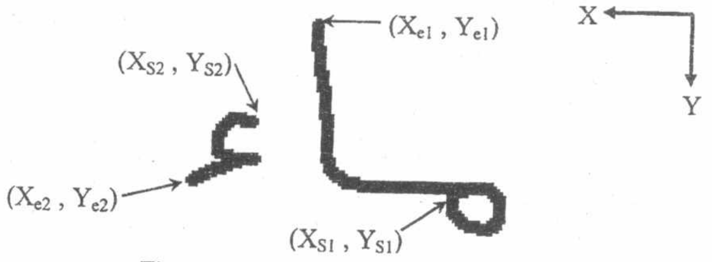

Figure (7) The letter ' م ' is a basic character

Table (4) Symbols and digits of Arabic Language

No.	Symbol	No.	Symbol	No.	Digit
1	.	11	(	1	٠
2	/	12	)	2	١
3	\	13	~	3	٢
4	_	14	؟	4	٣
5	+			5	٤
6	X			6	٥
7	[			7	٦
8	]			8	٧
9	{			9	٨
10	}			10	٩

Classification Algorithm

According to the geometric rule [5] of choosing the number of a hidden layer, it has been computed to be 8 nodes [6].

We should divide the examples set into two parts. 4300 examples are to be used in learning stage. Also, 2150 examples are to be used in testing stage.

Each person has written ten samples of each 42 characters.

Finally, a table consisting of two columns has been formed; the first column includes the character and the second column includes the corresponding feature vector elements, see Table (5).

After learning and testing, the ANN is now ready as a classification module. Some problems have appeared which will be explained next.

Table (5) The character and its corresponding feature vector

Character	Feature Vector
-	8
/	8 1
٦	5 4 3 2 3 2
م	1 2 1 2 7 6 7 8 2
ك	6 8 2 1 2 1 7

2.2.6.2 Minimum Distance Approach (MDA) as First Stage

If a new user uses the system and obtains incorrect characters, then he should add many samples of each unrecognized character. But, if we still use the ANN in classification, the later has to be learned and tested again. This will have taken 6 hours. Surely, it is not reasonable to force the user to do so.

We had to search for another way that supports adding new samples for any new user during the program execution without re-training the ANN. So, a classical approach (MDA) has been chosen as a first stage of recognition. If the length of the feature vector is less than 20, then it will be padded with zeros till its length equals 20.

The distance d is computed according to the following rule [3], then the least one is chosen,

$$d_i = \sum_{n}^{j=1} [(x_i - x_j)^2 + (y_i - y_j)^2]$$

where n is the number of character feature vectors in database.

A trained and tested ANN1 is used to classify between (ف, و , and ٩). Another trained and tested ANN2 is used to classify between (س and ٣). Also, a trained and tested ANN3 is used to classify between (ص , هـ , and ط). These 8 characters are the most misclassified ones after using ANN only.

The result of using MDA is introduced. The average recognition rate of the previous eight characters is clearly increased, Table (6). As a result, the recognition rates of the six characters {ش , ض , ظ , ؤ , ة , ق } , which are dependent on the previous eight characters, are obviously improved in turn.

3. Results of Recognition Experiment

Different persons have performed the experiment of recognition for all characters as follows:

The number of shapes of the whole Arabic characters is 42.

The number of persons is 15.

Each person has tested the system with 10 patterns for each shape.

The number of tested patterns per person is 420.

The number of tested patterns for all persons is 6300.

3.1 Using ANN Only

There are 28 classes with a recognition rate of range 89.100%. This range of is highly acceptable result.

But there are 14 classes (س , ٣ , ص , ط , هـ, ف , و , ٩ ش , ض , ظ , ة , ق ,and ؤ) with a recognition rate of range 80.89%. This result can be highly improved.

3.2 Using a hybrid of MDA and ANN

The recognition rate of the 14 most confused classes has been improved up to 90.95.6%, see Table (6). The first 28 classes still have its recognition rate range 89.100%.

As a result, the recognition rate of the 6 dependent characters {ش , ض , ظ , ة , ق , ؤ} has been increased.

Table (6) Analysis of recognizing the most miss classified 8 characters

No	Class	Recognition Rate	
		Using ANN Only	Using MDA and ANN
1	ف	83.3%	92.6%
2	و	83.3%	91.3%
3	٩	83.3%	94.6%
4	س	85.3%	91.3%
5	٣	83.3%	95.6%
6	ص	84.6%	93.3%
7	ط	82.6%	90.0%
8	هـ	83.3%	91.3%

Table (7) Classification Rate of a subset of handwritten sentences

No.	Handwritten Sentence	Recognized Sentence	Miss classified
1	بسم الله الرحمن لرحيم	بسم الله الرحمن الرحيم	0
2	الله اكبر ولله الحمد	الله اكبر و لله الحمد	0
3	الظريف والمؤدب والآخرين	الضريف والموءدب والأخرين	2
4	القوى والضعيف والظهور	القوي والضعيف والصهور	1
5	جمهورية مصر العربية	جمهورية مصر العربية	0
6	الكلية الفنية العسكرية	الكلية الفنية العسكرية	0
7	الصلاة على وقتها	الصلاة علي وقتها	0
8	افضل الأعمال ذكر الله	افضل الأعمال ذكر الله	0
9	السلم و الثعبان	السلم و الثعبان	0
10	شئ من الخوف	شئ من الخوف	0
11	بنك الائتمان الزراعى	بنك الائتمان الزراعي	0
12	الإسلام هو الحل	الإسلام هو الحل	0
13	الإسفنج يمتص الماء	الإسفنج يمتص الماء	0
14	التنظيم والإدارة	التنظيم والإدارة	0
15	(٤)٣-٦/٧\٨+٩[ ]؟	٣ع(٤)-؟ [ ] ٩ + ٨\٧/٦	1

3.3 Recognition rate of some ARABIC sentences

To test the proposed system for recognition of Arabic sentences, some Arabic sentences have been written and recognized. Samples of tested sentences are introduced along with the achieved results are shown in Table (7).

4. Conclusion

This paper presents the design and implementation of a new online handwriting Arabic recognition "OLAR" system using neural network techniques and classical approaches. The tablet feeds the recognition engine by a sequence of touched pixels. A set of features derived from the direction of pen movement called Turns. The recognition engine operates in real time where the recognition algorithm is based on the use of Minimum Distance and Neural Network classifiers. The recognition rate of "OLAR" is acceptable in such type of applications. "OLAR" provides multiple recognition types and allows users to write naturally in cursive Arabic language. In addition, "OLAR" exports the resulted sentences into a Word Document.

REFERENCES

[1] Aly Fahmy and M.Shaarawy, "A Real Time System For Handwriting Recognition", Artificial Intelligence Applications Conference, Cairo, January 1994.

[2] Lossev and Ilia., "Characters from Russia: ParaGraph's calligrapher Handwriting Recognition". Pen Computing Magazine pp 34-35, June 1999.

[3] M. Shaarawy, "Optical Character Recognition of Arabic Text", 7th International Con. For Mechanical Applications, Cairo, Mai 1996.

[4] M. Fahmy Tolbah, "Handwriting Alphanumeric Character Recognition", 15th National Radio Science Conference, Cairo, Egypt, 1997.

[5] J. Yakoto and K. Shang Li. " On.Line Handwriting Recognition System of Chinese Characters ", Chapman & Hall, London, 1998.

[6] M. Zaki, M. Shaarawy and H. El.Deep, "A Neural Network Approach for the Recognition of Handwriting Arabic Characters", 13th National Radio Science Conference, Cairo, Egypt, 1997.

[7] Khaled M. Badran, "Artificial Neural Networks Techniques And Their Implications", M. Sc. Thesis, Military Technical College, Cairo, 2000.

[8] M. M. Fouad, "Pen Based Computing for The Recognition of Arabic Text", M. Sc. Thesis, Military Technical College, Cairo, 2001.