Journal of Engineering Science and Technology
Vol. 18, No. 3 (2023) 1581 - 1598
© School of Engineering, Taylor's University

HANDWRITING DETECTION AND RECOGNITION OF ARABIC NUMBERS AND CHARACTERS USING DEEP LEARNING METHODS

AMAR DAOOD*, ALI AL-SAEGH, AHLAM FADHIL MAHMOOD

Department of Computer Engineering, College of Engineering,
University of Mosul, Mosul, Iraq
*Corresponding Author: amar.daood@uomosul.edu.iq

Abstract

Seeing electronic copies of ancient Arabic handwritten books is a great challenge. With the increased interest in recognizing Arabic letters/numbers in various forms and the emergence of deep learning algorithms, it will become possible. To date, the detection and identification of Arabic handwritten letters/numbers still require more work, especially with the scarcity of available data. Therefore, data were first collected in the Arabic language (letters and numbers) by hand in this work, as a contribution to solving the problem of data availability. The diversity of the data collected was adopted in designing the detection technology to correctly localize approximately 18,000 letters/numbers with a detection rate of 99.8%. Then the letters/numbers were automatically extracted from the images by the proposed segment process in preparation for the training procedure. For obtaining higher accuracy, many features were tested (BoW, HOG, SIFT, SURF, and LBPH), as well as many models of deep learning with the adoption of the transfer learning method. Proposed MobileNetV2's deep learning model outperformed the rest, achieving around 94% accuracy in real-time recognition. Thus, in the future, the work can be developed as an application on mobile devices to teach Arabic writing.

Keywords: Arabic handwriting recognition, Deep learning, Object detection, Object recognition, Object segmentation.

1. Introduction

Handwritten detection and recognition are very complicated tasks in the computer vision field. The detection process is used to determine the precise location of the texts or the characters inside the images. On the other hand, handwritten recognition is to identify the written texts or the characters [1, 2]. Although handwriting detection and recognition are big challenges in pattern recognition, these two processes have become very popular and used widely due to the increase in their usage in practical applications and various daily activities [3].

Using handwritten text recognition can vary from an easy task such as checking a scanner at the bank to a complicated mission like scanning old documents to store them in a digital form [4]. One of the main challenges in handwritten characters is the unlimited variety of handwritten styles by individual writers [5]. Additionally, it is a very dependable problem on the different languages.

The handwriting recognition process can be different from one language to another due to the change in the shapes and the attributes of the characters [6, 7], for example, the English language has 26 alphabetical letters while the Arabic language has 28 alphabetical letters.

There are a huge number of researches have been conducted to investigate this problem recently. All these works are designed for specific languages and unfortunately, one particular method is hard to apply to all languages [8]. Recently, many papers have been presented to solve the recognition problems of the English, Latin and Chinese handwriting letters. On the other hand, few works have focused on the Arabic letters, therefore it can be considered a wide-open area and it needs more attention.

The Arabic language is one of the most widely spoken languages in the world, spoken by more than 500 million people and its speakers are distributed in the Arab world as well as to many other regions. It is considered of the 10 most spoken languages in the globe [9]. However, solving Arabic character recognition is much tougher than the English counterpart because many Arabic characters resemble each other closely with slightly different shapes, for example, small lines and dots can change the characters completely. Additionally, the writing direction of the Arabic language is from the right to the left, which is out of usual, especially most in languages from the left to the right. Furthermore, some characters can be written in different shapes according to their position in the word, where the same letters can appear differently if they come in the beginning, middle, or at the end of the word [10].

This work is divided into three stages. The first stage is the data collection process. In his phase, distribute a 120-characters form to 150 people to fill out. Then, these forms were scanned for digital storage. Then the second stage of the detection process, in which the scanned forms were processed by dividing 18,000 images into 60 classes of characters. The final stage is the identification process. Then splitting the collected data into training and testing data. The trained data is used to train some models to perform the recognition and the test data is used to evaluate the trained models. The paper is organized into five sections: section 3 describes the proposed method, data collection, and the detection processes of the Arabic handwriting characters. Section 4 explains the recognition and classification results. Finally, the conclusions and the findings are illustrated in section 5.

2. Related Works

Many researchers have tried to solve the problem of handwritten identification. In [11] the authors proposed a system of Arabic handwriting letters. The system started by converting the images to grayscale form and then smoothing, and a filtering process was applied to remove the noise. The classification part was done by training a CNN network, the system achieved 94%. However, the trained network was trained on 28 alphabetical characters and did not consider the shape variation of individual letters based on their positions. The authors in [12] suggested a system to recognize English handwriting characters with three stages. Image processing is first applied to the input image where they used a webcam as a capturing device. Then, they performed the segmentation process by applying edge detection and dilation. Finally, the recognition process was done using MLP neural network. Although the proposed system was not complicated, the adopted network was trained on raw data of the style of the printed characters which is not efficient for realistic applications.

In [13] Bangla character recognition was proposed using deep learning networks. The paper compared five models to perform the features extraction process. The DenseNet model achieved the highest accuracy which was about 99%. In [14] the researchers used the EMNIST dataset to train a CNN network to perform the recognition of 26 letters, 10 digits, and 12 characters from the Tamil language. The proposed method did not offer a detection algorithm and assumed that all characters were detected properly.

In [15] a mobile app was proposed to perform the detection and recognition process. The segmentation was performed by finding the contours. The recognition was implemented by training a CNN using the EMNIST dataset. The authors achieved 89% of accuracy. In [16] the researchers suggested CNN and RNN networks recognize Pashtu Handwritten Numerals characters. The proposed models achieved 98% accuracy. However, the suggested system did not consider the detection process. In [17] the authors propose transfer learning and genetic algorithms to perform the recognition of the Arabic handwriting texts while the segmentation process was done using the standard optical character recognition technique. The proposed method achieved an accuracy of 92%.

In [18] a new dataset was presented by collecting handwritten samples from 7 to 12 years old children. The authors called their dataset Hijja which contained 29 letters. They trained a CNN model to perform the recognition process. However, the detection process was done manually by human operators. The authors in [19] suggested an offline recognition system for Khmer language handwritten text. the proposed model used a deep learning model to perform the recognition. A convolutional neural network was proposed in [20] to perform the recognition task for Vietnamese handwritten characters. The authors collected 72,500 labelled images of 29 characters from 500 different persons.

The researchers in [21] presented a detection method for Tifinagh characters. The detection system was based on key points features and a convolutional neural network, their OCR detection was tested using (IRCAM) database which contains 33000 images. The authors in [22] suggested different standalone and hybrid CNN architectures recognizing Arabic handwriting characters with the transfer-learning model on the MNIST dataset. In [23] Generative adversarial networks (GANs) were proposed, using two Neural Networks with unsupervised learning for recognizing Arabic handwritten characters. Table 1 summarizes the most important items in many previous works.

Table 1. Brief of some related works in the state-of-the-arts.

Ref.	Database	Language	Numbers/Letters /Word	Detection Method	Recognition Method	Recognition Rate
[18]	Hijja	Arabic	Letters	None	CNN	88%
[22]	Two set MNIST	Arabic	Numbers/Letters	None	CNN + SVM	90%
[23]	AHCD	Arabic	Letters	None	GAN	99.7%
[24]	CASIA-HWDB & ICDAR-2013	Chinese	Letters/word	Segmentation	Neural network	95.88%
[25]	CMATERDB	Arabic/ English	Numbers	None	Customized CNN	99.4%
[26]	CASIA-HWDB & ICDAR-2013	Chinese	letters/word	Otsu's method	DCNN-HMM	96.47%
[27]	USPS& C-Cube& Semeion& EMNIST	English	Numbers/Letters	None	Fukunaga–Koontz Network	98.60%
[28]	EMNIST	English	Numbers/Letters	None	line-segment feature analysis& SVM	98.9%
[29]	Multi-Database/10 datasets	Multilingual/ 8 languages	Numbers	None	CNN-SVM	96.23%
[30]	MINST& CMATERdb & CMATERdb	Arabic/English / Devanagari	Numbers	None	Hybrid orthogonal polynomials	100%
[31]	SDH2019.2	Tai Le	Letters	Binarization	Ensemble deep learning	98.85%
[32]	Meitei Mayek dataset	Meitei Mayek (Manipuri)	Letters	None	3-channel CNN	98.70%
[33]	ADBase& MADBase	Indian	Numbers	None	Ensemble deep transfer learning	99.83%
[34]	MINST&Kaggle alphabet	English	Numbers/Letters	None	Customized CNN	99.642%
[35]	Iranshahr	Persian/Arabic	Letters/word	Contour Line Detection-Gradient Modification	Autoencoder-CNN Networks	91.09%
[36]	AHCC-UCAS2016 & SCUT-COUCH2009	Chinese	Letters	None	End-to-end CNN classifier	98.02%

3. Proposed System

In the proposed work, data are collected and processed for identification.

3.1. Data collection

The first step of the work was to collect the data. There were few public datasets of Arabic handwritten letters which is not enough to contribute properly to this challenging problem. Especially, some of these datasets were collected to solve the numerical digits, and even when the alphabet characters were included, the shape variations based on the position were not considered. For example, Hijja's data set contains only characters and is prepared for training (no hashing required), which is far from the real handwriting data. Furthermore, most of these datasets are stored as separate images for each character, so they do not need pre-processing as the actual data in books requires. For this, letters and numbers were collected in one image for the participant, and then an algorithm was proposed to isolate each of them automatically. While Hijja's, MNIST, and AHCD datasets are prepared for training (no hashing required). To gather data, a form of (10*12) matrix was created, where the first two rows are used to collect numerical digits and the rest of the rows are used to collect alphabet letters. Then, the participants were asked to fill out the form in the same order to obtain a standard filled form, where the order of the letters in the combined forms can make the detection process automated. So that all Arabic letters in their different forms (characters at the beginning of the word, in the middle of the word, and at the end of the word, as well as the unconnected letters) were accommodated.

The designed form is organized as follows: the first row represents ten digits, and these digits are duplicated in the second row, the rest of the rows (10*10) are used to gather 50 characters with their duplicate. In total, 60 categories of 10 numbers and 50 alphabets were provided. Data collection is not an easy task and takes a huge amount of time and effort, especially because it is a peer-to-peer process, and it is necessary to check forms for each person individually and discard incorrectly filled out forms. Finally, samples from 150 participants were collected. The ages of the participants ranged from 18 to 50 years. About 75% of them are right-handed. Then all the collected models were scanned for the purpose of digitizing them with a resolution of 300 dpi, then all the models were converted to PNG images with dimensions of 1700 * 2300 pixels. Figure 1 shows an example of an input form.

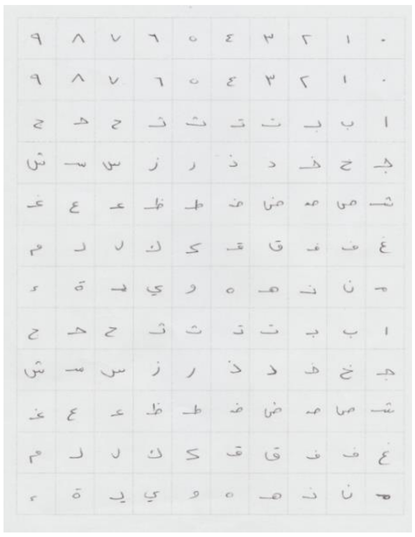


Fig. 1. Sample of the filled handwriting form.

3.2. Handwriting detection algorithm

The second phase of this work is the segmentation process to detect and localize each character in the scanned images to prepare the data for the training process in the third phase to create the prediction models. Firstly, all the scanned images were cropped to remove any unwanted noise, to do that, a graphical user interface was created to ask a user to give two points corners (top-left to the right bottom corner) to eliminate any area which is outside the handwritten tables that consist of a 120 characters cells. This GUI is used to make sure the detection process is performed correctly with minimal human interference and to validate the data subjectively.

The first step in this stage is to convert the images to grayscale and after that, use the thresholding to convert the grayscale to binary images. Figure 2 demonstrates the result of the thresholding process. Then, apply some of the image processing techniques to facilitate the segmentation process. Morphological transformations were used to refine the edges and boundaries of the characters; so an erosion process was applied to smooth and thin the edges of the characters.

The erosion process helps to remove all small and not interconnected background objects which are clearly not regions of our interest. The filtering process is also applied to remove all the small objects. Then, the Dilation process is performed to dilate the foreground object, this process is required to make each character one bulk, especially since some of the Arabic letters have some dots and lines which can be very critical to differentiate them. So, this process is necessary to keep these important details. Figure 3 shows the result of the erosion and the dilation process.

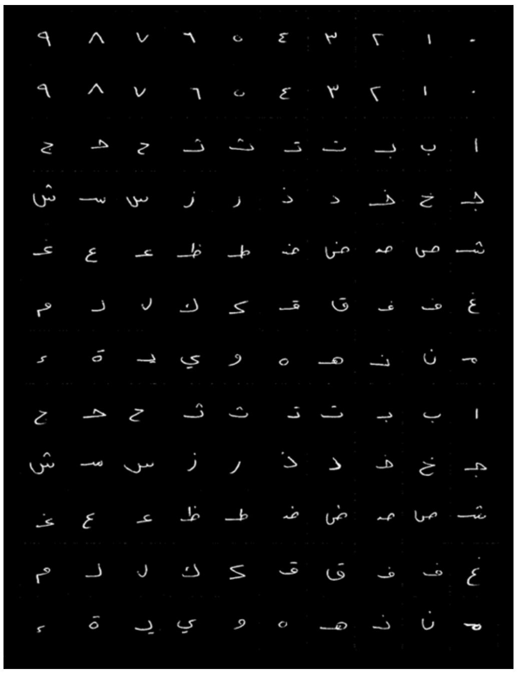


Fig. 2. The resulting image after thresholding.

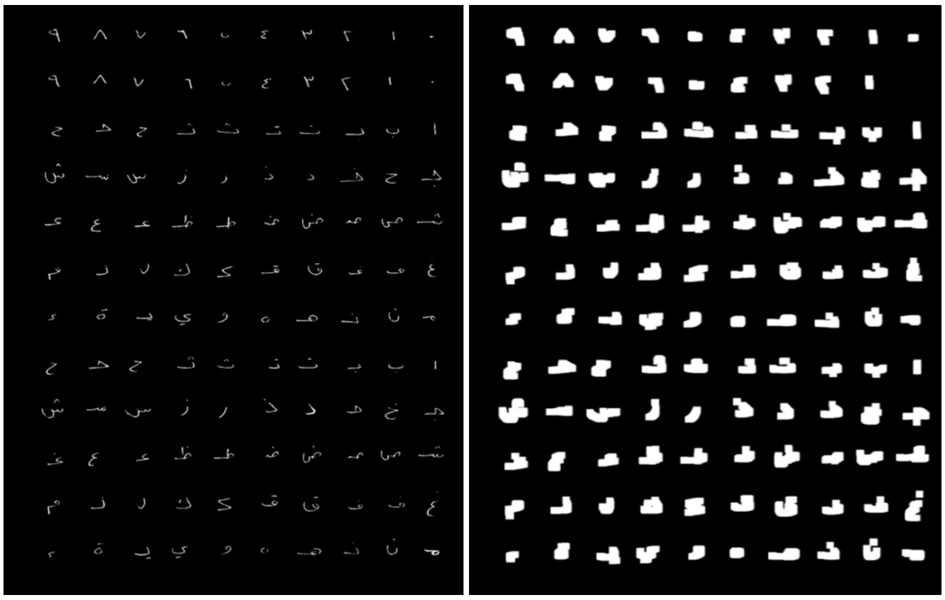
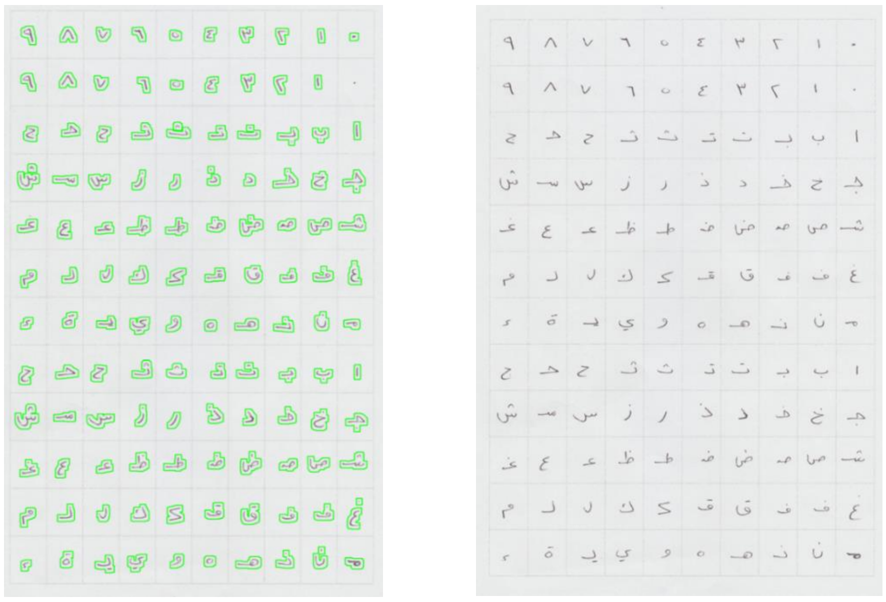


Fig. 3. Output image after: (left) Erosion and (right) Dilation.

To localize the characters, the contours of the resulting images were outlined. The perimeter is a curve that connects the same foreground objects. The contouring process creates X and Y coordinates which represent bounding boxes for the individual detected characters. The result of this process is illustrated in Fig. 4(a). These coordinates can be used to segment the detected characters; however, the contour process may miss some valid detection. As shown in Fig. 4(a), the first character of the second row is an example of this missing character. Therefore, a clustering algorithm was used to group the X and Y coordinates to determine 12 rows and 10 columns for correctly localizing the center of the characters.

K-Means clustering was applied on the Y-axis coordinates, so each row becomes a cluster with 10 characters only. For twelve rows that means the number of clusters is 12, K-means starts with 12 random centroids and each point is assigned to the closest centroid and the new centroid of each cluster is computed, this process continues until the positions of the centroids no longer changes. Then, repeat the clustering process on X coordinates to group 10 clusters. In the end, the clusters of centroids resulting from the aggregation of X and Y coordinates were indicated to obtain 120 centers providing an exact location of the detected character centers. The results of the assembly process are shown in Fig. 4(b). Next, these center coordinates were used to crop all detected characters. Figure 5 shows the procedure for the detection algorithm.

To validate the resulting images subjectively, using the GUI where the user can skim the detected bounding boxes to approve the detected characters. The GUI facilitates the capabilities of picking the wrong detections and cropping them manually by providing two-point corners. By skimming all the 150 scanned, the proposed detection algorithm miss-detected 35 characters that were detected manually by the GUI. This semi-automatic detection method can be able to detect 17965 out of 18000 characters. Thus, the proposed algorithm achieves a 99.8% of detection success rate.



(a) Contour **(b) Centres**

Fig. 4. Image result of the Contour and Centres from clustering the X and Y.

To measure the significant improvement of the clustering process in the detection algorithm, the detection process without the K-means algorithm was repeated. Then, the detection results are validated subjectively using the GUI. This time, the miss-detected are more than 500 characters to achieve less than 97% of detection success rate. The characters are lost due to the participant's writing style, some of the participants tend to write lightly. Additionally, the zero digits (which is the zero version of the Arabic numbering system '.') are represented by a small dot that is mostly getting lost after the filtering process of the erosion transformation. These missing characters can be recovered by the clustering process.

3.3. Handwriting characters recognition method

At the end of the detection phase, about 18,000 images were prepared where these individual images represent Arabic numbers and letters. The size of the images in the cropping process is 100 * 120 pixels. Data was collected for ten digits and about fifty alphabet characters. In the Arabic language, there are about 28 alphabet characters, however, the shape of some of them can be different depending on the position of the character in the word. So, extra classes are added which represent the same characters but with different shapes. For example, the letter Baa may have a different shape as shown in Fig. 6. Therefore, in total, data from 60-class of characters were collected.

Once the data is ready, the goal in this phase is to train a model that can identify 60 classes of characters. The data is divided into two parts, the first part is used for the training process to create a prediction model and this part of the data is called the training data. When the training is done the second part of the data is used to

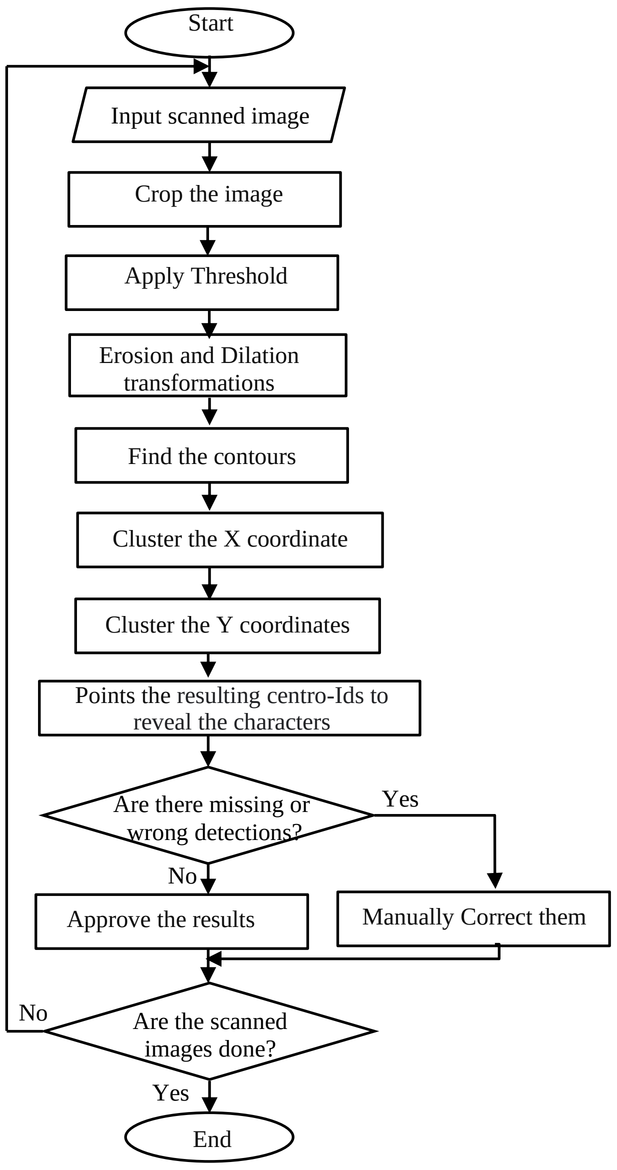

Start
Input scanned image
Crop the image
Apply Threshold
Erosion and Dilation transformations
Find the contours
Cluster the X coordinate
Cluster the Y coordinates
Points the resulting centro-Ids to reveal the characters
Are there missing or wrong detections?
Yes
No
Approve the results
Manually Correct them
Are the scanned images done?
No
Yes
End

Fig. 5. Flowchart of the proposed detection algorithm.

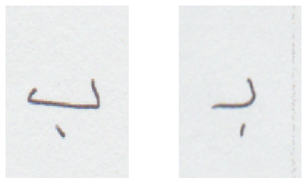


Fig. 6. Two shapes of the Baa letter.

test and perform an assessment evaluation to check the efficiency of the trained model, and this part of the data is called the testing data. Then, dividing the collected data into 80% of training and 20% of testing data. The training process can be done in two different paths. Object classification is an essential task in the computer vision field; thus researchers have put a huge effort to develop and improve the object recognition process. Generally, the techniques of object identification can be divided into two schools. The first one is the traditional school of computer vision which is based on hand-engineering features and then performing the recognition using classification algorithms. Recently, deep learning techniques are spreading widely in the computer vision field. The deep learning schools try to mimic the neurology of the human brain to estimate the optimal features for each particular data.

The traditional algorithms of object recognition are completely dependable on the selected features to describe the characteristics and attributes of the objects which need to be recognized. Additionally, selecting the best classification algorithm is a critical key to solving this problem. However, choosing the optimal features for a particular data is getting the attention of the researcher for a very long time and a huge amount of work has been developed to address this issue. For example, some of these works suggested features designed for hand lettering to preserve the shape of a particular object. These features capture the visual appearance of an object's shape and morphological attributes. On the other hand, texture features are introduced to discriminate objects that have a similar shape and different surface textures. However, texture features are not used to recognize digits or characters because texture characteristics are not very descriptive in terms of handwriting data [37, 38].

To perform the recognition, starting with some of the designed engineering features that describe the shape appearance. The bag of word algorithm is one of the early techniques to recognize objects in computer vision. The recognition process starts with dividing the image into grids and then extracting key-points features that describe the contour and the edges properties of the letters and digits. After creating a set of features vector, the clustering process is used to create a visual word dictionary. This dictionary is used to encode each image and transform the object features to histogram features. In the end, SVM classifier is used to implement the identification process [39, 40]. SVM classifier computes a hyperplane in high-dimensional space to classify the data into two classes. This hyperplane is given by the following equation:

$$F(x) = W^T x + b \quad (1)$$

In this research, the range of features was extended to include more morphological features. Where HOG ((Histogram Oriented Gradient) [41, 42], SURF (Speeded Up Robust Features) [43, 44], SIFT (Scale-Invariant Feature Transform) [45], and LBPH (Local Binary Pattern Histogram) [46, 47] features were used to preserve the shape attributes of the characters. Then, the SVM classifier was used to implement the recognition process. Features are extracted from the training data with the SVM classifier. And the testing data is used to assess the estimation of the trained classifiers. Experiments results are demonstrated in the results section.

Deep learning techniques offer promising results and impressive performance in computer vision tasks. A convolution neural network (CNN) is used mainly to perform image recognition. CNN models can learn the optimal features and classifier at the same time. Researchers and companies have devolved different architectures and proposed many models that can be used for recognition such as VGG16, VGG19 [48], Resenet50 [49], and Inception networks [50]. To compare the traditional methods with the deep learning models, transfer learning algorithms were used to take advantage of the gained knowledge from pre-trained models, where these models can be used as a base classifier to perform feature extractions. VGG16 networks are proposed as a base model. After removing the last layer of VGG16, the resulting network output was flattened, adding 1024 fully connected neurons. Another layer is connected of 512 neurons. Moreover, the leakage layer is attached with a factor of 0.5 to reduce the impact of the problem of overfitting. Finally, we added a SoftMax layer with 60 output nodes to represent the character classes in the proposed dataset. Figure 7 illustrates the proposed CNN model. Then, the training data was used to retrain the new resulted architecture model to perform a fine-tune process to optimize the network parameters. The test data is also used to evaluate the prediction model. The algorithm 1 summarized the steps of the recognition using deep learning in Fig. 8.

The results have shown the superiority of the deep learning model over the traditional algorithms. Deep learning can be able to learn better features to capture the characteristics and attributes of the collected dataset. Thus, extending the proposed experiments result to include more network models for achieving better accuracy. Using transfer learning to get the previous knowledge from the following pre-trained networks as the base model: VGG19, Resnet50, InceptionV3 [51], Xception [52], InceptionResNetV2 [53], MobileNetV2 [54], DenseNet121 [55], and NASNetLarge [56]. By repeating the same process of the fine-tuning technique, remove the last layer of each model and attach two fully connected layers of 1024 and 512 respectively. Then, adding a drop layer to decrease the overfitting effect. And added the output layer of the 60 classes. The training dataset is used to retrain the resulted architectures and the test data is used for validation. The results of these experiments are shown in the results section.

4. Results

The proposed work was initiated with experimental results by implementing the conventional methods to perform the classification. Start with the bag of visual word techniques to perform the identification process. BOW method encodes the image into histogram features to describe the image`s characteristics. With extracted some selected features to capture the shape variation of the characters. And used the following features: Histogram oriented gradient (HOG), scale-invariant feature transform (SIFT), speeded up robust features (SURF), and local binary pattern histogram (LBPH). After that, trained with the SVM classifier to create a prediction model. Secondly, the deep learning model was designed to compare with the traditional methods of image recognition. By utilizing the transfer learning to take advantage of the previous knowledge of VGG16. The results of these comparisons are shown in Table 2.

Furthermore, expanded the deep learning domain to include more pre-trained models to find a better representation of the optimal features of the proposed collected dataset, with nine models as the base classifier. Then, modified the network architectures by adding two more fully connected layers with one drop-out layer at the end of the networks. An output layer of 60-class was also attached to identify the 60-class of characters. Finally, retrain the models and test them using the training data and the test data. The results of these models are shown in Table 3.

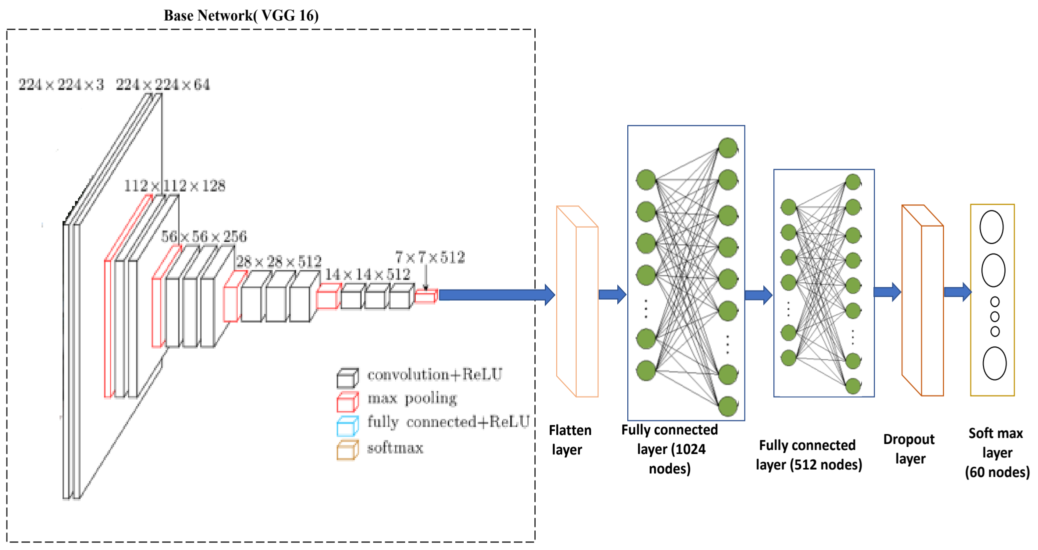


Fig. 7. The proposed CNN architecture.

Algorithm 1: pseudo code of the recognition method using deep learning

```
1:  Start.
2:  Input : d= the collected dataset.
3:  Output : M= trained models, C= matrix of accuracy and F-score for each model
4:  Divide the dataset into 80% training and 20 % testing.
5:  For k In list of  pre-trained models do :
6:      Remove the last layer of the base model.
7:        Add a fully connected layer of 1024 nodes with dropout layer.
8:        Add another fully connected layer of 512 nodes with dropout layer.
9:       Complete the designed network with soft-max layer of 60 outputs.
10:       For i In data training images  do:
11:               Retrain the whole network with the collected images to perform fine-tuning process.
12:               Apply backpropagation to update the weights of the designed network.
13:               Compute the objective function to check the stopping criteria
14:        End for
15:       For j In data testing images  do:
16:               Use the trained model to recognize the test image.
17:               save the predication results
18:       End for
19:       Compute the accuracy and F-score of the trained model.
20:       Evaluate the performance of the proposed model.
21:       save the trained model
22: End for
23: End
```

Fig. 8. Pseudo code of the deep learning recognition algorithm.

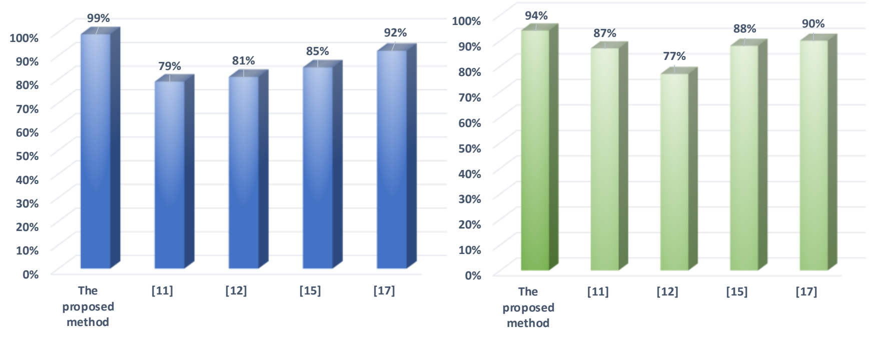
Table 2. Comparisons results of the traditional methods with the deep learning model to perform the classification.

Method	Accuracy	F Score
BOW	71.56%	70.22%
HOG	79.98%	79.07%
SIFT	75.66%	74.97%
SURF	73.48%	72.89%
LBPH	76.93%	76.02%
Deep learning model (VGG16)	91.27%	90.87%

Table 3. Comparison of deep learning models to perform the classification.

Base Model	Accuracy	F Score
VGG16	91.27%	90.87%
VGG19	92.81%	91.66%
Resnet50	86.93%	87.02%
InceptionV3	87.34%	87.12%
Xception	89.45%	88.79%
InceptionResNetV2	93.19%	92.66%
MobileNetV2	94.32%	93.78%
DenseNet121	92.09%	91.69%
NASNetLarge	91.47%	90.82%

Accuracy and F score are calculated as evaluation metrics to check the performance of the trained models. As presented in Table 2, the deep learning training model outperformed the results of the traditional methods of the recognition of handwriting characters. On the other hand, MobileNetV2 was the best base model in terms of characteristics representation, where the learned features of the trained network model achieved 94.32% accuracy and 93.78% F score. To benchmark the results of the proposed detection and recognition algorithms, some of the methods in the previous works are re-implemented. By applying the methods of the detection and the recognition of handwriting characters presented in [11, 12, 15, 17] on the proposed collected datasets. The comparison results are plotted in Figs. 9(a) and (b).



(a) Detection **(b) Recognition**

Fig. 9. Comparison of detection and recognition success rates with existing methods.

As demonstrated in [18] of Hijja data, the proposed model was used on their data to increase accuracy from 88% to 91% without detection result which depending on storing each character as a separate image. Additionally, two MNIST dataset used in [22] Arabic handwritten digits dataset which does not need the segmentation technique was tested. The proposed deep learning model improved the accuracy from 90 to 93.5%.

5. Conclusions

In this paper, a detecting algorithm for Arabic handwritten numbers and characters was proposed as a first step to handwriting Arabic text. Real data is collected, and a method is proposed to automatically segment letters/numbers. Then, testing different features with various deep-learning models to get the highest accuracy.

The suggested detection method is tested utilizing 150 collected forms on (Macbook pro 16gRAM). The forms are collected by 150 participants and each form is organized to write 120 characters. The 120 characters are divided into 20 digits and 100 characters of 10-class of digits and 50-class of letters. Thus, in total 18,000 images 60-class of characters are collected. The detection algorithm uses a threshold process and morphological transformations. Then, the contour is computed to localize the individual characters. After that, the clustering process is used to find the locations of the detected characters precisely. The clustering algorithm boosts the detection rate by 3%.

The proposed detection technique achieves about 99.8% of detection success rate. Furthermore, the generated data was used to train different models to perform handwritten character recognition.

The results demonstrated that the learned features from the deep learning model have better performance in terms of accuracy. The transfer learning technique is used to modify pre-trained models to make their architectures more suitable to identify the proposed collected dataset. The modified version of MobileNetV2 has achieved the highest accuracy of 94.32% and due to its good performance on mobile and tablet devices, the proposed work for handwriting learning can be applied in the kindergarten classroom or elementary school.

References

1. Li, Z.; Xiao, Y.; Wu, Q.; and Lu, H. (2020). Deep template matching for offline handwritten Chinese character recognition. *The Journal of Engineering*, 4, 120-124.
2. Shinde, S.S.; Pang, P.S.; and Bhattacharyya, D. (2021). A literature review on: handwritten character recognition using machine learning algorithms. *Smart Technologies in Data Science and Communication*, 210, 173-182.
3. Baldominos, A.; Saez, Y.; and Isasi, P. (2019). A survey of handwritten character recognition with mnist and emnist. *Applied Sciences*, 9(15), 3169.
4. Purohit, A.; and Chauhan, S.S. (2016). A literature survey on handwritten character recognition. *International Journal of Computer Science and Information Technologies* (*IJCSIT*), 7(1), 1-5.
5. Memon, J.; Sami, M.; Khan, R.A.; and Uddin, M. (2020). Handwritten optical character recognition (OCR): A comprehensive systematic literature review (SLR). *IEEE Access*, 8, 142642-142668.

6. Ali, A.A.A.; Suresha, M.; and Ahmed, H.A. (2019). Different handwritten character recognition methods: a review. *Proceeding of the* 2019 *Global Conference for Advancement in Technology* (*GCAT*). Bangalore, India, 1-8.
7. Singh, S.S.; and Karayev, S. (2021). Full page handwriting recognition via image to sequence extraction. *Proceeding of the ICDAR* 2021*:* 16*th International Conference*. Lausanne, Switzerland, 55-69.
8. Melnyk, P.; You, Z.; and Li, K. (2020). A high-performance CNN method for offline handwritten Chinese character recognition and visualization. *Soft Computing,* 24(11), 7977-7987.
9. Mohammed, D.A.; Mezher, A.A.; and Hadi, H.S. (2019). Off-line handwritten character recognition using an integrated DBSCAN-ANN scheme. *Indonesian Journal of Electrical Engineering and Computer Science*, 14(3), 1443-1451.
10. Djaghbellou, S.; Attia, A.; Bouziane, A.; and Akhtar, Z. (2020). Local features enhancement using deep auto-encoder scheme for the recognition of the proposed handwritten Arabic-Maghrebi characters database. *Multimedia Tools and Applications*, 81(22), 31553-31571.
11. El-Sawy, A.; Loey, M.; and El-Bakry, H. (2017). Arabic handwritten characters recognition using convolutional neural network. *WSEAS Transactions on Computer Research*, 5(1), 11-19.
12. Gulati, I.; Vig, G.; and Khare, V. (2018). Real time handwritten character recognition using ANN. *International Journal of Engineering Sciences & Research Technology*, 7(4), 357-362.
13. Alom, M.Z.; Sidike, P.; Hasan, M.; Taha, T.M.; and Asari, V.K. (2018). Handwritten Bangla character recognition using the state-of-the-art deep convolutional neural networks. *Computational Intelligence and Neuroscience*, Volume 2018, Article ID 6747098 , 1-13.
14. Kishan, S.A.; David, J.C.; and Femi, P.S. (2019). Handwritten character recognition using CNN. *International Journal of Research and Analytical Reviews* (*IJRAR*), 5(3), 241-245.
15. Shubham, S.M.; Shivam, S.; Saransh, G.; Sayam, D.; Monika, J.; and Rahul, S. (2019). Handwritten text recognition: with deep learning and android. *International Journal of Engineering and Advanced Technology* (*IJEAT*), 8(3), 819-825.
16. Khan, K.; Roh, B.H.; Ali, J.; Khan, R.U.; Uddin, I.; Hassan, S.; Riaz, R.; and Ahmad, N. (2020). PHND: pashtu handwritten numerals database and deep learning benchmar. *Plos one*, 15(9), e0238423, 1-19.
17. Balaha, H.M.; Ali, H.A.; Youssef, E.K.; Elsayed, A.E.; Samak, R.A.; Abdelhaleem, M.S.; Tolba, M.M.; Shehata, M.R.; Mahmoud, M.R.A.; Abdelhameed, M.M.; and Mohammed, M.M. (2021). Recognizing Arabic handwritten characters using deep learning and genetic algorithms. *Multimedia Tools and Applications*, 80, 32473-32509.
18. Altwaijry, N.; and Al-Turaiki, I. (2021). Arabic handwriting recognition system using convolutional neural network. *Neural Computing and Applications*, 33(7), 2249-2261.
19. Annanurov B.; and Noor N. (2021). A compact deep learning model for Khmer handwritten text recognition. *IAES International Journal of Artificial Intelligence*, 10(3), 8938-8938.

20. Truong Q.V.; Le H..; and Nhan N.T. (2020). Vietnamese handwritten character recognition using convolutional neural network. *IAES International Journal of Artificial Intelligence*, 9(2), 276-283.
21. Boutounte M.; and Ouadid Y. (2021). Characters recognition using keys points and convolutional neural network. *Indonesian Journal of Electrical Engineering and Computer Science*, 22(3), 1629-1634.
22. Albattah W.; and Albahli S. (2022). Intelligent Arabic handwriting recognition using different standalone and hybrid CNN architectures. *Applied Sciences*, 12(19), 10155.
23. Alwaqfi Y.M.; Mohamad M.; and Al-Taani A.T. (2022). Generative adversarial network for an improved Arabic handwritten characters recognition. *International Journal of Advances in Soft Computing & Its Applications*, 14(1), 176-195.
24. Wu, Y.-C.; Yin, F.; and Liu, C.-L. (2017). Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models. *Pattern Recognition*, 65, 251-264.
25. Ashiquzzaman, A.; and Tushar, A. (2017). Handwritten Arabic numeral recognition using deep learning neural networks. *Proceeding of the* 2017 *IEEE International Conference on Imaging, Vision & Pattern Recognition* (*icIVPR*). Dhaka, Bangladesh, 1-4.
26. Wang, Z. R.; Du, J.; Wang, W. C.; Zhai, J. F.; and Hu, J. S. (2018). A comprehensive study of hybrid neural network hidden Markov model for offline handwritten Chinese text recognition. *International Journal on Document Analysis and Recognition* (*IJDAR*), 21(4), 241-251.
27. Gatto, B.B.; Eulanda, M.S.; Kazuhiro, F.; Waldir, S.J.; and Kenny V.S. (2020). Fukunaga–Koontz convolutional network with applications on character classification. *Neural Processing Letters,* 52(1), 443-465.
28. Kim, C.M.; Ellen, J.H.; Kyungyong, C.; and Roy, C. P. (2020). Line-segment feature analysis algorithm using input dimensionality reduction for handwritten text recognition. *Applied Sciences*, 10(19), 6904.
29. Gupta, D.; and Bag, S. (2021). CNN-based multilingual handwritten numeral recognition: a fusion-free approach. *Expert Systems with Applications*, 165, 113784.
30. Abdulhussain, S.H.; Basheera, M.M.; Marwah, A. N.; Muntadher, Q. A.; Roslizah, A.; and Al-Haddad, S.A. (2021). A robust handwritten numeral recognition using hybrid orthogonal polynomials and moments. *Sensors*, 21(6), 1999.
31. Guo, H.; Yifan, L.; Doudou, Y.; and Jingying, Z. (2021). Offline handwritten Tai Le character recognition using ensemble deep learning. *The Visual Computer*, 38, 3897-3910.
32. Inunganbi, S.; Prakash, C.; and Khumanthem, M. (2021). Handwritten Meitei Mayek recognition using three ‐ channel convolution neural network of gradients and gray. *Computational Intelligence*, 37(1), 70-86.
33. Alkhawaldeh, R.S.; Moatsum, A.; Nawaf, F.A.; Wafa'Za'al, A.; and Ammar, A. (2022). Ensemble deep transfer learning model for Arabic (Indian) handwritten digit recognition. *Neural Computing and Applications*, 34(1), 705-719.

34. Saqib, N.; Khandaker F.H.; Venkata P.Y.; and Ahmed, A. (2022). Convolutional-neural-network-based handwritten character recognition: an approach with massive multisource data. *Algorithms*, 15(4), 129.
35. Khosravi, S.; and Abdolah, C. (2022). Recognition of Persian/Arabic handwritten words using a combination of convolutional neural networks and autoencoder (AECNN). *Mathematical Problems in Engineering*, Volume 2022, Article ID 4241016, 1-15
36. Hu, M.; Xiwen, Q.; Jun, H.; and Xuangou, W. (2022). An end-to-end classifier based on CNN for in-air handwritten-chinese-character recognition. *Applied Sciences*, 12(14), 6862.
37. Balaha, H.M.; Hesham A.A.; and Mahmoud B. (2021). Automatic recognition of handwritten Arabic characters: a comprehensive review. *Neural Computing and Applications*, 33(7), 3011-3034.
38. Dhivya, S.; and Usha, G.D. (2021). Study on automated approach to recognize characters for handwritten and historical document. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 20(3), Article No.: 37, 1-24.
39. Csurka, G.; Dance, C.R.; Fan, L.; Willamowski, J.; and Bray, C. (2004). Visual categorization with bags of keypoints. *Workshop on Statistical Learning in Computer Vision*, *ECCV*, 15(1), 1-22.
40. San Biagio, M.; Bazzani, L.; Cristani, M.; and Murino, V. (2014). Weighted bag of visual words for object recognition. *Proceeding of the* 2014 *IEEE International Conference on Image Processing* (*ICIP*). Paris, France, 2734-2738.
41. Dalal, N.; and Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceeding of the* 2005 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (*CVPR'*05). San Diego, CA, USA , 886-893.
42. Kitayama, M.; and Kiya, H. (2019). HOG feature extraction from encrypted images for privacy-preserving machine learning. *Proceeding of the* 2019 *IEEE International Conference on Consumer Electronics-Asia* (*ICCE-Asia*). Bangkok, Thailand, 450-461.
43. Bay, H.; Tuytelaars, T.; and Van, G.L. (2006). Surf: Speeded up robust features. *Proceeding of the European Conference on Computer Vision*, *ECCV* 2006. Berlin, Germany, 404-417.
44. Vardhan, A.H.; Verma, N.K.; Sevakula, R.K.; and Salour, A. (2015). Unsupervised approach for object matching using speeded up robust features. *Proceeding of the* 2015 *IEEE Applied Imagery Pattern Recognition Workshop* (*AIPR*). Washington, USA, 1-8.
45. Lowe, D.G. (2004). Distinctive image features from scale-invariant key points. *International Journal of Computer Vision*, 60(2), 91-110.
46. Ojala, T.; Pietikainen, M.; and Maenpaa. T.S. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 971-987.
47. He, Y.; Sang, N.; and Huang, R. (2011). Local binary pattern histogram based Texton learning for texture classification. *Proceeding of the* 2011 18*th IEEE International Conference on Image Processing*. Brussels, Belgium, 841-844.

48. Simonyan, K.; and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *Proceeding of the* 3*rd International Conference on Learning Representations* (*ICLR*2015). San Diego, USA, 1-14.
49. He, K.; Zhang, X.; Ren, S.; and Sun, J. (2016). Deep residual learning for image recognition. *Proceeding of the* 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*). Las Vegas, USA, 770-778.
50. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. (2015). Going deeper with convolutions. *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition*. Boston, MA, USA, 1-9.
51. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *Proceeding of the* 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*). Las Vegas, USA, 2818-2826.
52. Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. *Proceeding of the* 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, USA , 1251-1258.
53. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; and Alemi, A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. *Proceeding of the AAAI*-17*: Thirty-First AAAI Conference on Artificial Intelligence*. San Francisco, California USA, 1-12.
54. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. *Proceeding of the* 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, USA, 4510-4520.
55. Huang, G.; Liu, Z.; Der Maaten, L.V.; and Weinberger, K.Q. (2017). Densely connected convolutional networks. *Proceeding of the* 2017 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*). Honolulu, USA, 4700-4708.
56. Zoph, B.; Vijay, V.; Jonathon, S.; and Quoc, V.L. (2018). Learning transferable architectures for scalable image recognition. *Proceeding of the* 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, USA, 8697-8710.