Printed and Handwritten Arabic Characters Recognition and Convert It to Editable Text Using K-NN and Fuzzy Logic Classifiers

Zamen F. Jaber
Computer Department, College of Computers and Mathematics Sciences, University of Thi-Qar
zamenfadhel@yahoo.com

Abstract

In this paper we suggest an off-line isolated Arabic characters recognition system and this system has an ability to recognize printed and handwritten Arabic character and then convert these characters to printed text by mixed image processing techniques and artificial intelligent system .These techniques used to find rigid features for each Arabic character to distinguish it from another characters in Arabic language. K-Nearest Neighbors (K-NN) classifier was used to classify the printed character and fuzzy logic to classify handwritten Arabic character. Different font of printed character have font type (Arabic transparent, Times New Roman, Arial, simplified Arabic fixed) and font size 14 in order to test the quality of our system, each printed character in Arabic alphabet entered nine times, (6) of them in Arabic transparent font while other (3) is in Times New Roman, Arial, simplified Arabic fixed respectively while each handwritten character enter six times three of them used to training and the remaining (3) are used to testing. So 324 printed character are entered to our system ,the system successes in recognize 301 character form printed characters with recognition ratio is 92.9%.while 216 handwritten Arabic character entered to our system which successes in recognize 208 character of them with recognition ratio is 96.6%. Elapsed time in execute our system to perform recognition process for one character is 0.04 seconds.

Keywords: image process, optical character recognition, K-Nearest Neighbors, Fuzzy logic**.**

1. Introduction

Pattern recognition is concerned with the automatic detection and classification of objects or events. Character recognition (CR) automation occupies a big and intensive research of the pattern recognition research area. It has attracted the attention of researchers during the last five decades. CR means translating images of characters into an editable text; in other words, it represents an attempt to simulate the human reading process.

The objective of optical character recognition (OCR) is automatic reading of optically sensed document text materials to translate human readable characters to machine-readable codes. Research in OCR is popular for its various application potentials in banks, post-offices and defense organizations. Other applications involve reading aid for the blind, library automation, language processing and multi-media design [1].

The CR can be an on-line or off-line type. If the CR system has the ability to trace the points generated by moving a special pen on a special screen, the system

belongs to the on-line type, while the system belongs to the off-line type when it accepts only the pre-scanned text images to perform the recognition function. The off-line type deals with printed and handwritten texts, while the on-line type deals with handwritten texts only [2].

However, developing OCR systems for other languages such as Arabic didn't receive the same amount of attention. Arabic is the official language of all countries in North Africa and most of the countries in the Middle East and is spoken by 234 million people [3,4]. It is the sixth most commonly used language in the world. When spoken Arabic varies across regions, but written Arabic, sometimes called "Modern Standard Arabic" (MSA), is a standardized version used for official communication across the Arab world [5]. The characters of Arabic script and similar characters are used by a greater percentage of the world's population to write languages such as Arabic, Farsi (Persian), and Urdu [6].

This paper is organized as follows. In section 2, we briefly introduce preliminaries for some image processing techniques, in Section 3 presents our proposed method which consists of read character image, preprocessing, feature extraction and classification stages. The experimental results are given in section 4 and finally conclusions are gathered in section 5.

2. Preliminaries

2.1 Morphological operation closing

There are collections of non-linear operations related to the shape or morphology of features in image, which are affecting the structure or shape of an object. There are two basic morphological operators: erosion and dilation. Opening and closing are two derived operation in terms of erosion and dilation. Dilation allows object to expand, thus potentially filling in small holes and connection disjoint objects. Erosion shrinks objects by etching away (eroding) their boundaries. These operations can be customized for an application by proper selection of the structuring element, which determines exactly how the objects will be dilated or eroded [7].

Often when noisy image are segmented by thresholding, the resulting boundaries are quite ragged, the objects have false holes, and the background is peppered with small noise objects. Successive openings or Closing can improve the situation markedly. Sometimes several iterations of erosion, followed by the same number of dilations, produce the desired effect [8]. The following algorithm displays the process of image closing for gray scale image [9].

1- Set flat structural element (z) with proper size.
2- Translate the structural element to all location in the image f(x,y)
3- At each translated location, the rotated structural element values are added to the image (f) pixel value and the maximum is computed result in R1 image:

$$R1(x,y) = \max\{f(x-x',y-y') \mid (x',y') \in Dz\}$$

4- Translate the structural element to all location in the image R1(x,y).
5- At each translated location, the rotated structural element values are subtracted from the image (f) pixel value and the minimum is computed result in R2 image:

$$R2(x,y) = \min\{R1(x+x',y+y') \mid (x',y') \in Dz\}$$

6- Subtract the last result image from the original:

$$R=(f-R2)$$

Algorithm1of image smoothing by apply closing [9]

2.2 Connected Components Analysis

A common approach to identifying isolated symbol images begins by grouping together touching (connected) pixels to create a set of connected components. Given a binary input page p, individual pixels can be represented by $p(x, y) = v$ where x denotes the horizontal and y the vertical distance (measured as the number of pixels) away from the origin of the page. Often, this origin is set to the top-left corner of the page. The pixel value v for binary intensity images implies that v must be one of 0, or 1 with the convention that foreground pixels will have v = 1, and background pixels v = 0. Using this description then, a foreground pixel$p(x, y)$ is said to be connected to a second foreground pixel $p(x', y')$ if and only if there is a sequence of neighboring foreground pixels $p(x_1, y_1), \ldots p(x_n, y_n)$ where $x = x_1, y = y_1$and $x' = x_n, y' = y_n$. Two pixels $p(x_i, y_i)$ and $p(x_j, y_j)$are said to be neighboring under what is called an 8-connected scheme exactly when $|x_j - x_i| \leq 1 \; and \; |y_j - y_i| \leq 1$. The same two pixels are considered neighboring under a 4-connected scheme when one of the following two cases holds: either $|x_j - x_i| \leq 1 \; and \; |y_j - y_i| = 0$, or $|x_j - x_i| = 0$ and $|y_j - y_i| \leq 1$. For 4-connected schemes, neighboring pixels must be vertically or horizontally adjacent, whereas in 8-connected schemes, neighboring pixels can be vertically, horizontally, or diagonally adjacent. Both of these schemes are illustrated in Figure 1. A single connected component c_i is then defined as a set of foreground pixels such that each is pair wise connected with each other pixel in that set. The pixels of each connected component are assigned the same unique label (which we assume is a positive integer), and so the task of connected components analysis then is to turn some input image p of pixel intensity values, into a second labeled image p' where each foreground pixel is assigned the value of the connected component to which it belongs (background pixels are typically all given the same label like 0, which is distinct from each foreground label) [10]. Each connected component can be thought of as a member of the same equivalence class, where the equivalence relation between pixels is based on the concept of them being 4 or 8-connected. To illustrate this, a small sample binary image is presented, along with its resultant 8-connected components labeling in Figure2 [10].

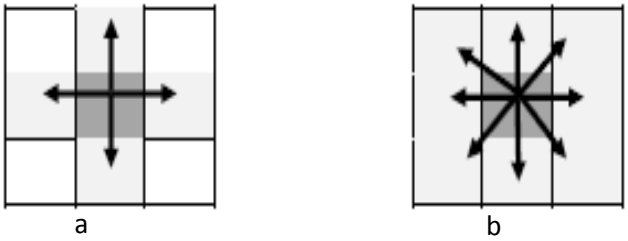


Figure 1: A single pixel and its 4-connected (a), and 8-connectedneighborhood (b)

1	0	0	1	1	0	0	1
1	0	1	1	0	0	0	1
1	1	1	1	0	0	1	1
0	0	1	1	0	1	1	1
0	0	0	0	0	0	0	1
1	1	0	0	0	0	1	0
0	1	1	0	0	0	0	0
0	1	0	1	0	0	1	0
1	1	1	0	0	0	0	0
1	1	0	0	1	1	1	0

a

1	0	0	1	1	0	0	2
1	0	1	1	0	0	0	2
1	1	1	1	0	0	2	2
0	0	1	1	0	2	2	2
0	0	0	0	0	0	0	2
3	3	0	0	0	0	2	0
0	3	3	0	0	0	0	0
0	3	0	3	0	0	4	0
3	3	3	0	0	0	0	0
3	3	0	0	5	5	5	0

b

Figure 2: A Small binary input image (a), and its resultant 8-connected components labeling (b)

2.3 Euler number

The *Euler number* (or genus), of a binary image is the difference between the number of connected components (objects), and the number of holes. Let the binary image be represented by 0-1 pixel matrix of size (N ×M), in which an object (background) pixel is denoted as 1. In a binary image, a *connected component* is a set of object pixels such that any object pixel in the set is in the 8 (or 4) neighborhood of at least one object pixel of the same set. A *hole* is a set of background pixels such that any background pixel in the set is in the 4- (or 8-) neighborhood of at least one background pixel of the same set and this entire set of background pixels is enclosed by a connected component. The sets referred to in the definition are sets contained in the image. As an example, the figure of '9' shown in Figure 3 has a single connected component and only a single hole. So, its Euler number is 0 [11].

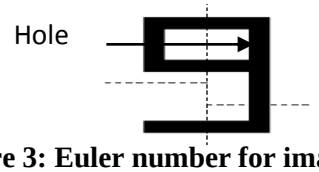


Figure 3: Euler number for image of digit 9

The most efficient and simplest algorithm, reported so far, for computing Euler number of an image works by looking into local patterns. Consider the following set of $2x2$ pixel patterns called bit quad:

$$Q_1 = \left\{ \begin{matrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0' & 0 & 1' & 0 & 0' & 0 & 0 \end{matrix} \right\}$$

$$Q_2 = \left\{ \begin{matrix} 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1' & 1 & 1' & 1 & 0' & 0 & 1 \end{matrix} \right\}$$

$$Q_3 = \left\{ \begin{matrix} 0 & 1 & 1 & 0 \\ 1 & 0' & 0 & 1 \end{matrix} \right\}$$

Let C_1, C_2, C_3 be the number of patterns Q_1, Q_2, Q_3, respectively in the image S. It has been that under the definition of 4 – connectivity the Euler number can be computed as [11]:

$$E(s) = \frac{1}{4}(C_1 - C_2 + 2C_3) \qquad (1)$$

And for 8- connectivity

$$E(s) = \frac{1}{4}(C_1 - C_2 - 2C_3) \qquad (2)$$

For example in figure 4 if X= the number of occurrences of the $2x2$ pattern: [0 0; 0 1] and V=the number of occurrences of the $2x2$ pattern:[0 1;1 1] then $Euler = X - V = 1 - 1 = 0$.

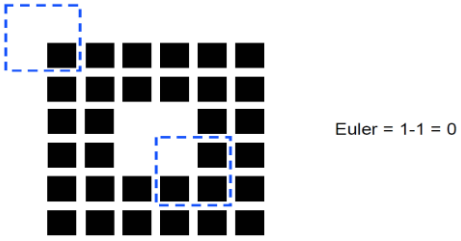


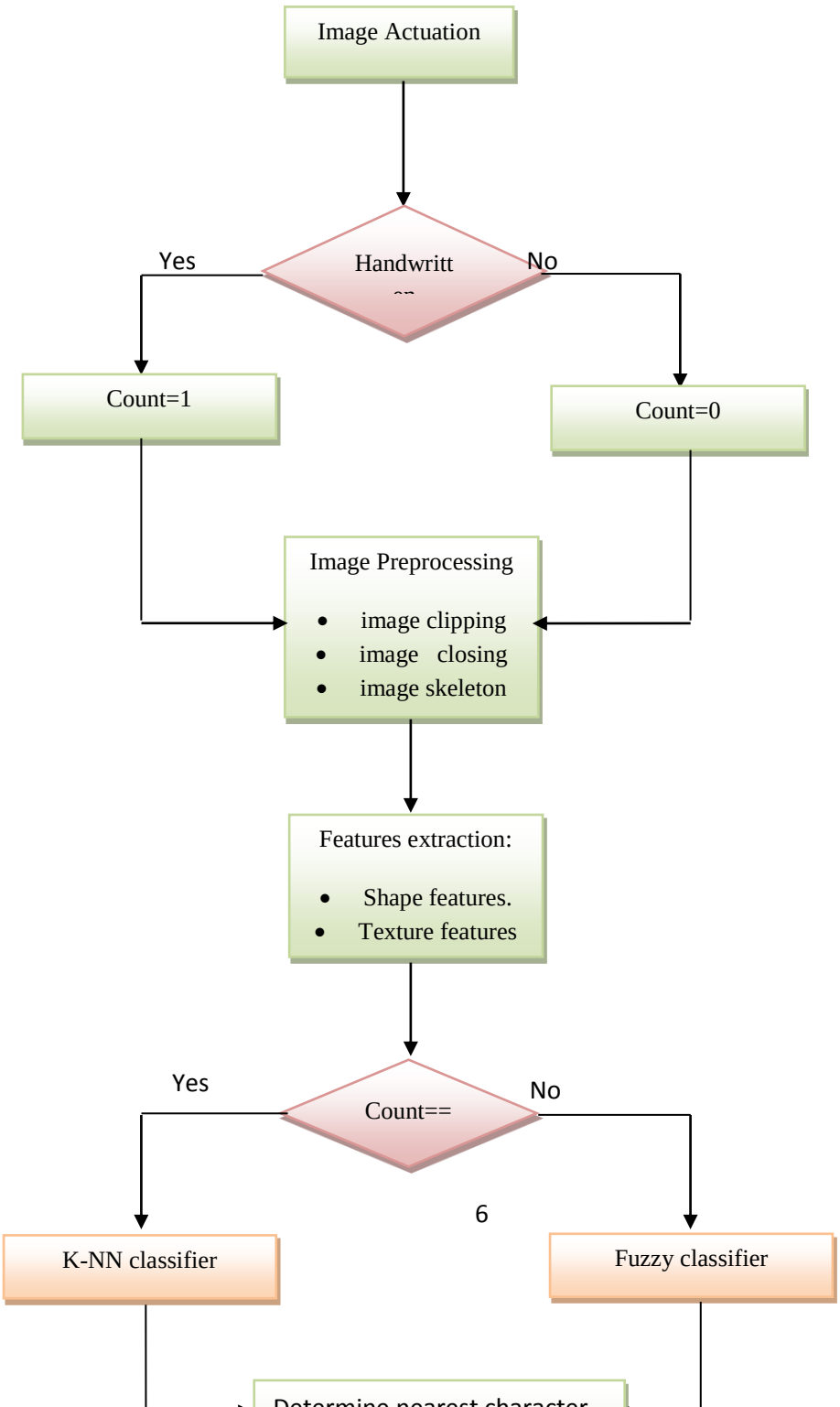
Figure 4: Object one connected component and one hole

3. Proposed Recognition system stages

There are four major stages for our proposed character recognition system, each stage perform particular role to repair the character image to next stage in proposed recognition system, The list below show this stages.

1. Characters Image Actuation stage.
2. Image preprocessing stage.
3. Feature extraction stage.
4. Characters classification &Text saving stage.

Figure5 shows the block diagram of this system in details, where each major stage include number of internal steps where the image Actuation stage include scanning and testing the image if it belong to printed character or to handwritten character, preprocessing stage include image clipping, image closing and image thinning, Feature extraction stage include extract shape and texture features for character, classification stage include k-NN classifier for printed character and fuzzy logic classifier for handwritten characters.



3.1 Character image Acquisition

The test images have been scanned by using HP Scan Deskjet F2400 scanner with horizontal and vertical resolution 300 dpi, bit depth 24, and the image store in JPEG format, we prepare counter set it is value to 1 if image belong to handwritten character and this counter set to 0 if the image belong to printed character.

3.2Image Preprocessing

The images which have been acquisition is 3-D we must convert these images into 2-D because 2-D images is easier to dealing with it and speed up in computation is greater than we dealing with 3-D images.

The image in gray level has noise resulted from scanner so we must remove it to increase the rate of recognition and minimize the error as possible. Gaussian filter is used to remove the noise which appear as random inflection in gray level inference in original image, filter window was 3×3 and standard deviations is 0.5.

Now we convert the gray filtered image to binary image using global threshold .The resulted image has 0 value to foreground (character lines) and 1-value to background. The image is negative by convert each 0 to 1 and each 1 to 0 to give the interest value to character and uninterested value to background.

3.2.1 Image clipping

Horizontal and vertical projections are used to calibrating the beginning and ending of character to riddance from undesired areas of image. We suggest algorithm2 to perform image clipping, which it depend in working on horizontal and vertical projection as it is defined in equation (3), (4) respectively [12]:

$$H(i) = \sum_{j} I(i, j) \ldots \quad (3)$$

$$V(j) = \sum_{i} I(i, j) \ldots \quad (4)$$

step1: $count \leftarrow 0, count2 \leftarrow 0;\ emp \leftarrow [];\ emp2 \leftarrow [];\ img\_out \leftarrow [];$
step2: scan the image I row by row.
step3: find horizontal projection H(i) to each row as in equation (1).
step4: if $H(i) > 0$

$$count \leftarrow count + 1;$$
$$emp(count) \leftarrow index\ of\ this\ row;$$

end;

In figure 6 we display some of the characters images before and after apply algorithm2.

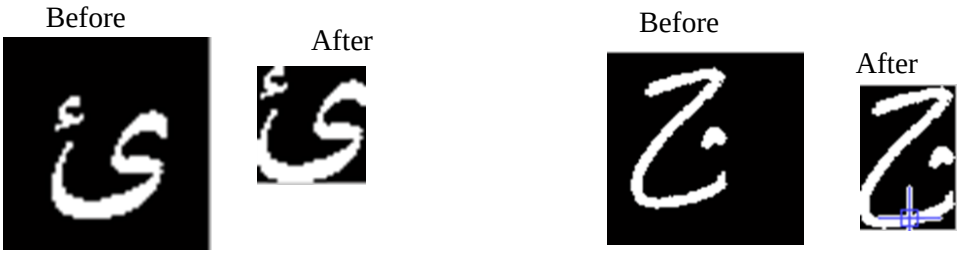


Figure 6: Image before and after remove undesired region

3.2.2 Image closing

In our proposed system we use morphological closing operation with square structural element with size 2×2 in order to smooth the counter of character image and filling the holes in it with same concept of the algorithm1, figure7display characters images before and after apply morphological operation closing.

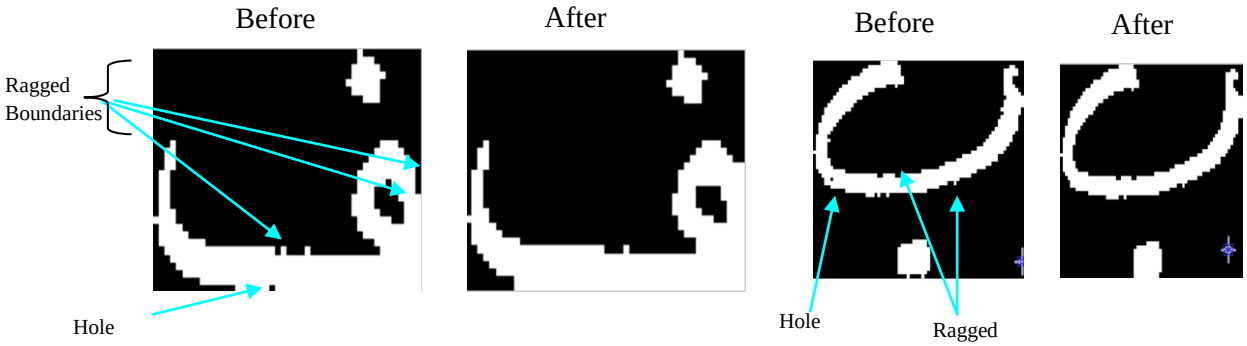


Figure 7: Images before and after morphological closing

3.2.3 Charactersskeleton

The goal of skeleton is to eliminate the thickness of character in order to extract the character backbone which is consider common features in all entering character (in training character image and all eight testing image for the same printed character and same thing done in 6 handwritten character), so that the object in resulted image has one pixel thick, as shown in figure8.

Figure 8: Images before and after thinning process

3.3 Features extraction

In this stage we extract two group of features which is *shape* features vector include 14 features , *texture* feature with 2 features vector include connected component numbers and Euler number and as result we group these two vectors in one that include all 16 features.

3.3.1 Shape features

F1=the rate of high to width.
F2= the rate of black pixels to white pixels.
F3= number of horizontal transitions
F4= number of vertical transitions

The ***horizontal and vertical transition*** is a technique used to detect the curvature of each character and found to be effective for this purpose. The procedure runs a horizontal scanning through the character box and finds the number of times that the pixel value changes state from 0 to 1 or from 1 to 0 The total number of times that the pixel status changes, is its horizontal transition value. Similar process is used to find the vertical transition value. Matrix of each character divided into four ***regions*** as shown in figure9 to extract the features from feature F5- to feature F14.

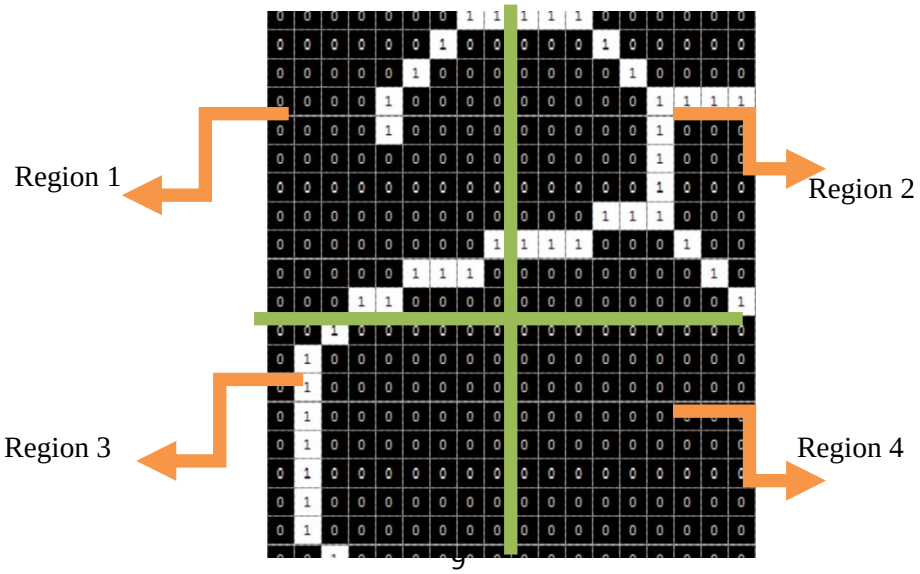


Figure9: character image regions

F5= Black Pixels in Region 1/White Pixels in Region 1
F6= Black Pixels in Region 2/White Pixels in Region 2
F7= Black Pixels in Region 3/White Pixels in Region 3
F8= Black Pixels in Region 4/White Pixels in Region 4
F9= Black Pixels in Region 1/Black Pixels in Region 2
F10= Black Pixels in Region 3/Black Pixels in Region 4
F11=Black Pixels in Region 1/Black Pixels in Region 3
F12= Black Pixels in Region 2/Black Pixels in Region 4
F13= Black Pixels in Region 1/Black Pixels in Region 4
F14= Black Pixels in Region 2/Black Pixels in Region 3

3.3.2 Texture features

There are 2 type of texture feature depend on the following:

a. ***Connected components number (F15):*** the explicit feature in Arabic characters is most characters include secondary part such as "points and Hamza" and the position of this secondary part differs according to letter they appear in it. For instance the letter "ب" appear the point below while character "ن" its point appear in above and so on as it shown in table1.

Table 1: Arabic characters and position of secondary

doesn't contain secondary part	one above secondary part	one below secondary part	more than one above secondary part	more than one below secondary part	middle secondary part
ا،ح،د،ر،س،ص،ط،ع،ل،م،ه،و،لا،ء	ن،ف،خ،ذ،آ،أ،ؤ، ز	ب،إ	ت،ث،ق،ة ،ش	ي	ك،ج

A Recursive connected component labeling algorithm with 8-neigbores search is used to label each object in image with unique name to differentiate it from other objects in the image as shown in algorithm3.

```
Function  RecConnComp (P)
        LP ←Negate(P) .                          %this flips foreground pixels in P to -1
        label ←0
        LP←FindComps (LP, label)
        return LP .                               % the final labeled component image
end function
Function  FindComps (LP, label)
NumRow ←NumRows(LP) .                    %determine the number of rows in LP
NumCol ← NumCols(LP) th. %determine number of columns in LP
for  r = 1  to NumRow  do
for c = 1  to NumCol  do
if LP(r, c) = = −1 then
                    label ←label + 1
                    LP←Search(LP, label, r, c)
              end if
end for
       end for
return LP
end function
```

b. Euler number (F16) is measure of the topology of an image. It is defined as the total number of objects in the image minus the number of holes in that objects. We can use either 4- or 8-connected neighborhoods. We compute Euler number for each character as concept explains in section 2.3.The Euler number of letter "ب"is 2 and Euler number for character "و"is 0 and so on based on number of object and holes in each character.

Table2 and Table3 show features vectors for some printed and handwritten character respectively.

Table 2: Features vector for some printed character

char	features vector															
	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16
ا	0.17	0.99	69	22	0.2	0.33	0.18	0.54	0.5	0.58	0.82	0.95	0.47	0.3	1	1
ب	1.32	0.99	90	104	0.06	0.05	0.07	0.11	0.94	0.7	0.7	0.52	0.48	0.05	2	2
ق	0.77	1	146	135	0.01	0.11	0.1	0.13	0.06	0.77	0.06	0.84	0.05	0.11	3	1

Table 3: Features vector for some handwritten character

char	features vector															
	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16
ا	0.27	1.001	106	37	0.18	0.025	0.08	0.15	5.4	0.7	1.5	0.2	1.125	0.02	1	1
ب	1.24	0.99	210	246	0.02	0.03	0.03	0.05	0.7	0.7	0.8	0.7	0.58	0.03	2	2
ث	1.04	0.99	276	382	0.007	0.016	0.033	0.03	0.46	1.1	0.2	0.5	0.25	0.01	4	4

3.4 Character classification

3.4.1 Printed character classification

As we explain in first stage from our proposed system if the character image is printed text the counter *count* set to binary value 0 else it assign to binary value 1. We now test if count =0 Then ***K Nearest-Neighbors*** method is used to classification stage which is consider easy and in the same time is very strong classification method. It

used in many applications especially in classification process. When using K Nearest-Neighbors in classification for each class V in training set the features vector which is labeled $fv$is computed then the features of unknown vector which is represent by symbol$U$also computed. To assigning the class of character$R$we will compute it's identicalness with each class by compute a distance between fv and$U$based on the following equation:

$$dv = \sqrt{\sum_{j=1}^{N}(U_j - fv_j)^2} \ldots\ldots(5)$$

Where $j = 1, 2, 3, \ldots N$ (N : number of computed features).

Then we must classify the character to nearest class using the following equation:

$$d_R = min\,(dv) \quad \ldots\ldots\ldots(6)$$

Where R=Number of classes.

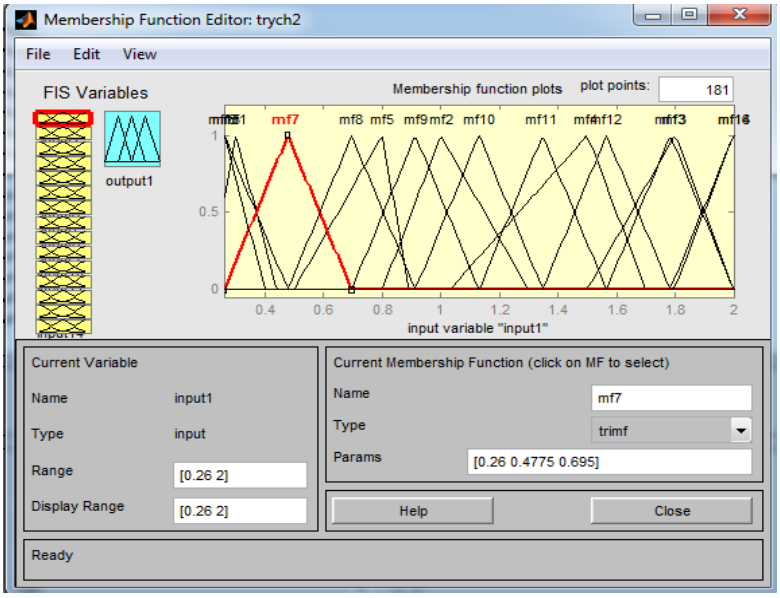
We use connected component number in other job which is divide the database of printed character to four group according to number of connected component to reduce the error in classification using K-NN classifier of to classify database according to number of connected component for each character. Group1 contain letter with one connected component, group2 contain letter with more one connected component above, group3contain letter with more one connected component bellow, group4 contain letter with more one connected component in middle.

Then we measure the difference distance between database vectors and inputting image vector we use Euclidian distance to compute this difference then we select the vector which gave us minimum cost this will produce ASCII code for character we print the character belong to this ASCII code in text file.

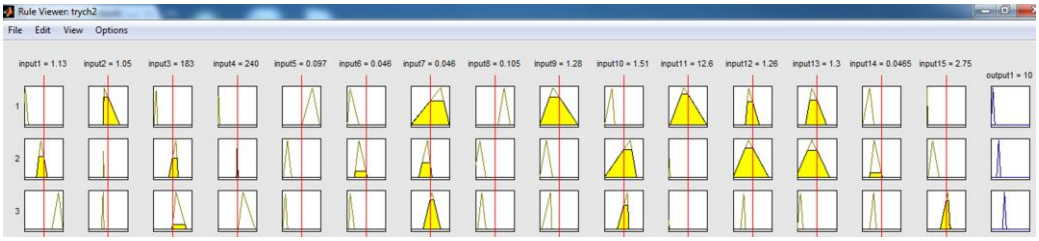
3.4.2 Fuzzy logic classifier

We now test if count =1this mean that character is handwritten and we must use a Fuzzy logic classifier. There are many types of fuzzy neural network, for the forward multi-layer fuzzy neural network, the typical ones are Mamdani model, Sugeno model and Tsukamoto model. The simple and widely used Mamdani model is applied in this paper. For simplicity, as shown in figure10 we assume the following rules in the fuzzy reasoning system of Mamdani model to know the inputting character pictures represent which character in Arabic alphabet as following:

R1: If (input1 in[0.2-0.42]) and (input2 in [1-1.3]) and (input3 in [90-106]) and (input4 in [37-51]) and (input5 in [0.11-0.18]) and (input6 in [0.004-0.02]) and (input7 in[0.003-0.08]) and (input8 in [0.12-0.17]) and (input9 in [5.4-21]) and (input10 in [0.03-0.7]) and (input11 in[1.5-24]) and (input12[0.03-0.2]) and (input13 in [0.7-1.1]) and (input14 in [0.004-0.02]) and (input15 [0.99-1.1]) and (input16 in [0-0.1]) then (output1 is alf) this character is ا.



a



b

Figure 10: (a-b) fuzzy logic rules for the proposed

4. Experiments and Results

We perform many experiments to determine the effective of our system using Matlab7.12.0 (R2011a), where used 9 image for each printed character the first six is all from font "**Arabic transparent**" one of them used to extract feature vector (training) and the other five for testing. The seventh character in font Times New Roman while the eighth and ninth characters from font Arial and simplified Arabic fixed respectively also used to testing system with different font from testing font. All characters were written in size 14 the public recognition rate was 92.9%. The isolated Arabic character we entered to the system all it Arabic character in computer keyboard which its number 36 characters. The table4 displays the recognition results for printed character in our proposed system.

1: refer to correct recognition.
0: refer to error recognition.

Table 4: Recognition results of the proposed system

no.	char	font type								
		Arabic transparent (14) character image name						Timesnewroman (14) testing	Arial (14) testing	Simplified Arabic fixed (14) testing
		a training	B testing	C testing	d testing	e testing	f testing			
1	ا	1	1	1	1	1	1	1	1	1

2	ب	1	1	1	1	1	1	1	1	1
3	ت	1	1	1	1	1	1	1	1	0
4	ث	1	1	1	1	1	1	1	1	1
5	ج	1	1	1	1	1	1	1	1	1
6	ح	1	1	1	1	1	1	1	1	1
7	خ	1	1	1	0	0	0	0	0	0
8	د	1	1	1	1	1	1	1	1	1
9	ذ	1	1	1	1	1	1	1	1	1
10	ر	1	1	1	1	1	1	1	1	1
11	ز	1	1	1	1	1	1	1	1	1
12	س	1	1	1	1	1	1	1	1	0
13	ش	1	1	1	1	1	1	1	1	1
14	ص	1	1	1	1	1	1	1	1	0
15	ض	1	1	1	1	1	1	1	1	0
16	ط	1	0	1	0	1	1	0	1	0
17	ظ	1	0	1	0	1	0	1	0	1
18	ع	1	1	1	1	1	1	1	1	1
19	غ	1	1	1	1	1	1	1	1	1
20	ف	1	1	1	0	0	1	1	1	0
21	ق	1	1	1	1	1	1	1	1	1
22	ك	1	1	1	1	1	1	1	1	1
23	ل	1	1	1	1	1	1	1	1	1
24	م	1	1	1	1	1	1	1	1	1
25	ن	1	1	1	1	1	1	1	1	1
26	ه	1	1	1	1	1	1	1	1	1
27	و	1	1	1	1	1	1	1	1	1
28	ي	1	1	1	1	1	1	1	1	1
29	ء	1	1	1	1	1	1	1	1	1
30	ؤ	1	1	1	1	1	1	1	1	1
31	إ	1	1	1	1	1	1	1	1	1
32	أ	1	1	1	1	1	1	1	1	1
33	لا	1	1	1	1	1	1	1	1	1
34	آ	1	1	1	1	1	1	1	1	1
35	ى	1	1	1	1	1	1	1	1	1
36	ئ	1	1	0	0	0	1	1	1	1
percentage		100%	94%	97%	86%	91%	94%	94%	94%	80%
Public recognition percentage of proposed system = 92.9%										

Table5 and table6 show various experimental results of the multi-layer fuzzy neural network trained for the recognition of handwritten text for two characters are "ا"and "ج". Our proposed system success in recognize the character identity where we get on recognition rate is 100% in training phase and rate 96.6% in testing phase.

Table 5: Recognition results of handwritten"ا"

image/char	1	2	3	4	5	the result
a1.jpg	13	5	2	2	4	correct training
a2.jpg	12	3	2	4	2	correct training
a3.jpg	11	4	1	2	3	correct training
a4,jpg	12	3	2	4	2	correct testing
a5.jpg	8	6	1	2	4	correct testing
a6.jpg	8	4	2	4	5	correct testing

Table 6: Recognition results of handwritten"ج"

image/char	1	2	3	4	5	the result	
e1.jpg	5	8	6	5	10	correct training	
e2.jpg	5	9	6	6	14	correct training	
e3.jpg	4	8	7	7	15	correct training	
e4.jpg	5	9	5	7	14	correct testing	
e5.jpg	6	12	6	7	14	correct testing	
e6.jpg	4	10	5	4	8	error testing	

5. Conclusion

This paper is focused on Arabic character recognition method. The characters are recognizing based on image processing techniques and using K-NN and fuzzy classifier. Our proposed system has ability to recognize character in different font , this consider good step in character recognition where all OCR system depend on the font type while our system is able to passing this difficult with good recognition results which can be explored in futures work to recognize connected character in Arabic text. We shall also try in future work to use another classifier such as Neural Network, Support Vector Machine classifiers and compare it is result with our approach result's. Elapsed time in execute our system to perform recognition process is 0.04 seconds.

References

[1] *M. Fakir, M. M. Hassani,* "On The Recognition Of Arabic Characters Using Hough Transform Techniques", *Malaysian Journal of Computer Science, Vol. 13 No. 2, December 2000.*

[2] M. Zand, A. N. Nilchi,and S. A. Monadjemi,"Recognirion- based Segmentation in Persian Character Recognition" , world Academy of science , Engineering and Technology, 2008.
[3] Hashemi, M., Fatemi, O., & Safavi, R. ,"Persian script recognition". Proceedings of the third Int.Conference on document analysis and recognition. II, 869-873, 1995.
[4] Allam, M. "Segmentation versus Segmentation-free for Recognizing Arabic Text". Document Recognition II, SPIE, 2422, 228-235,1995.
[5] Ethnologue: Languages of the World, 14th ed. SIL Int'l, 2000.
[6] Lorigo, AuthorLiana M., & Govindaraju, Venu , "Offline Arabic Handwriting Recognition": A Survey. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE. 28, 1, 2006.
[7] Umbaugh Scot E, "computer Vision and Image Processing", Prentice Hall, NJ, ISBN 0-13-264599-8, 1998.
[8] H. Zhou, J. Wu, and J. Zhang, "Digital Image Processing Part II", Ventus publishing APS, 2010.
[9] R. C. Gonzalez, R. E. Woods, Steven L."Digital Image Processing Using MATLAB(R)". Prentice Hall, 2003.
[10] L. Shapiro and G. Stockman, "Computer Vision", Prentice Hall, March, 2000.
[11] S. Dey , A. Bishnu , T. Acharya, "A Co-processor for Computing the Euler Number of a Binary Image using Divide-and-Conquer Strategy", Indian Statistical Institute,2007.
[12] S. Umbaugh, "Computer Vision and Image processing: A Practical Approach using CVIP Tools", Prentice-Hall, Inc., USA, 1998.

تمييز الحروف العربية المكتوبة بخط اليد والحروف المطبوعة آلياً وتحويلها إلى نص قابل للتحرير باستخدام مصنفي الجوار الأقرب والمنطق المضبب

زمن فاضل جابر
قسم الحاسبات، كلية علوم الحاسبات والرياضيات، جامعة ذي قار

الخلاصة

في هذا البحث تم اقتراح نظام تمييز ضوئي للحروف العربية المنفصلة وهذا النظام له القدرة على تمييز الحروف المكتوبة بخط اليد والمكتوبة باستخدام الآلة الطابعة حيث يتم تحويل هذه الحروف إلى نص مطبوع قابل للتحرير وذلك باستخدام تقنيات معالجة الصور وأنظمة الذكاء الاصطناعي. وهذه التقنيات استعملت لإيجاد خصائص متينة لكل حرف لتمييزه عن الحروف الأخرى في هذه اللغة. استخدمنا مصنف الجوار الأقرب لتصنيف الحروف المطبوعة آلياً بينما استخدمنا مصنف المنطق المضبب استخدم لتصنيف الحروف المكتوبة بخط اليد. إذ أُدخلت الحروف المطبوعة إلى النظام بخطوط مختلفة (Arabic transparent, Times New Roman, Arial, simplified Arabic fixed) وجميعها بالحجم 14 وذلك لاختبار جودة النظام، كل حرف مطبوع آلياً في الأبجدية العربية أُدخل تسع مرات ست منها بالخط Arabic transparent والثلاث المتبقية كانت بالخطوط Times New Roman, Arial, simplified Arabic fixed على الترتيب، وبذلك نكون قد أدخلنا 324 حرف إلى النظام إذ نجح النظام في تمييز 301 حرف منها وبنسبة تمييز مقدارها 92,9%. بينما تم إدخال 216 حرف مكتوب يدوياً و نجح النظام في التعرف على 208 حرف منها وبنسبة تمييز مقدارها 96,6% .معدل وقت التنفيذ الذي يستغرقه النظام في عملية التمييز للحرف الواحد يقدر بـ 0,04 ثانية.