International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 4, August 2021, pp. 3584~3592
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i4.pp3584-3592

Online handwriting Arabic recognition system using k-nearest neighbors classifier and DCT features

Mustafa Ali Abuzaraida[1], Mohammed Elmehrek[2], Esam Elsomadi[3]
[1]School of Computing, College of Arts and Sciences, Universiti Utara Malaysia, Sintok, Kedah, Malaysia
[2,3]Faculty of Information Technology, Misurata University, Misurata, Libya

Article Info

Article history:

Received Aug 24, 2020
Revised Jan 19, 2020
Accepted Mar 5, 2021

Keywords:

Arabic script
Classification
Feature extraction
Handwriting recognition
Segmentation

ABSTRACT

With advances in machine learning techniques, handwriting recognition systems have gained a great deal of importance. Lately, the increasing popularity of handheld computers, digital notebooks, and smartphones give the field of online handwriting recognition more interest. In this paper, we propose an enhanced method for the recognition of Arabic handwriting words using a directions-based segmentation technique and discrete cosine transform (DCT) coefficients as structural features. The main contribution of this research was combining a total of 18 structural features which were extracted by DCT coefficients and using the k-nearest neighbors (KNN) classifier to classify the segmented characters based on the extracted features. A dataset is used to validate the proposed method consisting of 2500 words in total. The obtained average 99.10% accuracy in recognition of handwritten characters shows that the proposed approach, through its multiple phases, is efficient in separating, distinguishing, and classifying Arabic handwritten characters using the KNN classifier. The availability of an online dataset of Arabic handwriting words is the main issue in this field. However, the dataset used will be available for research via the website.

Corresponding Author:

Mustafa Ali Abuzaraida
School of Computing, College of Arts and Sciences
Universiti Utara Malaysia
Sintok, Kedah, Malaysia
Email: abuzaraida@uum.edu.my

1. INTRODUCTION

Online approaches of recognizing handwritten scripts remain an ongoing research issue over the last four decades [1, 2]. Nevertheless, this research area is getting more attention recently due to the widely use of touch screen notebooks, handheld computers, and smart phones around the world [3, 4]. In contrast, in the offline scripts recognition field, the efforts have been more dedicated to comparing the online handwritten recognition approaches for most languages [5-7]. Online text recognition refers to handwritten texts on touch screen or tablet type devices. In essence, online recognition deals with the coordinate pairs of handwriting in real-time while the writing is taking place. Offline recognition is fairly simple, such systems take a complete image of the script from a digital input source, such as a scanner, and binarize it using a threshold technique, so that the image pixels are either on (1) or off (0) for typewritten [8, 9] or handwritten text [10, 11]. Online recognition of Arabic script is a challenging problem for many reasons. Handwritten characters suffer not only from scale, location, and orientation variation, but also from person-dependent deformations that produce a large set of handwriting samples per character [12]. Additionally, writing on a digital platform is not as accurate as writing on paper. There are many applications that require real-time automatic recognition

of handwritten scripts. With the growth in popularity of handheld devices, the number of applications that require automatic text and digit recognition is also increasing. Though keyboard applications exist in these devices, there are many application areas that require users to write on the screen or tablet of the device either through a stylus pen or using the tip of their fingers [13, 14]. The main motivation of this research paper is to design and implement a methodology for real-time Arabic handwriting recognition. Our objective is to achieve a high classification accuracy by proposing a novel approach of Arabic handwriting text pre-processing and segmentation. A lot of research focusing on online recognition has concentrated on English, whereas concentration on Arabic script recognition has been less focused on in previous researches.

This paper aims to propose enhanced solutions to the major challenges of the Arabic handwritten form, which represents size variations, dimensional variations, and irregularity of stylus points. In this paper, a novel methodology for online recognition of Arabic handwritten words in real-time is presented. The technique is based on a directional-based segmentation, enhanced structural features and an efficient classifier. The system is tested on dataset consisting of more than 2500 handwritten Arabic word samples. These words were passed through four consecutive phases to reach the state of separating each character of each sample and classifying them correctly. An average 99.10% accuracy is achieved by the proposed system. This paper is organized as follows: Section 2 introduces the methodology used for the online Arabic handwriting recognition system, Section 3 shows the results and discussion of classification process, Section 4 is the conclusion and future work.

2. PROPOSED METHOD

The architecture of the proposed system is illustrated in Figure 1 and consists of five main phases: In the first phase, raw data is collected accurately using a touching device as coordinates of X and Y. In the second phase, pre-processing phase is applied for removing noise or distortions that are present in the raw data. This phase includes four steps: size normalization, centering, point resampling, and pen or finger movement's direction calculation.

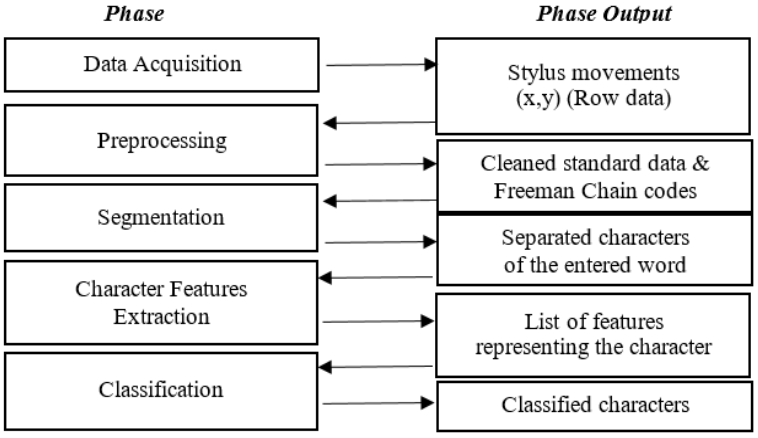

Figure 1. Proposed system phases

In the phase of segmentation, data can be represented by characters or sub-stroke levels. Thus, the nature of every character or sub-stroke can be analysed in individual processes [2, 15]. In the feature extraction phase, features that define each character that are obtained were: Discrete cosine transform (DCT) coefficients that determine each character form, character position within the word and character dots with their position. Finally, the classification phase is done using the k-nearest neighbors (KNN) classifier to recognize the handwritten samples and verify the accuracy of the proposed method. The 5 phases are described below.

2.1. Dataset

The data acquisition phase is the first step of the pattern recognition system [16, 17]. It is used for feeding raw data to be used in subsequent phases to train and test the proposed system [18]. For this phase, a sample of 2500 words were obtained using a software interface to convert each handwritten word to coordinates (x, y) of pen trajectory. This dataset was collected by another research project [19].

2.2. Pre-processing phase

The pre-processing phase is applied to reduce noise and aberrations that may occur in acquired texts because of the limitations of the hardware or software used while writing the text. The noise or aberrations include irregular text size, non-centered text, missing some points of the lines and curves of the texts paths because of the high speeds of writing and uneven distances of points from neighbouring positions [11]. In this phase, several steps are included. Every step performs a specific function for filtering the raw data. Also, it may help to improve the recognition accuracy rate overall and it is considered an important phase of pattern recognition systems. The following steps are applied in the pre-processing phase of the proposed system:

- Size normalization is performed for recognizing any type of script and avoid ambiguity by exchanging the acquired text size into a standard size. The step is done by the (1).

$$S_{new}=S / S\left(x_{max},y_{max}\right) \quad (1)$$

  where S is the coordinates series of x and y.
- After resizing, the coordinates have to be transferred into the centre (x0,y0) to be sure that text is transferred to the same spot relative to the origin. The resizing step is set based on (2).

$$S_{new}=S- S\left(x_{min},y_{min}\right) \quad (2)$$

- Then, the resampling process removes redundant points acquired by the digital device and adds missing points in text trajectories. The resampling procedure is summarized in Figure 2.

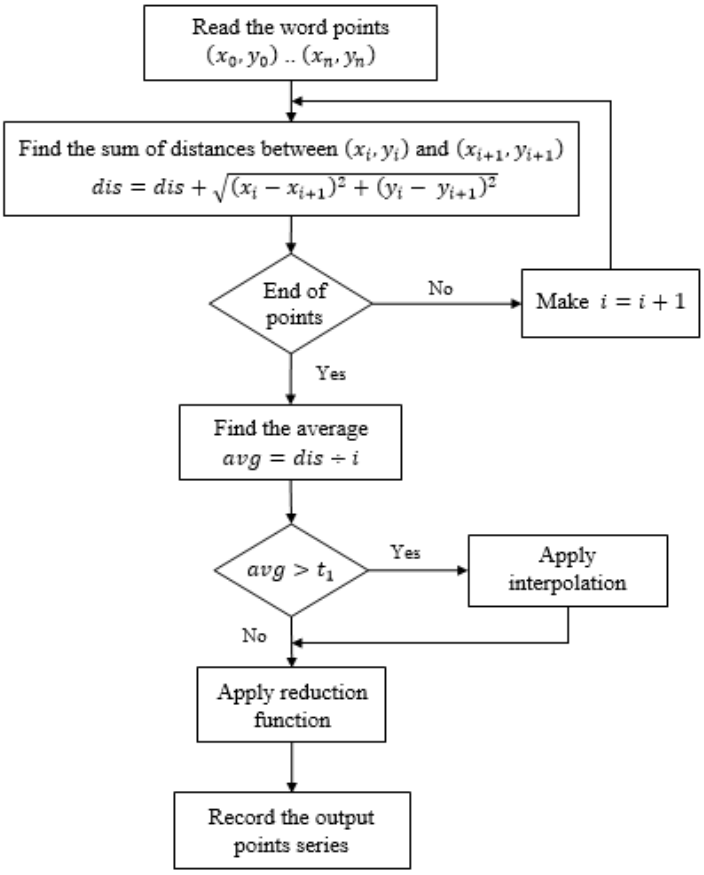

Figure 2. Steps of normalization algorithm

Finally, the Freeman Chain code [20] is performed to represent the direction of movements for every word. This code represents the directions of the pen movements which are listed from 1-8 codes as illustrated in Figure 3. Figure 4 shows the results of performing the pre-processing steps on the word "لـه".

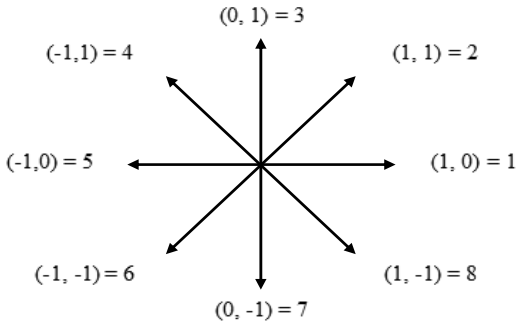

Figure 3. The main writing directions

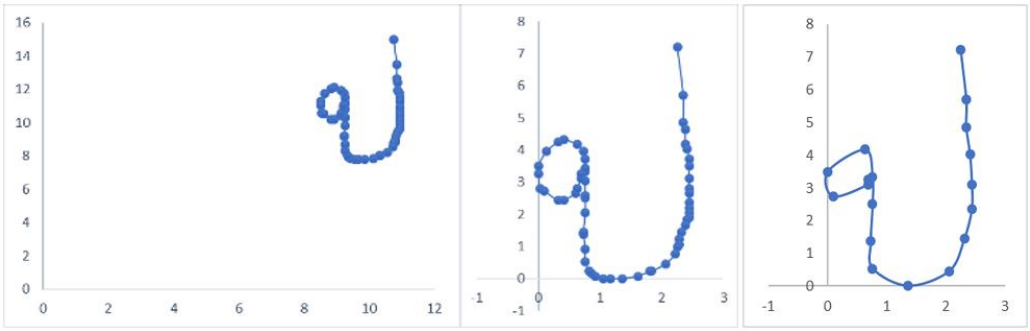

Figure 4. The word "لله" after resizing, centering and resampling processes

2.3. Segmentation

In this phase, data will be presented in isolated characters or sub-strokes. Thus, the nature of every character or sub-stroke is analysed individually. For this purpose, a new segmentation algorithm has been created and developed. First, the handwritten word is divided into a set of strokes that are gathered in groups representing the characters of the written word. In such an operation, Freeman Chain codes were grouped in three sets to describe three main directions: Up, Down and the horizontal direction for left and right. These sets have been labelled as A, B, C areas respectively as shown in Figure 5.

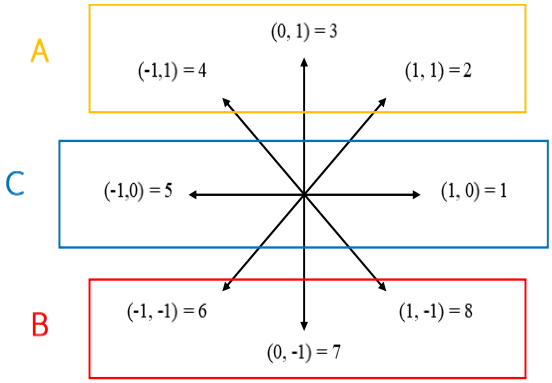

Figure 5. The three directions sets

Strokes are created based on each class of A, B, and C. For A or B classes, points belonging to the same class are grouped into a single stroke. Because these classes (A and B) have opposite directions, they cannot be merged in the same stroke. Points in horizontal class (C) are added to previous stroke if they occur after any vertical class (A or B). However, if a word starts with a sequence of points in class C, they are grouped into a single stroke. Furthermore, if the distance between any two points in the sample is larger than

a specific threshold (t = 2.5), it will be a character dot or a transition from a character to another one. In this case, a new stroke will be created to store the sequence of points that starts from the second point. Figure 6 explains the initial strokes of the word "له" after performing the first part of the segmentation algorithm.

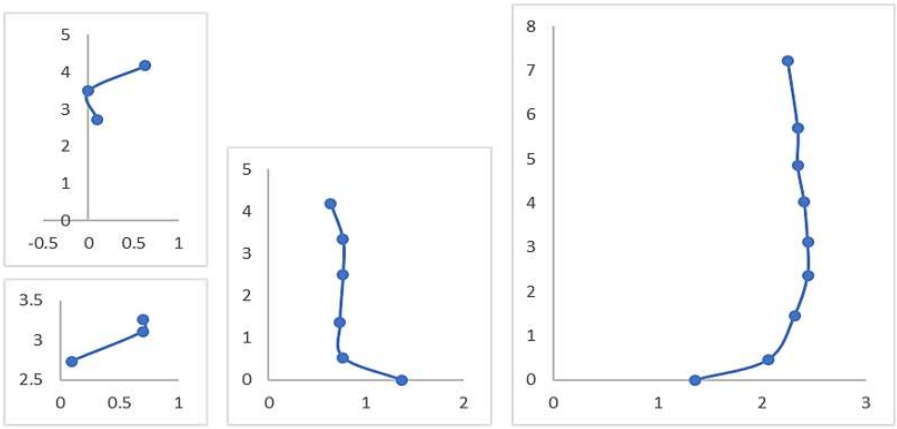

Figure 6. The initial strokes after applying the first part of the segmentation phase

In the second part of the segmentation algorithm, the resulting strokes are filtered and merged or removed according to a set of specific principles explained below. The baseline (writing line), that is a hidden line, describes the position of characters or dots in the word as calculated. By finding out the baseline, final forms of each character can be determined.

First rule, if a stroke contains three points or less and does not relate to any other stroke (in other words, it is not a complementary stroke of other strokes), the stroke will be dots belonging to a character. Otherwise it will be a character or a part of a character. Then, the remaining strokes are passed through a set of tests to identify the form of each stroke and determine its relationship to its adjacent strokes or sometimes distant strokes. For example, a stroke that consists of points in the vertical direction (Class A or B) will be a character Alif ('ـا') or character Lam ('ـل') if the distances between these points are vertical and not far apart. In this case, this stroke has reached its final form. Except for these cases, any stroke is considered a part of a character.

Moreover, if a stroke intersects with another one, these strokes are combined together in a single stroke that will be a loop in a character as in Fa ('فـ'), and Sad ('صـ'). Also, if a stroke has a connection point with the next stroke and they have opposite parts horizontally or vertically, they are grouped together. Examples of characters that have opposite parts such as Noon ('نـ'), Ba ('بـ'), Jeem ('جـ'), Lam ('لـ'), and Lam-Alif ('لا'). These steps are repeated for each stroke until it satisfies all the principles and cannot be merged any more. Figure 7 illustrates the results of performing segmentation algorithm on the previous example "له". After obtaining the final form of characters, the number and position of dots will be calculated. Dots and the state of each character are recorded in text files as numeric codes as explained in Table 1.

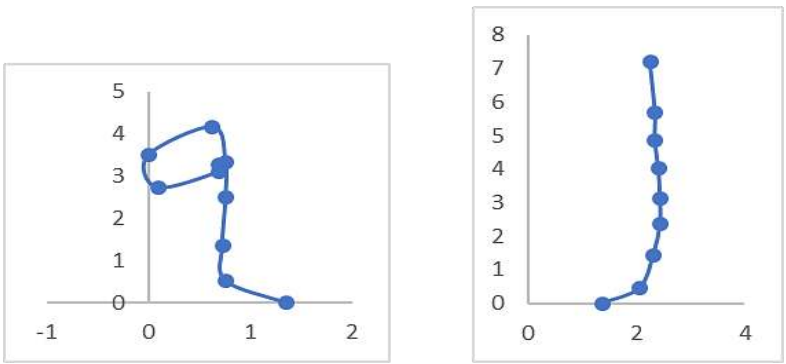

Figure 7. The output of segmentation algorithm on the word "له"

Table 1. The state and dots codes and descriptions

	Code	Description
State	1	The initial form of the character (connected only from the left), i.e. "ـل".
	2	The middle form (connected from the both, left and right), i.e. "ـبـ".
	3	The character is at the end of the word or connected only form the right, i.e. "ـه".
	4	The isolated form (not connected), i.e. "و".
Dots	1	Single dot above the character.
	-1	Single dot below the character.
	2	Two dots above the character.
	-2	Two dots below the character.
	3	Three dots above the character.
	0	No dots in the character.

2.4. Character features extraction

This phase aims to realize that just some features of data points are important to be used in the pattern recognition tasks [21-23]. For this study, frequency domain features x and y signals are used. By performing the segmentation phase, a sequence of strokes which represent characters of a word, will create a result. The sampling points of multiple strokes would look like:

$$\{(x_{s_0 0}, y_{s_0 0}) \dots (x_{s_0 i}, y_{s_0 i})\}, \dots, \{(x_{s_n 0}, y_{s_n 0}) \dots (x_{s_n j}, y_{s_n j})\}$$

In which, each parenthesis indicates a stroke assuming the whole word would have strokes. The sample point's number in each stroke can be more$i$or less$j$samples of the same stroke. In other words, they are not necessarily equal. The discrete cosine transform (DCT) allows it to reconstruct a sequence of accurate from only a few DCT coefficients. It is used for converting the corresponding character signals to frequency domain coefficients [24]. The feature vectors will be constructed using the derivative cosine transform coefficient. This will be considered for each class which was found to be actual and exact for describing the characteristics for every Arabic letter. In fact, signals representing characters typically have the lowest frequency components, so the eight lowest coefficients for all of the axis signals were used for determining the feature vector [24]. Totally, there will be 16 coefficients in the features vector to represent the shape of every character:

$$FV = [X_1, X_2, \dots\dots, X_n, Y_1, Y_2, \dots\dots, Y_n]$$

Additionally, these coefficients are combined with features of state and dots of each character. Therefore, the final number of features used to describe and classify each character would be 18 coefficients.

2.5. Classification

In this study, k-nearest neighbor (KNN) classifier was applied by using Lazy-IBK Weka classifier. This technique is used as the recognizer for classifying and recognizing the Arabic handwritten characters. Before classification, a dataset of features was prepared by 115 class labels that identify each character. Dataset records were 6650 in total represent characters' features of the collected handwritten words. The dataset that was used to train and test the KNN classifier will be described in the next sub-sections.

Any new character can be classified by discovering the k-nearest neighbors among the training data. This process is done by categorizing the candidates of the 115 class after getting each of their weight. Here, a minimum distance function is required for comparing the similarity of writing points. Euclidean distance is used in this classifier within the testing points and all the training points to conclude K smallest Euclidean distances as the nearest neighbors. Equation (3) was used to find the smallest Euclidean distances between two features factors.

$$d(C_{testing}, C_{training}) = \sqrt{\sum_{i=1}^{n} (fv_{testi} - fv_{traini})^2} \quad (3)$$

where $C_{testing}$ is the testing character and $C_{training}$ is the matched character from the training set. Also, fv_{testi} and fv_{traini} are the feature vectors of the testing and training process.

Weka software (Waikato environment for knowledge analysis, version 3.8) was used for implementing the KNN classifier by applying the Lazy-IBK classifier. This environment platform gives an easy way for comparing among a variety of machine learning techniques. The training and testing process

can easily be performed on datasets using a collection of a number of visualization methods and tools. KNN is considered as an instance-based learning example. Here, in the stored training dataset the classification process for any new unclassified case can be classified basically by matching it to similar cases in the training set [25, 26].

In this study, two different test modes were performed. In the first mode, called full training Set mode, the whole dataset is used to train and test the model. In the second mode, called split criterion mode, the classifier was trained on a specific proportion of the records and tested on the remainder. The splitting proportion ranged from 90% to 10% each time. The number of nearest neighbors used by the KNN classifier was set as 1-nearest neighbor. The results and observations of using each of the two test modes of the KNN classifier are explained below.

3. RESULTS AND DISCUSSION

The classification process determines the efficiency of the recognition system and the validity of processes that each phase of the system are comprised of. Therefore, the k-nearest neighbor (KNN) classifier was used as the primary recognition engine. Results obtained in this study were based on two test modes: Full training set and split criterion. In the full training set mode, testing is implemented on the same set the classifier is trained on. Whereas in the split criterion mode, the classifier is trained in a specific proportion of the data initially to build the model, and then to test the model using the rest of the records. Table 2 illustrated the accuracy results of the KNN classifier for each mode.

In the full training set test mode, the whole dataset was used to train and test the model and therefore the classification ratio was high (99.10%). In the split criterion mode, the dataset was divided by different percentages at each time. The best results were obtained when the records were divided into (90%) for training and (10%) for testing. The classifier was able to classify (95.64%) of characters correctly, while the lowest percentage was (91.29%) when the records were divided into (10%) for training and (90%) for testing.

The classifier was able to obtain high classification percentages of not less than (90%) in all cases, and the characters that were incorrectly classified were less than (9%) in the split criterion mode. This indicates that the proposed approach can classify most of the characters properly and also indicates that the phases of pre-processing, segmentation, and features extraction had succeeded in representing the characters properly which enabled the system achieving these results.

The speed of the model building did not exceed (0.02) seconds, whereas, the test period ranged from (0.33) seconds to (0.89) seconds in the split criterion mode and (4.11) seconds in the full training set mode. The precision (P) rates of the classifier for each test mode were above (0.9) as shown in Table 3. The highest rate was (0.993) in the full training set mode, while it was between (0.962) and (0.916) in the split criterion mode. Overall, it can be seen that the KNN classifier achieved a high precision rate and accuracy within a short period of time.

Table 2. KNN classifier accuracy results

Training Set	Testing Set	Correctly Classified	Incorrectly Classified
100%	10%	99.10%	0.90%
90%	20%	95.64%	4.35%
80%	30%	94.66%	5.34%
70%	40%	94.49%	5.51%
60%	50%	94.25%	5.75%
50%	60%	94.29%	5.71%
40%	70%	94.16%	5.84%
30%	80%	93.92%	6.08%
20%	90%	93.29%	6.71%
10%	100%	91.29%	8.71%

Table 3. Performance measures for each test mode

Split Percentage	TP Rate	FP Rate	Precision (P)
100%	0.991	0.001	0.993
90%	0.956	0.001	0.962
80%	0.947	0.001	0.950
70%	0.945	0.001	0.950
60%	0.942	0.001	0.943
50%	0.943	0.001	0.944
40%	0.942	0.001	0.943
30%	0.939	0.001	0.941
20%	0.933	0.001	0.934
10%	0.913	0.002	0.916

4. CONCLUSION

The complexity of online handwritten text recognition is high compared with offline recognition. Therefore, fewer studies and methods have been developed for online handwritten text recognition compared with offline recognition. Online character recognition is used in many applications like, translation using handheld devices, smart boards, and writing on touch screen devices. Many smart devices are currently available in the market which supports applications with handwriting texts. However, Arabic language needs more attention and focus due to it having a few number of applications that support the language as the main reason for having limited applications is the structure of writing Arabic words.

In this research, a dataset contained 2500 handwritten samples represents all Arabic characters' cases. These samples were processed and filtered during the pre-processing phase, and divided into their constituent characters during the segmentation phase. In the feature extraction phase, features of characters were gained using discrete cosine transform coefficients (DCT). Finally, the feature vectors of 18 coefficients were used to determine the characters' class and classify them to one of the predetermined classes using k-nearest neighbors (KNN) classifier.

After obtaining the final results, the KNN classifier was able to classify up to (99.10%) of 6650 characters. Thus, it is possible to say that the proposed system and its algorithms had been able to prove their effectiveness and efficiency in recognizing the characters of the handwritten words and thus recognize these words as a single unit. In regards to this finding, most of the studies test the performances based on private dataset which could not be available publicly. In this study, dataset were obtained from previous research. However, the mentioned studies were conducted without the segmentation step. In this matter, the output of the current study with these studies are not comparable.

After conducting this study on online Arabic text recognition and implementing the system, it is recommended for future studies to improve the segmentation algorithm so it includes different writing modes such as slant writing and overlapping characters. Finally, it can be said that the proposed system through its multiple phases had been able to achieve the desired objectives of this study and can be exploited in other future studies based on its concepts.

REFERENCES

[1] M. A. Abuzaraida, A. M. Zeki, and A. M. Zeki, "Problems of writing on digital surfaces in online handwriting recognition systems," in *2013 5th International Conference on Information and Communication Technology for the Muslim World, ICT4M 2013*, 2013, doi: 10.1109/ICT4M.2013.6518879.

[2] M. M. Albadawi and H. A. Abd Alshafy, "Extended Feature Extraction Technique (Edge Direction Matrixes) For Online Arabic Handwriting Recognition," *Univ. Khartoum Eng. J.*, vol. 8, no. 1, 2018.

[3] M. Abd ElNafea and S. Heshmat, "Efficient Preprocessing Algorithm for Online Handwritten Arabic Strokes," in *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, 2019, pp. 64–69, doi: 10.1109/ITCE.2019.8646643.

[4] B. M. Al-Helali and S. A. Mahmoud, "Arabic Online Handwriting Recognition (AOHR): A Survey," *ACM Comput. Surv.*, vol. 50, no. 3, pp. 1-35, Jun. 2017, doi: 10.1145/3060620.

[5] A. Qaroush, B. Jaber, K. Mohammad, M. Washaha, E. Maali, and N. Nayef, "An efficient, font independent word and character segmentation algorithm for printed Arabic text," *J. King Saud Univ. Inf. Sci*., pp. 1-15, 2019.

[6] A. Lamsaf, M. Aitkerroum, S. Boulaknadel, and Y. Fakhri, "Lines segmentation and word extraction of arabic handwritten text," in *Proceedings of the 3rd International Conference on Smart City Applications*, 2018, pp. 1–7.

[7] A. Lamsaf, M. Aitkerroum, S. Boulaknadel, and Y. Fakhri, "Text Line and Word Extraction of Arabic Handwritten Documents," in *The Proceedings of the Third International Conference on Smart City Applications*, 2018, pp. 492–503.

[8] A. Zoizou, A. Zarghili, and I. Chaker, "A new hybrid method for Arabic multi-font text segmentation, and a reference corpus construction," *J. King Saud Univ. Inf. Sci*., vol. 32, no 5, pp. 576–582, 2018.

[9] N. Tagougui, M. Kherallah, and A. M. Alimi, "Online Arabic Handwriting Recognition: A Survey," *International Journal on Document Analysis and Recognition*, pp. 1–18, 2012.

[10] F. S. Al-Anzi and D. Abu Zeina, "Toward an enhanced Arabic text classification using cosine similarity and Latent Semantic Indexing," *J. King Saud Univ.-Comput. Inf. Sci.*, 2017, doi: 10.1016/j.jksuci.2016.04.001.

[11] J. Ramdan, K. Omar, and M. Faidzul, "A novel method to detect segmentation points of Arabic words using peaks and neural network," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 7, no. 2, pp. 625–631, 2017.

[12] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio. Speech. Lang. Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[13] M. Abuzaraida and A. Zeki, "Online recognition system for handwritten Arabic mathematical symbols," in *2013 International Conference on Advanced Computer Science Applications and Technologies,* 2013, pp. 223–227.

[14] M. A. Abuzaraida and S. M. Jebriel, "The detection of the suitable reduction value of Douglas-Peucker algorithm in online handwritten recognition systems," in *2015 IEEE International Conference on Service Operations And Logistics, And Informatics (SOLI)*, 2015, pp. 82–87.

[15] H. Boubaker, N. Tagougui, H. El Abed, M. Kherallah, and A. M. Alimi, "Graphemes Segmentation for Arabic Online Handwriting Modeling," *JIPS*, vol. 10, no. 4, pp. 503–522, 2014.

[16] M. Abd ElNafea and S. Heshmat, "Novel Databases for Arabic Online Handwriting Recognition System," in *2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE)*, 2020, pp. 263–267.

[17] S. A. Mahmoud, H. Luqman, B. M. Al-Helali, G. BinMakhashen, and M. T. Parvez, "Online-KHATT: An Open-Vocabulary Database for Arabic Online-Text Processing," *The Open Cybernetics & Systemics Journal*, vol. 12, no. 1, pp. 42-59, 2018.

[18] M. A. Abuzaraida, A. M. Zeki, and A. M. Zeki, "Online recognition system for handwritten hindi digits based on matching alignment algorithm," in *Proceedings - 3rd International Conference on Advanced Computer Science Applications and Technologies, ACSAT 2014*, 2014, pp. 168–171, doi: 10.1109/ACSAT.2014.36.

[19] M. A. Abuzaraida, A. M. Zeki, and A. M. Zeki, "Online database of Quranic handwritten words," *Journal of Theoretical and Applied Information Technology,* vol. 62, no. 2, pp. 485–492, 2014.
[20] L.-D. Wu, "On the chain code of a line," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 3, pp. 347–353, 1982.
[21] L. A. Abdul-Rahaim, "Design Proposed Features Extraction Recognition System of Latin Handwritten Text Based on 3D-Discrete Multiwavelet Transform," *Am. J. Electr. Electron. Eng.*, vol. 3, no. 2, pp. 51–63, 2015.
[22] J. M. Alghazo, G. Latif, A. Elhassan, L. Alzubaidi, A. Al-Hmouz, and R. Al-Hmouz, "An Online Numeral Recognition System Using Improved Structural Features–A Unified Method for Handwritten Arabic and Persian Numerals," *J. Telecommun. Electron. Comput. Eng*., vol. 9, no. 2–10, pp. 33–40, 2017.
[23] H. Akram and S. Khalid, "Using features of local densities, statistics and HMM toolkit (HTK) for offline Arabic handwriting text recognition," *J. Electr. Syst. Inf. Technol.*, vol. 4, no. 3, pp. 387–396, 2017.
[24] I. Khodadad, M. Sid-Ahmed, and E. Abdel-Raheem, "Online Arabic/Persian character recognition using neural network classifier and DCT features," in *2011 IEEE 54th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2011, pp. 1–4.
[25] D. T. Larose, "An introduction to data mining," *Trad. Adapt. Thierry Vallaud*, 2005.
[26] A. Lamsaf, M. Aitkerroum, S. Boulaknadel, and Y. Fakhri, "Recognition of Arabic Handwritten Text by Integrating N-gram Model," in *Innovations in Smart Cities Applications Volume 4: The Proceedings of the 5th International Conference on Smart City Applications*, 2021, pp. 1490–1502.

BIOGRAPHIES OF AUTHORS

Mustafa Ali Abuzaraida received his bachelor's degree in (Computer Science) from Misurata University, Misurata, Libya in 2001. He obtained his master degree in (Intelligent System) from University Utara Malaysia (UUM) in 2006 and a Ph.D. in (Computer Science) from International Islamic University Malaysia (IIUM) in 2015. His research interest is on Image processing, Data science, Data mining, Artificial intelligence application, and Natural Language Processing (NLP) mainly on text normalization and sentiment analysis and recognizing online handwritten text. Currently, he is working at School of Computing, College of Arts and Sciences, University Utara Malaysia as international senior lecturer.

Mohammed Elmehrek received the bachelor's degree in (Computer Science) from Misurata University, Faculty of Information Technology in 2017. He is curntly a master student and resesearch assistans. His research area is data maining and Machine learning approaches.

Esam Elsomadi received the bachelor's degree in (Computer Science) from Misurata University, Faculty of Information Technology in 2017. He is curntly a master student and resesearch assistans. His research area is data maining and Machine learning approaches.