International Journal of Hybrid Information Technology
Vol.7, No.5 (2014), pp.413-428
http://dx.doi.org/10.14257/ijhit.2014.7.5.38

A Framework for Arabic Handwritten Recognition Based on Segmentation

A. Lawgali[1], M. Angelova[2] and A. Bouridane[2]

[1]*University of Benghazi, Libya*
[2]*Northumbria University,Newcastle upon Tyne, UK*
[1]*ahmed.lawgali@uob.edu.ly*

Abstract

Automatic off-line Arabic handwriting recognition still faces a big challenges. Due to the cursive nature of the Arabic language, most of published works are based on recognition of a whole word without segmentation. This paper presents a new framework for the recognition of handwritten Arabic words based on segmentation. This framework involves two phases (training phase and testing phase). In the training phase, Arabic handwritten characters were trained to be recognized, while in the testing phase, words were segmented into characters for recognition. Classification is achieved in two steps (classification of the segmented characters and classification of the word). A dictionary is constructed and used to correct any errors occurring during the previous stages of the recognition process. This work has been tested with IFN/ENIT database and a comparison made against some existing methods and promising results have been obtained.

Keywords: *Arabic character segmentation*

1. Introduction

Off-line recognition of text plays a significant role in several applications such as the automatic sorting of postal mail or editing old documents. Automatic off-line recognition of text is the ability of the computer to distinguish characters and words. It can be divided into the recognition of printed and handwritten characters. Printed characters have one style and a size for any given font. However, handwritten characters have styles and sizes, which vary both for the same writer and between different writers. Many languages use Arabic characters such as Persian, Urdu and Jawi [1]. However, little research has been carried out in the field of Arabic handwritten character recognition compared to research in Latin and Chinese [2] counterparts. The main problem relates to the cursive (the way successive characters are connected together depending on their positions) nature and its peculiar ligatures (a prevalent glyph which replaces two or more characters) of Arabic script. This makes the segmentation of words into individual characters a difficult task [3]. Despite attempts to apply methods developed for cursive Latin to Arabic, it is generally insufficient to segment Arabic text [2]. In addition, the ligature increases the difficulty of segmentation, which does not allow algorithms developed for other scripts to be applied to the Arabic script [3]. Several studies have proposed methods based on recognition of the whole word without segmentation [4-11]. Alma'adeed *et al.* [12] presented a system for recognition of handwritten Arabic words based on Hidden Markov Model (HMM). El-Hajj *et al.* [13] proposed a system using baseline dependant features and HMM for classifying handwritten Arabic words. Alma'adeed [5] introduced a system using Artificial Neural Network (ANN) for unconstrained handwritten Arabic words. Alkhateeb *et al.* [7] presented a technique for the recognition of handwritten

ISSN: 1738-9968 IJHIT
Copyright ⓒ 2014 SERSC

Arabic words where Discrete Cosine Transform (DCT) is used for extracting features of the word. These features are then fed into a neural network for classification. Other studies have assumed the characters are already segmented, in order to avoid the segmentation process [14, 15]. In addition, algorithms [16, 17] which split the words into characters have also been developed. Zheng *et al.* [18] introduced a method using a vertical histogram to segment printed Arabic character. Hamid and Haraty [19] proposed an over-segmentation algorithm for segmenting handwritten Arabic texts based on features such as holes, end points and corner points. A neural network is then used to reject or accept the segmentation points. Sari *et al.* [20] introduced a method based on morphological rules analysis of a word in order to extract segmentation points. Lorigo *et al.* [16] proposed a technique for over-segmenting the word and using the knowledge of character shapes to reject extra segmentation points. Shubair *et al.* [21] presented an approach to identify the curves and strokes of words and used special conditions to extract segmentation points. Wshah *et al.* [17] introduced an algorithm for the segmentation of handwritten Arabic words based on the concept of skeletons and contours. Husam *et al.* [22] presented a technique based on segmenting a word into primitives and then use a neural network for validating segmentation points based on certain features, such as directions. Most of the currently proposed segmentation algorithms do not solve the problem of overlapping characters in Arabic handwriting and also do not attempt to recognize the words after their segmentation. The segmentation stage is the most difficult stage and is main source of errors in recognition. The process of segmentation still remains a challenge in text recognition and new techniques to improve it are needed [2]. Recognition of whole words without segmentation has limitations because it just can classify the trained words, while the recognition based on segmentation does not have this restriction. In this paper, a new framework for handwritten Arabic recognition based on segmentation is presented which include dictionary for verification of the result. The framework includes two phases: training phase and testing phase.

The organization of the paper is as follows. Section 2 discusses the challenges and motivation for the proposed segmentation technique. Section 3 describes in detail the proposed framework while Section 4 discusses the results and their analysis. Section 5 gives the conclusion and some future work.

2. Challenges and Motivation

Arabic script is written from right to left and is composed of 28 characters, with no capital or lower case. Each character has two or four shapes, the shape of the character depending on its position in the word. The dots play a significant role in Arabic characters. The shape of some characters is similar but the difference arises with position and number of dots, such as (ب, ت, ث), which can occur either above or below the characters. Two characters in the alphabet have three dots, three have two dots and ten have one dot. Dots may appear as two distinct dots or may be connected into a line in handwritten text. Furthermore, short marks such as a '"hamza", can be placed above or below five particular characters or can appear as isolated characters. Some Arabic characters have a loop, such as (ص, ف, و). The cursive nature of Arabic text means that characters of a word are connected through an imaginary horizontal line called a baseline. Also, there are lines which appear above and below the baseline, called ascenders and descenders, as shown in Figure 1 [1].

 Copyright ⓒ 2014 SERSC

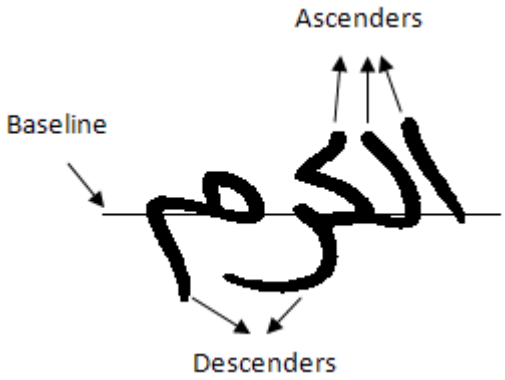
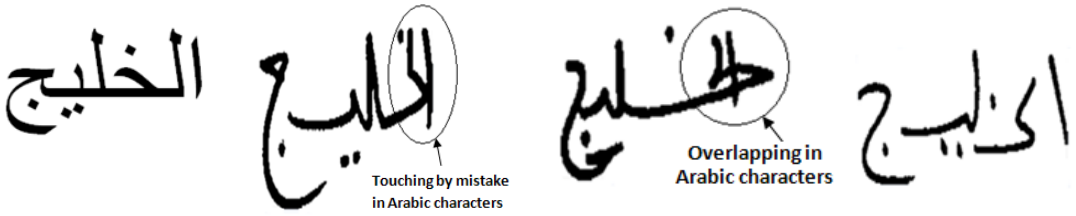


Figure 1. Baseline, Ascenders and Descenders as shown in a Word

In addition, six characters do not connect to a subsequent character in a word and this causes a separation of the word into parts. These parts are called sub-words. Spaces separate words and short spaces separate sub-words. There are several challenges as shown in Figure 2 to segment handwritten Arabic word into characters, due to the cursive nature of Arabic text. These include:



(a) Printed word (b) Handwritten word (1) (c) Handwritten word (2) (d) Handwritten word (3)

Figure 2. Three Versions of Handwritten Arabic Word Illustrating the Challenges in Segmentation

- The same word in handwritten Arabic word could have several styles and sizes. Figure 2(a) shows a printed Arabic word, while Figure. 2(b), 2(c), and 2(d) show the same word but handwritten, each in a different calligraphic style.
- Two or more characters in handwritten Arabic language can be combined vertically and presented by different shapes as shown in Figure. 2(c). This overlap between the neighbouring characters is called a ligature. It means that the second character might appear before the first one in some cases [1].
- In some cases, two characters may touch by mistake as shown in Figure. 2(b).

Moreover, in handwritten text, different characters may appear to be similar and it is difficult even for the human eye to find the difference [23]. There are differences between the length and the width of Arabic characters, for example (ا, ب). Also, the same character may appear differently in its various forms, such as (ع, عـ) [2]. Moreover, the great similarity between some of the handwritten characters makes the classification after segmentation process another challenge as shown in Figure. 3.

Copyright ⓒ 2014 SERSC

Figure 3. Similarity between the Characters (Faa ـفـ and Ain ـعـ)

3. A Framework based on Segmentation

The proposed framework for handwritten Arabic recognition based on segmentation involves two phases: training phase and testing phase as illustrated in Figure 4.

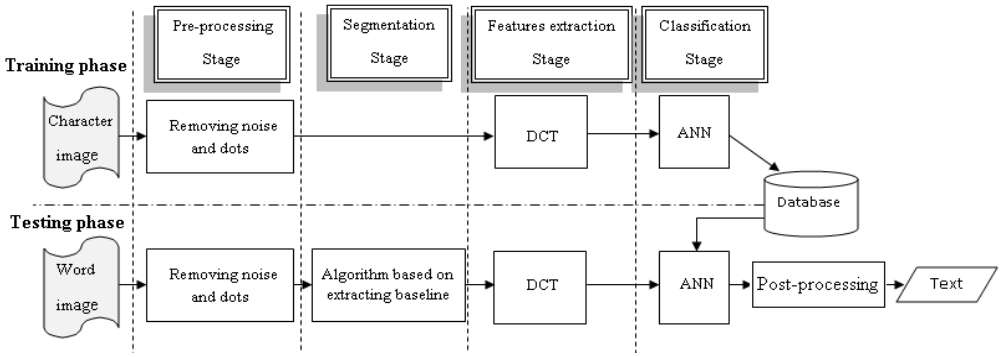


Figure 4. The Proposed Framework for Handwritten Arabic Recognition based on Segmentation

3.1. Training Phase

In this phase, Arabic handwritten characters are trained to be recognized. In the training phase our database HACDB is used containing 5800 shapes of handwritten Arabic characters [24] where 800 overlapping shapes of Arabic characters were added. The overlapping characters are added as one shape as shown in Figure. 5. HACDB is designed for this application to cover all shapes of Arabic characters and was written by 50 writers with their age ranging between 14-50. The training phase consists of three stages: pre-processing, feature extraction and classification.

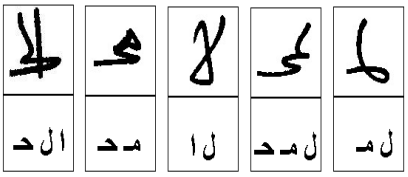


Figure 5. Illustration of Overlapping in Arabic Characters Forms

3.1.1. Pre-processing Stage

Pre-processing attempts to remove the details that have no discriminative power in the process of recognition (*i.e.*, redundant data). Firstly, images of characters are converted into a binary format: values of background pixels as 1 (white) and values of foreground pixels as 0 (black). The small objects (not part of writing considered as noise) are also removed. The dots

 Copyright ⓒ 2014 SERSC

and marks, such as "hamza" are removed from the characters since they can affect the classification and also in order to reduce the number of classes used. In handwriting, the thickness and sizes of characters usually vary. Before extracting the features of the characters, the original characters are normalized by normalizing the thickness and sizes. This is achieved by extracting the skeletons of characters which are then increased at a steady rate for all characters to achieve normalization. The images of characters are resized as *128x128* for normalization purposes. Figure 6 illustrates the difference in thickness and size of the characters before and after normalization.

(a) Before Normalization **(b) After Normalization**

Figure 6. Normalization of Characters

3.1.2. Feature Extraction Stage

Due to similar appearance of some different characters, the selection of the method for feature extraction remains the most important step for achieving a high recognition accuracy. In this paper, Discrete Cosine Transform (DCT) is to capture discriminative features of Arabic handwritten characters. DCT is widely used in the field of digital signal processing applications [25].

- **Discrete Cosine Transform:**

DCT is a technique to convert data of the image into its elementary frequency components characterized by DCT coefficients [25]. DCT coefficients *f(u,v)* of an image of *m* rows and *n* columns *f(m,n)* are computed by:

$$f(u,v) = \propto(u) \propto(v) \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \cos\left[\frac{(2m+1)\pi u}{2M}\right] \cos\left[\frac{(2n+1)\pi v}{2N}\right] \quad \textbf{(1)}$$

where

$$\propto(u) = \begin{cases} \frac{1}{\sqrt{M}}, & u = 0 \\ \sqrt{\frac{2}{M}}, & 1 \le u \le M-1 \end{cases}$$

$$\propto(v) = \begin{cases} \frac{1}{\sqrt{N}}, & v = 0 \\ \sqrt{\frac{2}{N}}, & 1 \le v \le N-1 \end{cases}$$

DCT clusters high value coefficients in the upper left corner and low value coefficients in the bottom right of the array *(m,n)*. The higher value DCT coefficients are then extracted in a zigzag fashion and stored in a vector sequence, see Figure. 7. By applying DCT, an image of a character is represented by this vector. Thus, one of the main characteristics of DCT is its

Copyright ⓒ 2014 SERSC

ability to convert the energy of the image into a few coefficients [7]. The numbers of DCT coefficients chosen in the classification stage was 250 coefficients for *128x128* rather than all coefficients for all images as discussed in our paper [24. These features are utilized for recognition in the classification stage.

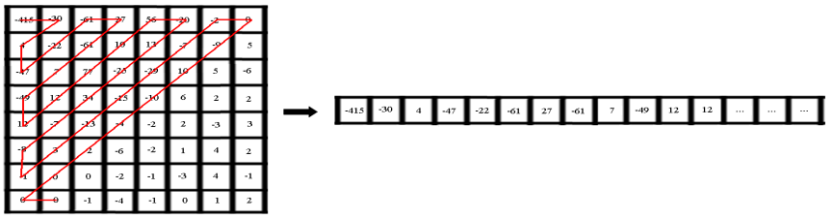


Figure 7. Rearranging DCT Coefficients from Zigzag Order into One Vector

3.1.2. Classification Stage

A classifier is used to identify the characters by using their features obtained by applying DCT which are compared and saved as models for the trained classes. Features of an unknown character segmented from a word in the testing phase will be extracted and compared with the features of the training models to identify the unknown character. Common methods in the classification stage are an artificial neural network (ANN) [26].

- **Artificial Neural Network (ANN)**

ANN is a nonlinear system which is used widely for problems which are not explicitly formulated, such as the pattern classification [1, 27]. It has been used to deal with the features that have been extracted from the characters. ANN consists of processing elements with weights which are learned from the training data. Three layers were used in this research for the architecture of the network: input layer, hidden layer and output layer. Figure 8 depicts example of the architecture of 3-layer ANN. The input layer is fed by the features of the characters. Therefore, the number of nodes in this layer depends on the number of input features of the network. The last layer is the output layer and the number of its nodes is based on the desired outputs. The hidden layer lies between the input and output layers. The feed forward network multi-layer perceptron (MLP) back propagation (BP) with a supervised training algorithm is used in this work. It is the best known paradigm of training the neural network to classify patterns [27]. A classifier is used to identify the characters by using their features obtained by applying DCT. These are then compared and saved as models for the training stage.

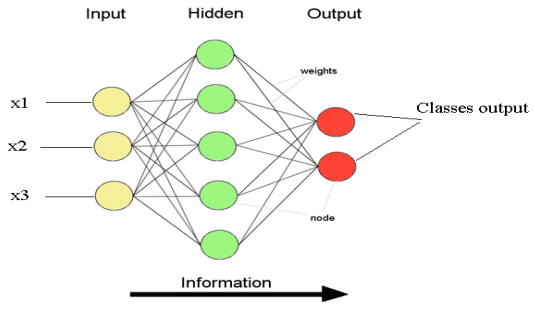


Figure 8. Example of the ANN Architecture with 3-layer

Copyright ⓒ 2014 SERSC

Due to the similarity of some Arabic handwritten characters which makes their recognition difficult. In some cases, the classification depends on contextual information to recognize the character. To reduce the number of similarities between the characters, Arabic characters are divided into four groups (isolated characters, at the beginning of the word, in the middle of the word and at the end of the word). Each group is trained independently. This reduces the similarities between these groups and increases the rate of accuracy.

3.2. Testing Phase

In the testing phase, Arabic handwritten words were segmented into characters and then the features of these characters extracted in order to recognize them. Words were read from the IFN/ENIT database written by 411 different writers [28] containing 26459 handwritten Tunisian town/village names, and consisting of about 212,000 characters and 115,000 pieces of Arabic words.

3.2.1. Pre-processing Stage

Pre-processing was applied to remove the details that have no discriminative power in the process of recognition (*i.e.*, are redundant). Noise removal and binarization were carried out in the development of the database. The dots and marks, such as 'hamza' were removed from the words since they can affect the baseline extraction [29].

3.2.2. Segmentation Stage

As described in [16], the segmentation points are identified at the end of a character and the beginning of the next one and are usually located in the region surrounding the baseline. Our algorithm [30] to extract segmentation points is applied in this paper. Improvements have been added to the algorithm such as detecting the dots and location of the character and solving the problem of space between parts of a single character. The algorithm consists of the following tasks:

- **Words sub-words Segmentation**

There are six characters which do not connect to a subsequent character in a word as it causes a separation of the word into sub-words. Therefore, this process depends on spaces in the word segmented into sub-words. However, in some cases there is a space (or uncompleted line) between parts of a single character as shown in Figure. 9(a).

(a) Incorrect Case **(b) Correct Case**

Figure 9. Correct and Incorrect Cases for Character ط

The algorithm has been improved to solve the problem of space in a single character in Figure 9 (a), by assuming that there is no space in a single character except if there is a part above it (like the vertical line). Each sub-word is stored as an image and the order of sub-words starts from right to left. After the segmentation process, the small parts which appear are considered as noise and removed. The foregrounds (white areas) were also removed from sub-words where rectangular borders with two pixels around the sub-words are kept. Figure 10 shows how the small parts appear as noise.

Copyright ⓒ 2014 SERSC

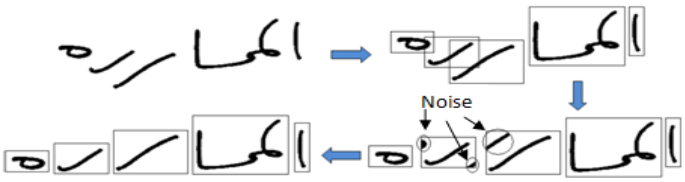


Figure 10. Removing Noise from the Sub-word

- **Baseline Detection**

As most of the connection points between the characters lie on baseline (BL), the BL extraction is important task for character segmentation [1]. As Figure [11] shows, that in handwritten text, the sub-words may not be located on the same line. Therefore, detecting the BL for sub-words, rather than the words containing more than one sub-word, is more efficient and will result in more successful results [29].

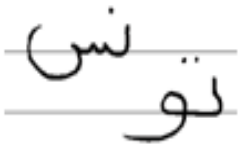


Figure 11. Two Sub-words and Two Baselines

The horizontal and vertical projection of the skeleton of a sub-word is computed by:

$$HP(i) = \sum_{i} P(i,j) \quad VP(j) = \sum_{j} P(i,j) \quad , \tag{2}$$

where *HP(i)* and *VP(j)* are the value of the horizontal and vertical projection, respectively, and *P(i, j)* is the pixel value of the binary image at location *(i, j)*. The first estimation of the baseline (FEBL) is measured by finding the highest peak of horizontal projections of sub-word. The second estimation baseline (SEBL) is measured by calculating the average distance between the branch points (BPs) of the sub-word. The BL is calculated from the average distance between the FEBL and the SEBL as shown in Figure 12.

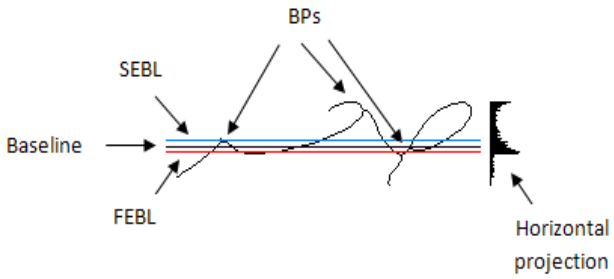


Figure 12. Process of Baseline Detection

Copyright ⓒ 2014 SERSC

- **Extraction of Segmentation Points**

In this task, the descenders of sub-word having a starting point below the BL are deleted. After that, the vertical projection is used to determine candidate points as segmentation points (SPs) by using Equation 2. The method proceeds by using the first estimation segmentation point (FESP), if the candidate point lies in an area close to BL such as points P1, P2, P3 and P5 in Figure 13. If the candidate point is far from the BL threshold level, it is ignored (point P6). The algorithm tests the left side of each SP to determine the number of branches available. If there is not a BP such as P4 in Figure 13 this SP is removed except if the last character is Alif (ﺎ) such as P2. SPs are applied to the original word without diacritics to extract the shapes of characters. For further details of these first three tasks see [30].

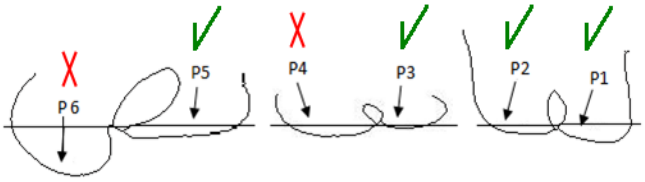


Figure 13. Selection of Segmentation Points

- **Location of Character and Dots:**

To identify dots as parts of characters, the matching correlation coefficient algorithm [31] was applied. The algorithm matches an image of the character with the original image of the word and selects a region of interest in the original image. The dots can occur above or below the characters and their sizes may vary, therefore the area above or below the character is searched to detect presence of dots. The position (above or below) and number of dots are also detected. The location of the character is detected by testing the right side and left side of the character as follows: If the characters do not touch from the right side or left, the shape of the character is isolated. If they touch from the left side, the shape of the character is at the beginning of the word. If touching occurs from the right side only, the shape of the character is at the end of the word. Otherwise, it is in the middle of the word if there touching from both sides. Figure 14 illustrates the dots and location of the character. Moreover, the position of the character in the word (first, second, ...*etc*) is calculated. The information (such as position and number of dots, position of the character in the word, and the shape of character) is saved to assist in the recognition of the character.

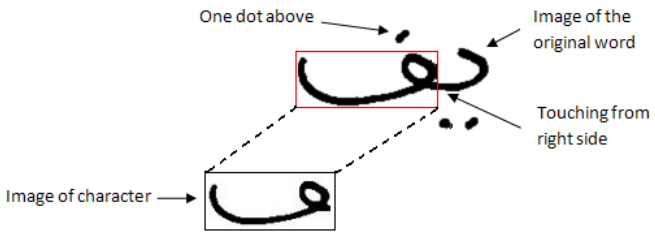


Figure 14. The Dots and Location of the Character

Copyright ⓒ 2014 SERSC

3.2.3. Feature Extraction Stage

After the segmentation process, the normalization process in section 3.1.1 is applied to ensure that an unknown character segmented from the word has the same attributes (*i.e.*, size) as characters trained. 250 DCT coefficients are extracted from the unknown character after segmentation and passed to ANN to the classification stage.

3.2.3. Classification Stage

This stage consists of the following two tasks: classification of the character and classification of the word. In the first task, an ANN is used to identify the shape of unknown character by using its features obtained by applying DCT. The unknown segmented character is classified into its group (isolated characters, characters at the beginning of the word, in the middle of the word and at the end of the word) by testing the right side and left side of the character. Once training is done, in order to classify the shape of a character, 250 DCT coefficients of the character are fed to the ANN as shown in Figure 15. For characters with similar shape where the difference arises with the position and number of dots, (such as ب, ت, ث) occurring either above or below the characters, the character is identified by detecting the number and position of dots of its shape. Figure 16 illustrates the classification of the character.

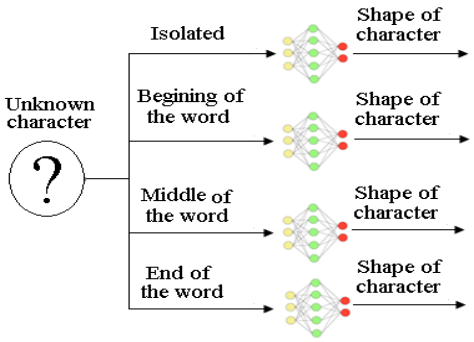


Figure 15. Classification of the Shape of the Character

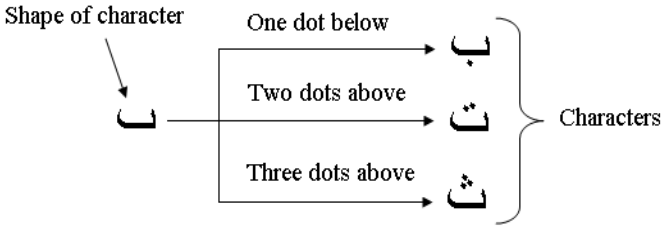


Figure 16. Classification of the Character

- **Post-processing**

In this task, the segmented characters are classified as words. To achieve this, a dictionary is constructed where the characters are submitted to the dictionary with the file name of the word. In some cases, after classifying all segmented characters, there are still some misclassifications. Therefore the dictionary is used to correct any error occurring during the

 Copyright ⓒ 2014 SERSC

recognition process. The dictionary contains all trained words needed for recognition and consists of two tables. The first table contains the characters of the nearest word(s) as well as a label identifying the word while the second one contains the file names and labels of the words designed to ensure a correct recognition of words. Segmented characters of the word are compared with each row in the first table of the dictionary. The label of the row(s) having the higher number(s) of matches is (are) extracted and used in the second table as well as the word in this row considered as output text. It is worth noting that an unknown word may have more than one possible similar correspondent in the database as identified by the highest number of matches. Likewise, the file name of the word and the label that has the highest number of matches are compared in the second table to evaluate the result. Figure 17 illustrates the classification of the word with the dictionary.

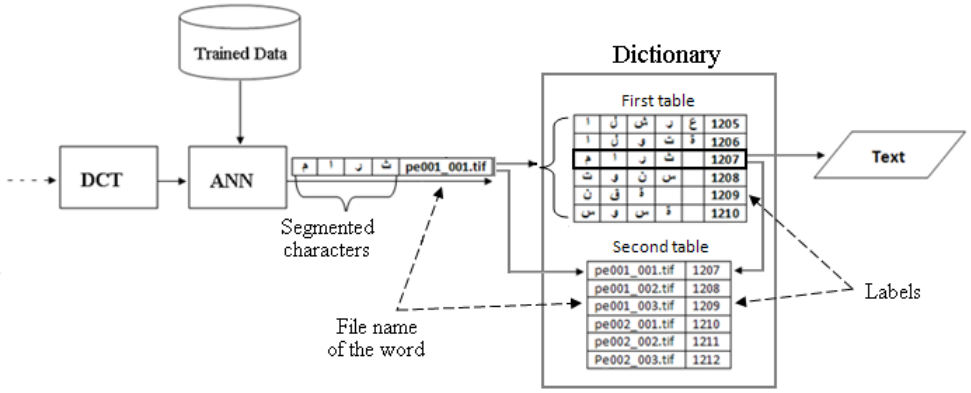


Figure 17. Classification of the Word

4. Experimental Results

Experiments were carried out using two databases. The first database is HACDB with 6,600 handwritten Arabic characters constructed by the authors and used in the training phase. The characters are trained to distinguish the segmented characters of the words. The DCT with ANN is used for the feature extraction of shapes of characters in the classification stage. A set (e) of IFN/INIT database containing 6,033 Arabic handwritten words is used in the testing phase. The experiments were carried out for all stages (segmentation of the word into characters, classification of the segmented characters and classification of the word).

4.1. Segmentation of the Word into Characters

The shape of a segmented character may appear to be similar to another character thus making it difficult to recognize In addition, there are some errors such as segmenting one character into more than one segment as shown in Figure 18. Figure 19 and Figure 20 illustrate correct and incorrect results obtained using our algorithm. Figure 20(a) and Figure 20(b) show one character segmented into more than one part; this causes an increase in the number of segmented characters from the word. Conversely, Figure 20(c) and Figure 20(d) show two characters joining together as one segment; this causes a decrease in the number of segmented characters from the word.

Copyright ⓒ 2014 SERSC

(a) The Shape Appear Similar to more than One (b) Segmenting One Character into more than One Character Segment

Figure 18. Some Factors Affecting the Accuracy of the Result

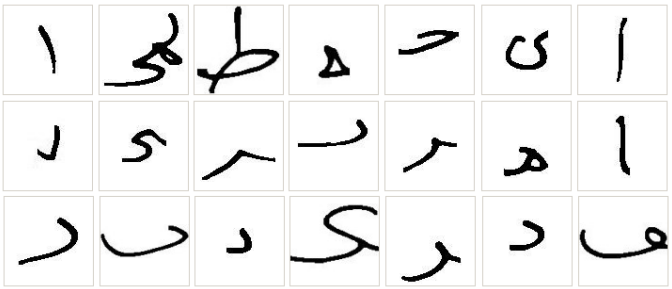


Figure 19. Illustration of Correct Results of Segmentation Algorithm

(a) Character ي Segmented Into Part (b) Character ـسـ Segmented Into Parts

(a) Characters ـا and ط Segmented as One Segment (b) Characters ا and ر Segmented as one Segment

Figure 20. Illustration of Incorrect Results of Segmentation Algorithm

4.2. Classification of the Segmented Characters

After segmenting the word into characters, ANN is used to classify the shapes of unknown characters by using their DCT based features. The characters are recognized by detecting the number and positions of dots on the shapes. Overlapping characters are recognized and correctly classified as illustrated in Figure 21.

 Copyright ⓒ 2014 SERSC

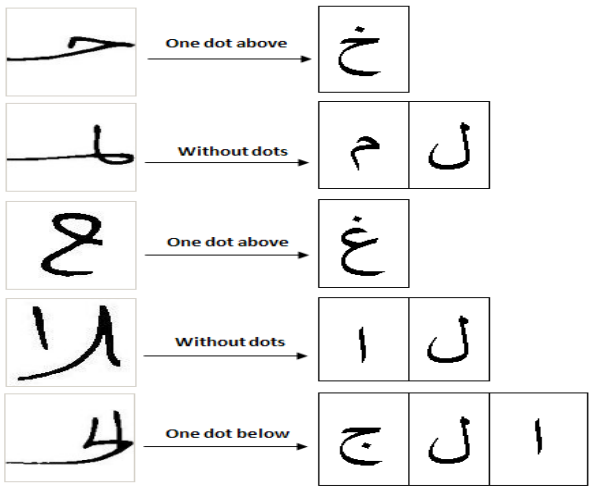


Figure 21. Illustration of the Classification of Overlapping Characters

4.2. Classification of Words

After classifying all characters of the words, the dictionary is used to correct any error(s) occurring during the previous stages of the recognition process. However, IFN/ENIT database contains a number of similar words such as(مثلين, متلين) and (فرنانه, فريانه). The identified characters of a word are compared with each row in the first table of the dictionary. The word of the row(s) that has (have) the highest number of matches is (are) classified as output text. In some cases, if a misclassification of some characters occurs, there may be more than one outputs. Figure 22 illustrates the second and fourth rows of the dictionary having the most matches with the segmented characters resulting in two outputs.

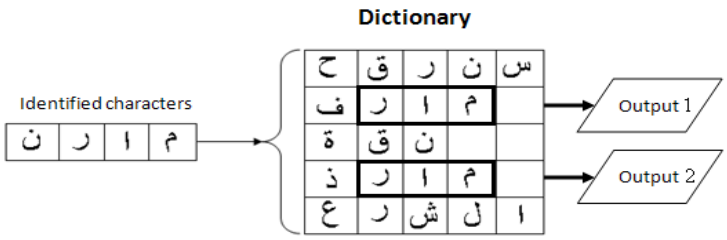


Figure 22. Classification of a Word with Two Possibilities

Table 1 shows our results carried out using set (e) from IFN/ENIT database with 6,033 words. The results show that 85.36% classifications the word correctly with one output, 3.53% with two outputs and 1.84% with three outputs. The overall accuracy of correct classification is 90.73%.

Table 1. Recognition Rate for 6,033 Handwritten Arabic Words

Experiment data	One possibility	Two possibilities	Three possibilities	Overall accuracy
IFN/ENIT database (set e)	85.36%	3.53%	1.84%	90.73%

Copyright ⓒ 2014 SERSC

To our knowledge, most of the currently proposed segmentation algorithms do not attempt to recognize the words after their segmentation. Therefore, our work was compared with other approaches that recognize a whole word without segmentation. These approaches were tested on the same data set and the results of the comparison shown in Table 2. Approach 1 [9] used K Nearest Neighbour classifier to classify handwritten Arabic word. An approach 2 [10] based on HMM for classification, while ANN used in [11]. These approaches identified whole words without segmentation and used IFN/ENIT database for evaluation (sets (a,b,c,d) for training and set (e) for testing). Some handwritten words are very difficult to read/acquire and most of the existing methods fail to efficiently segment them. Furthermore, existing approaches based on recognition of the whole word without segmentation do not achieve high accuracies. One of the most significant advantages of methods for recognition of handwritten words based on segmentation, rather than recognition of the whole word without segmentation, is that it does not have limitation to classify only trained words. Our approach has this advantage and has demonstrated its capability for achieving accurate recognition based on segmentation.

Table 2. Comparison Our Result with Previous Works

Approaches	Accuracy	Method
Approach 1 [9]	76.04	Without segmentation
Approach 2 [10]	83.55	Without segmentation
Approach 3 [11]	89.90	Without segmentation
Our approach	90.73	Based on segmentation

5. Conclusion and Future Work

This paper has presented a framework for the recognition of handwritten Arabic words. The framework proposed involves two phases (training phase and testing phase). In the training phase, the Arabic handwriting characters are used and most highest value DCT coefficients of each character stored as features. HACDB database containing 6,600 shapes of handwritten Arabic characters including overlapping characters was used in this phase. In the testing phase, handwritten Arabic words are segmented. Then DCT and ANN are used for feature extraction and classification, respectively. Classification is achieved in two steps (classification of the segmented characters and classification of the word). A dictionary is constructed and used to correct any error(s) occurring during the previous stages of the recognition process. As future work, the segmentation algorithm should be improved by further investigating the more complex problem and development of a huge database for handwritten Arabic characters containing most shapes of handwritten characters.

References

[1] L. M. Lorigo and V. Govindaraju, "Offline Arabic handwriting recognition: a survey", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 5, **(2006)**, pp. 712–724.
[2] A. M. Zeki, "The segmentation problem in Arabic character recognition the state of the art", First International Conference on Information and Communication Technologies, 2005. ICICT 2005, **(2005)**, pp. 11–26.
[3] A. A. Aburas and M. E. Gumah, "Arabic handwriting recognition: Challenges and solutions", International Symposium on Information Technology, ITSim 2008., vol. 2, **(2008)**, pp. 1-6.
[4] S. Alma'adeed, C. Higgins and D. Elliman, "Off-line recognition of handwritten arabic words using multiple hidden markov models", Knowledge-Based Systems, AI 2003, the Twenty-third SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, vol. 17, no. 2-4, **(2004)**, pp. 75–79.

 Copyright ⓒ 2014 SERSC

[5] S. Alma'adeed, "Recognition of off-line handwritten Arabic words using neural net- work", Geometric Modeling and Imaging-New Trends, 16-18, **(2006)**, pp. 141–144.
[6] I. A. Jannoud, "Automatic Arabic hand written text recognition system", American Journal of Applied Sciences, vol. 4, **(2007)**, pp. 857–864.
[7] J. H. AlKhateeb, R. Jinchang, J. Jianmin, S. S. Ipson and H. El-Abed, "Word-based handwritten Arabic scripts recognition using dct features and neural network classifier", 5th International Multi-Conference on Systems, Signals and Devices, 2008. IEEE SSD 2008, 20-22, **(2008)**, pp. 1–5.
[8] J. H. AlKhateeb, O. Pauplin, J. Ren and J. Jiang, "Performance of hidden markov model and dynamic bayesian network classifiers on handwritten Arabic word recognition", Knowledge-Based Systems, vol. 24, **(2011)**, pp. 680-688.
[9] J. H. AIKhateeb, F. Khelifi, J. Jiani and S. S. Ipson, "A new approach for off- line handwritten arabic word recognition using KNN classifier", IEEE International Conference on Signal and Image Processing Applications, **(2009)**, pp. 191–194.
[10] J. H. AlKhateeb, J. Ren, J. Jiang and H. Al-Muhtaseb, "Offline handwritten Arabic cursive text recognition using hidden markov models and re-ranking", Pattern Recognition Letters, vol. 32, **(2011)**, pp. 1081–1088.
[11] J. H. AlKhateeb, J. Ren, J. Jiang and S. S. Ipson, "Unconstrained Arabic handwritten word feature extraction: A comparative study", Sixth International Conference on Information Technology: New Generations, **(2009)** April, pp. 1655-1656.
[12] S. Alma'adeed, C. Higgens and D. Elliman, "Recognition of off-line handwritten Arabic words using hidden markov model approach", 16th International Conference on Pattern Recognition, vol. 3, **(2002)**, pp. 481–484.
[13] R. El-Hajj, L. Likforman-Sulem and C. Mokbel, "Arabic handwriting recognition using baseline dependant features and hidden markov modeling", Proceedings of the Eighth International Conference on Document Analysis and Recognition, ICDAR '05, Washington, DC, USA, IEEE Computer Society, **(2005)**, pp. 893–897.
[14] A. Asiri and M. S. Khorsheed, "Automatic processing of handwritten Arabic forms using neural networks", in IEC (Prague), **(2005)**, pp. 313–317.
[15] A. Mowlaei, K. Faez and A. T. Haghighat, "Feature extraction with wavelet transform for recognition of isolated handwritten Farsi/Arabic characters and numerals", 14th International Conference on Digital Signal Processing, 2002. DSP 2002, vol. 2, **(2002)**, pp. 923–926.
[16] L. Lorigo and V. Govindaraju, "Segmentation and pre-recognition of Arabic handwriting", Proceedings of the Eighth International Conference on Document Analysis and Recognition, vol. 2, **(2005)**, pp. 605–609.
[17] S. Wshah, S. Zhixin and V. Govindaraju, "Segmentation of Arabic handwriting based on both contour and skeleton segmentation", 10th International Conference on Document Analysis and Recognition, 2009. ICDAR '09., 26-29, **(2009)**, pp. 793–797.
[18] L. Zheng, A. H. Hassin and X. Tang, "A new algorithm for machine printed Arabic character segmentation", Pattern Recognition Letters, vol. 25, **(2004)** November, pp. 1723–1729.
[19] A. Hamid and R. Haraty, "A neuro-heuristic approach for segmenting handwritten Arabic text", International Conference on Computer Systems and Applications, ACS/IEEE 2001, **(2001)**, pp. 110–113.
[20] T. Sari, L. Souici and M. Sellami, "Off-line handwritten Arabic character segmentation algorithm: ACSA", in Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition, **(2002)**, pp. 452–457.
[21] A. Shubair, A. Al-Nassiri and A. Rosalina, "Off-line Arabic handwritten word segmentation using rotational invariant segments features", International Arab Journal of Information Technology, vol. 5, no. 2, **(2008)**, pp. 200–208.
[22] H. A. Al-Hamad and R. Abu Zitar, "Development of an efficient neural-based segmentation technique for arabic handwriting recognition", Pattern Recognition., vol. 43, no. 8, **(2010)**, pp. 2773–2798.
[23] D. Motawa, A. Amin and R. Sabourin, "Segmentation of Arabic cursive script", Proceedings of the 4th International Conference on Document Analysis and Recognition, ICDAR '97, Washington, DC, USA, IEEE Computer Society, **(1997)**, pp. 625–628.
[24] A. Lawgali, A. Bouridane, M. Angelova and Z. Ghassemlooy, "Handwritten Arabic character recognition: Which feature extraction method?", International Journal of Advanced Science and Technology, vol. 34, **(2011)**, pp. 1–8.
[25] A. Al-Haj, "Combined dwt-dct digital image watermarking", Journal of Computer Science, vol. 3, no. 9, **(2007)**, pp. 740–746.
[26] O. H. Assma, O. O. Khalifa and A. Hassan, "Handwritten Arabic word recognition: A review of common approaches", International Conference on Computer and Communication Engineering, ICCCE 2008., 13-15, **(2008)**, pp. 801–805.
[27] A. K. Jain, M. Jianchang and K. M. Mohiuddin, "Artificial neural networks: a tutorial", Computer, vol. 29, no. 3, **(1996)** March, pp. 31-44.

Copyright ⓒ 2014 SERSC

[28] M. Pechwitz, S. S. Maddouri, V. Mrgner, N. Ellouze and H. Amiri, "IFN/ENIT - database of handwritten Arabic words", Colloque Inter. Francophone sur lEcrit et le Document, CIFED 2002, **(2002)**, pp. 129–136.
[29] A. AL-Shatnawi and K. Omar, "A comparative study between methods of Arabic base- line detection", International Conference on Electrical Engineering and Informatics, ICEEI 2009, vol. 01, **(2009)**, pp. 73–77.
[30] A. Lawgali, A. Bouridane, M. Angelova and Z. Ghassemlooy, "Automatic segmentation for Arabic characters in handwriting documents", 18th IEEE International Conference on Image Processing (ICIP2011), **(2011)** September, pp. 3529 –3532.
[31] Y. Wu, "Template matching using correlation coefficients," http://www.mathworks.com/matlabcentral/fileexchange/28590-template-matching-using-correlation-coefficients.

 Copyright ⓒ 2014 SERSC