Computer Systems
Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2023.033902

ARTICLE

Fireworks Optimization with Deep Learning-Based Arabic Handwritten Characters Recognition Model

Abdelwahed Motwakel[1,*], Badriyya B. Al-onazi[2], Jaber S. Alzahrani[3], Ayman Yafoz[4], Mahmoud Othman[5], Abu Sarwar Zamani[1], Ishfaq Yaseen[1] and Amgad Atta Abdelmageed[1]

[1]Department of Computer and Self Development, Prince Sattam bin Abdulaziz University, AlKharj, 16278, Saudi Arabia

[2]Department of Language Preparation, Arabic Language Teaching Institute, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh, 11671, Saudi Arabia

[3]Department of Industrial Engineering, College of Engineering at Alqunfudah, Umm Al-Qura University, Makkah, 24211, Saudi Arabia

[4]Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, 21589, Saudi Arabia

[5]Department of Computer Science, Faculty of Computers and Information Technology, Future University in Egypt, New Cairo, 11835, Egypt

*Corresponding Author: Abdelwahed Motwakel. Email: a.ismaeil@psau.edu.sa

Received: 01 July 2022 Accepted: 07 November 2022 Published: 13 September 2024

ABSTRACT

Handwritten character recognition becomes one of the challenging research matters. More studies were presented for recognizing letters of various languages. The availability of Arabic handwritten characters databases was confined. Almost a quarter of a billion people worldwide write and speak Arabic. More historical books and files indicate a vital data set for many Arab nations written in Arabic. Recently, Arabic handwritten character recognition (AHCR) has grabbed the attention and has become a difficult topic for pattern recognition and computer vision (CV). Therefore, this study develops fireworks optimization with the deep learning-based AHCR (FWODL-AHCR) technique. The major intention of the FWODL-AHCR technique is to recognize the distinct handwritten characters in the Arabic language. It initially pre-processes the handwritten images to improve their quality of them. Then, the RetinaNet-based deep convolutional neural network is applied as a feature extractor to produce feature vectors. Next, the deep echo state network (DESN) model is utilized to classify handwritten characters. Finally, the FWO algorithm is exploited as a hyperparameter tuning strategy to boost recognition performance. Various simulations in series were performed to exhibit the enhanced performance of the FWODL-AHCR technique. The comparison study portrayed the supremacy of the FWODL-AHCR technique over other approaches, with 99.91% and 98.94% on Hijja and AHCD datasets, respectively.

KEYWORDS

Arabic language; handwritten character recognition; deep learning; classification; parameter tuning

1 Introduction

Arabic remains the official language for major nations in the Middle East region. It was the original language of the Hadith and the Quran; therefore, it has become extensively learned by Muslims across the globe. There were three major methods of the Arabic language. Firstly, traditional Arabic was utilized in Islamic manuscripts like the Quran [1]. The Arabic dialect is classified into seven categories: Maltese, Egyptian, North African, Levantine, Yemeni, and Gulf, Iraqi [2]. Arabic has become part of the Semitic linguistic family. Writings in Arabic were written and read from left to right in contrast to English linguistics. The Arabic alphabet has twenty-eight letters [3]. It has just 2 vowels: (أ/alif and ي/yaa’). Every letter in Arabic is printed in many shapes based on the word’s position. For instance, the letter “ف/faa’” is printed as “ـفـ/ فـ / ف” based on the position that is placed in the middle, in the beginning, or the word end [4].

The utility of diacritics has become common in Arabic. Diacritics were named Harakāt or Tashkīl in Arabic, and it is employed as a phonetic guide. In certain cases, diacritics help reduce ambiguity, as multiple words in Arabic could share the same spelling but contains various meaning and pronunciations [5,6]. For instance, the Arabic word “شعر,” “شَعْرٌ” denotes hair, “شَعَرَ” represents feel, and “شِعْر” indicates poetry. In a few decades, numerous studies have focused on researching Arabic handwriting due to its distinct properties compared to other languages [7]. It was a cursive script which has twenty-eight basic letters in which, the majority of which changed their shape following their position across the globe (ـه, ـهـ, هـ, ه].. Handwriting analysis is done to recognize the script, detect the writer, and recognise the expressive state of the writer [8,9]. There were 2 branches of handwriting detection. Initially, offline in which, the handwriting can be attained from a digital input source. In this case, it attains a binary image in which the pixel values are either 0 or 1 [10]. Another one, termed online handwriting script, was captured by touch-sensitive gadgets. After dynamic data was obtainable, like pen pressure, (x, y) coordinates, the temporal order of trajectory, and velocity profile, with the advancement of deep learning (DL), convolutional neural networks (CNNs) have shown an important ability to recognize handwritten characters of various languages [11]: Malayalam, Latin, Devanagari, Chine, etc. Many studies enhanced the CNN structure for achieving good handwritten character recognition performances. But neural network (NN) with excellent performance generally needs a moral tune of CNN hyperparameter and the finest choice of implied optimizing techniques [12]. Additionally, many trained datasets were needed to achieve outstanding performances [13].

In [14], the authors devised and advanced a deep ensemble architecture in which ResNet-18 architecture can be exploited for modelling and classifying character images. To be specific, it implemented ResNet-18 by accumulating a dropout layer after all convolutional layers and compiled it in multiple ensembling methods to detect isolated handwritten Arabic characters mechanically. Alkhateeb [15] presented an efficient technique for devising a mechanism to recognize an isolated handwritten Arabic character depending on a DL method. The DL methods depended on the CNNs. It plays a significant part in any computer vision (CV) application. The CNN technique can be trained and developed with handwritten Arabic characters offline, utilizing 3 Arabic handwritten character recognition data sets. Saleem et al. [16] modelled an innovative, effective segmenting algorithm by modifying an ANN technique and making it suitable for the binarizing phase related to blocks. Our modified approach can be integrated into a novel actual rotation method for achieving precise segmenting via scrutinising the histogram of binary images. Additionally, project an innovative structure for correct text rotation that would help us establish a segmentation approach that could facilitate text abstraction from its background. The radon transforms and Image projections were utilized and enhanced by machine learning (ML) depending on a co-occurrence matrix for producing binary images.

Elleuch et al. [17] implied Convolutional Deep Belief Networks (CDBN) to textual image data has Arabic handwritten script (AHS) and assessed it on 2 distinct databases classified by the high or low-dimensional properties. Along with the advantages of deep networks, the mechanism can be protected against overfitting. In [18], the recognition and segmentation stages will be addressed. The text segmentation difficulties and a solution set for every challenge were presented. The CNN DL approach was utilized in the recognition stage. The use of CNN results in important enhancements over distinct ML classifier methods. It enables the automated feature extraction of images. Fourteen native CNN structures were suggested once a set of try-and-error trials.

This study develops fireworks optimization with a deep learning-based AHCR (FWODL-AHCR) technique. The major intention of the FWODL-AHCR technique is to recognize the distinct handwritten characters in the Arabic language. It initially pre-processes the handwritten images to improve their quality of them. Then, the RetinaNet-based deep convolutional neural network is applied as a feature extractor to produce feature vectors. Next, the deep echo state network (DESN) model is utilized to classify handwritten characters. Finally, the FWO algorithm is exploited as a hyperparameter tuning strategy to boost recognition performance. A series of simulations have been performed to exhibit the enhanced performance of the FWODL-AHCR technique.

2 Materials and Methods

In this study, a new FWODL-AHCR technique has been developed to recognize the distinct handwritten characters in the Arabic language. It initially pre-processes the handwritten images to improve their quality of them. Then, the RetinaNet-based deep convolutional neural network is applied as a feature extractor to produce feature vectors. Next, the FWO with DESN model is utilized to classify handwritten characters. Fig. 1 portrays the overall block diagram of the FWODL-AHCR approach.

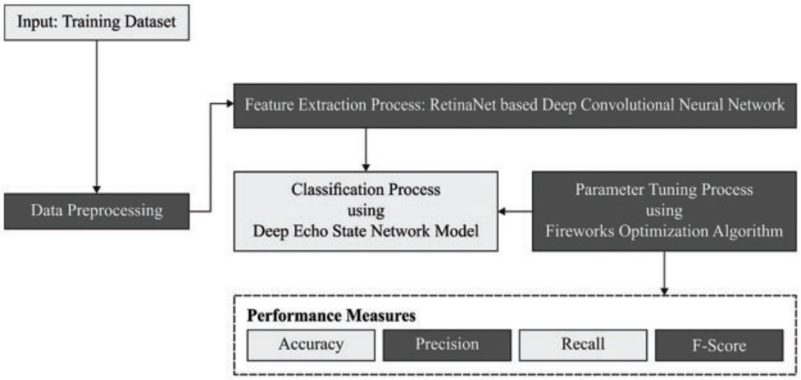

Input: Training Dataset
Feature Extraction Process: RetinaNet based Deep Convolutional Neural Network
Data Preprocessing
Classification Process using Deep Echo State Network Model
Parameter Tuning Process using Fireworks Optimization Algorithm
Performance Measures
Accuracy
Precision
Recall
F-Score

Figure 1: Overall block diagram of FWODL-AHCR approach

2.1 *Pre-Processing*

Initially, a new Wienmed filter (WIMF) is employed to remove the undesirable portions and inappropriate noise in the background of an image, thus confining the higher or lower frequency that perceives or improves the image boundary. The WIMF is the cooperative form of two filters, namely median and Wiener filters [19]. Both filters are integrated to reduce errors and noise distribution in an image efficiently. The filter aims to replace the neighbourhood and noisy image pixels, which were previously arranged based on the image intensity. Now, the m mean value of every pixel has $a\ (PQ)$ sized mask matrix and is evaluated as follows:

$$m = \frac{1}{PQ}\sum\nolimits_{p,q\in A} a\,(p,\ q) \tag{1}$$

In Eq. (1), $(p,\ q)$ denotes the pixel dimension of an image, a represents all the images, and A indicates an image. The Gaussian noise variance is shown below, σ^2 indicates the variance of Gaussian noise in an image, and n represents the noise feature. Now, the existing noise feature was traced.

$$\sigma^2 = \frac{1}{PQ}\sum\nolimits_{p,q\in A} n\,(p,\ q) - m^2 \tag{2}$$

The Wiener formula for the novel image pixel is as follows, w indicates the wiener parameter. Now, the traced noise feature has been eliminated from the data.

$$w\,(p,\ q) = \sigma^2\,[n - a\,(p,\ q)] \tag{3}$$

Thus, the WIMF eliminates undesirable noise in an image, and the attained image is moved to the classification process.

2.2 RetinaNet Feature Extraction

The RetinaNet-based deep convolutional neural network is applied as a feature extractor to produce feature vectors. The RetinaNet-based transfer learning (TL) mechanism is employed in the study to derive feature vectors [20]. An input map based on the individual layer still accomplishes the resultant map. Assume $\in R^{h'\times w'\times c'}$ (h: height, w: width, c: channel) are RGB images. Every layer gets X and the set of parameters W as input and output images $y \in R^{h'\times w'\times c'}$, for instance, $y = f\,(X,\ W)$. It makes an activation map to illustrate the reaction of that filter in every spatial region. To calculate the input X with a set of filters $W \in R^{\bar{h}\times\bar{w}\times\bar{c}\times c'}$ and $b \in R^{c'}$ bias is added in the following:

$$y_{i'j'k'} = f\left(b_{k'} + \sum_{i=1}^{\bar{h}}\sum_{j=1}^{\bar{w}}\sum_{d=1}^{c} W_{ijdk} \times X_{i'+i,j'+j,d'}\right). \tag{4}$$

Then, the max-pooling layer is applied to decrease the parameter and computation with the size of imputing shapes. It estimates the maximal response of each image channel from the $\bar{h} \times W$ sub-window that is implemented as a subsampling function:

$$y_{i'j'k'=} \max_{1<i<\tilde{h}},\ 1 < j < \tilde{w} X_{i'+ij'+j,k}. \tag{5}$$

At last, the fully connected (FC) layer is a group of layers which incorporates the data extracted from the preceding layer. This layer gets an X input, processes them, and the final FC layer produces a $1D$ vector of size. RetinaNet primarily comprises ResNet, a feature pyramid network (FPN), and two fully convolution networks (FCNs). The ResNet applies a network layer. The common type of network layers is 50, 101, and 152 layers. The 101 layers with optimum trained efficiency are selected. It eliminates the architecture of echocardiography using ResNet. An FPN is a technique to eliminate the features of every dimension from an image with a CNN structure.

Focal loss: it is an adapted form of the cross entropy (CE), and it is shown as follows:

$$CE\ \ (p,\ y) = \begin{cases} -\log\,(p) & if\ y = 1 \\ -\log\,(l-) & otherwise \end{cases}, \tag{6}$$

In Eq. (6), $y \in [-1, +1]$ denotes the ground truth type, and $p \in [0, 1]$ signifies the prediction probability to $y = 1$.

$$p_t = \begin{cases} p, & \mathit{if}\ y = l \\ l - p, & \mathit{otherwise} \end{cases} \tag{7}$$

The previous equation is rewritten by:

$$CE\ (p,\ y) = CE\,(p_t) = -\log\ (p_t) \tag{8}$$

To overcome the issue of data imbalance amongst the negative and positive instances is formulated:

$$CE\,(p_t) = -\alpha_t \log\,(p_t)\,. \tag{9}$$

Amongst them,

$$\alpha_t = \begin{cases} \alpha, & \mathit{if}\ y = 1, \\ l - \alpha & \mathit{otherwise}, \end{cases} \tag{10}$$

In Eq. (10), $\alpha \in [0, 1]$ signifies the weight factor. To overcome the problem, the concentrating parameter C was defined for obtaining the focal loss process:

$$FL\,(p_t) = -\alpha_t(1 - p_t)^{\gamma} \log\ (p_t)\,. \tag{11}$$

2.3 Handwritten Character Recognition Using DESN Model

At this stage, the DESN model is utilized to classify handwritten characters. DESN is a straight stack of reservoirs. In the t time step, the primary layer of DESN is fed through the external input (t), and every successive layer is fed through the output of the preceding one [21]. The unit size in every layer is flexible due to the architecture of DESN. In the study, the amount of units N_R in every layer is similar. $X^{(l)}\,(t) \in \mathbb{R}^{N_R}$ indicates the state of layer 1 at t time. For robustness prediction and fast convergence, a bias to reservoir weight vector is adopted for every layer of DESN, formulated as follows:

$$X^{(l)}\,(t) = \left(1 - \alpha^{(l)}\right) X^{(l)}\,(t-1) + \alpha^{(l)} \tanh\,(W_{in}^{(l)} i^{(l)}\,(t) + \theta^{(l)} + \hat{W} X^{(l)}\,(t-1)), \tag{12}$$

In Eq. (12), $W_{in}^{(l)} \in \mathbb{R}^{N_R \times N_R}$ denotes the input weight matrix of the layer, $\hat{w}^{(l)} \in \mathbb{R}^{N_R \times N_R}$ indicates the recurrent weight matrix of the l layer, and $\theta^{(l)} \in \mathbb{R}^{N_R \times N_R}$ represents the bias-to-reservoir weight vector of l layer. The superscript (l) shows the parameter and hyper-parameter of the l layer. *tanh* signifies the element-by-element application of hyperbolic tangent, and $\alpha^{(l)}$ characterizes the leaky parameter at *the l* layer. Additionally, $i^{(l)}\,(t)$ signifies the input of the l-th layer at the t time step as follows:

$$i^{(l)}\,(t) = \begin{cases} \mu^{(t)},\ \mathit{if}\ l = 1, \\ X^{(l-1)}\,(t)\,,\ \mathit{if}\ l > 1. \end{cases} \tag{13}$$

Especially, $W_{in}^{(1)} \in \mathbb{R}^{N_R \times N_R}$ characterizes the input weight matrix of the initial layer. $W_{in}^{(l)} \in \mathbb{R}^{N_R \times N_R}$ signifies the input weight matrix of consecutive layers, $l > 1$, N_L signifies the number of reservoir layers, and $y\,(t) \in \mathbb{R}^{N_Y}$ refers to the output of DESN that is evaluated as follows:

$$y\,(t) = W_{out} X\,(t) + \theta_{out}. \tag{14}$$

$W_{out} \in \mathbb{R}^{N_Y \times N_L N_R}$ embodies the output weight matrix, and $\theta_{out} \in \mathbb{R}^{N_Y}$ denotes the bias-to-readout weight vector. $X(t)$ denotes the global state of DESN as:

$$X(t) = \left(X^{(1)}(t), X^{(2)}(t), \ldots, X^{(N_L)}(t)\right) \in \mathbb{R}^{N_L N_R}. \tag{15}$$

2.4 Hyperparameter Tuning

This study exploits the FWO algorithm as a hyperparameter tuning strategy to boost recognition performance. Recently, the study presented an FWO algorithm that is a swarm intelligence (SI) optimization technique based on the explosion of fireworks (FW) [22]. Once the FW detonates, the sparks are in all places. The detonation procedure of the FW is considered the search behaviour in the local space. It is highlighted that FW of different qualities produces dissimilar sparks once they explode. Higher-quality FW produces numerous sparks once they explode. The explosion of FW forms a circle, and the spark is focused in the explosion centre.

On the other hand, the worst FW produces fewer sparks once it explodes, and the sparks expand to a peculiar form. In the context of the SI approach, an FW is considered a solution candidate. A better FW implies that a solution candidate is positioned closer to the global optimum solution in a promising region. As a result, additional sparks are produced nearby better FWs to discover the optimal global solution, and the search radius is smaller. The worst FW implies that the location of the solution candidate is not perfect. Hence, the search radius must be large, decreasing the spark produced. Fig. 2 demonstrates the flowchart of the FWO technique.

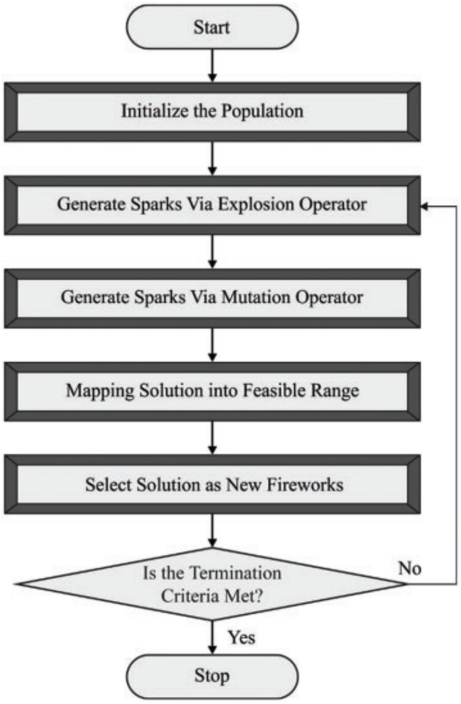

Start
Initialize the Population
Generate Sparks Via Explosion Operator
Generate Sparks Via Mutation Operator
Mapping Solution into Feasible Range
Select Solution as New Fireworks
Is the Termination Criteria Met?
No
Yes
Stop

Figure 2: Flowchart of FWO technique

From the above mentioned, better FW must generate additional sparks, whereas worst FW produces fewer sparks. The approximation of the number of sparks produced by each FW is shown in

Eq. (16). Good FWs are closer to the global optimum, so the explosion amplitude is small, whereas the worst FW is the opposite. The amplitude of explosion for every FW can be determined by:

$$S_i = \hat{S} \cdot \frac{y_{\max} - f(x_i) + \xi}{\sum_{i=1}^{n}(y_{\max} - f(x_i)) + \xi} \tag{16}$$

$$A_i = \hat{A} \cdot \frac{f(x_i) - y_{\min} + \xi}{\sum_{i=1}^{n}(f(x_i) - y_{\min}) + \xi} \tag{17}$$

From the expression, $y_{\min} = \min(f(x_i))$, $y_{max} = \max(f(x_i))$, $\widehat{S}$ and $\hat{A}$ are constants that control the explosion spark number and explosion amplitude size individually.

Notably, FW design two ways of producing sparks; one is an explosion spark for a standard search, and its approach is demonstrated in Algorithm 1. Another is the Gaussian spark, a mutation model, and the algorithm is demonstrated in Algorithm 2.

Algorithm 1: Generate "explosion sparks"

```
Initialization of the position of spark: $\widetilde{x}_j = x_i$;
    $z = round(d.\ \mathrm{rand}(0, 1))$;
Arbitrarily choice $z$ dimension of $\widetilde{x}_j$;
Estimate the displacement: $h = A_i.\ \mathrm{rand}(-1, 1)$;
for every dimension$\widetilde{x}_k^{\mathrm{j}} \in$ {pre-selected $z$ dimension of $\widetilde{x}_j$} do
   $\widetilde{x}_k^{\mathrm{j}} = \widetilde{x}_k^{\mathrm{j}} + h$;
   if $\widetilde{x}_k^{\mathrm{j}} < \chi_k^{\min}$ or $\widetilde{x}_k^{\mathrm{j}} > x_k^{max}$ then
      map $\widetilde{x}_k^{\mathrm{j}}$ to the possible space: $\widetilde{x}_k^{\mathrm{j}} = x_k^{\min} + \left|\widetilde{x}_k^{\mathrm{j}}\right| \% \left(x_k^{max} - x_k^{\min}\right)$;
  end if
end for
```

Algorithm 2: Generate "Gaussian sparks"

```
Initialization of the position of spark: $\widetilde{x}_j = x_i$;
   $z = round(d.\ \mathrm{rand}(0, 1))$;
Arbitrarily choice $z$ dimension of $\widetilde{x}_j$;
Compute the coefficient of Gaussian explosion: $= Gaussian(1,1)$;
for every dimension $\widetilde{x}_k^{\mathrm{j}} \in$ { pre-selection$z$ dimension of $\widetilde{x}_j$} do
  $\widetilde{x} = \widetilde{x} \cdot g$;
   if $\widetilde{x}_k^{\mathrm{j}} < \chi_k^{\min}$ or $\widetilde{x}_k^{\mathrm{j}} > x_k^{\max}$ then
      map $\widetilde{x}_k^{\mathrm{j}}$ to the potential space: $\widetilde{x}_k^{\mathrm{j}} = x_k^{\min} + \left|\widetilde{x}_k^{\mathrm{j}}\right| \% \left(x_k^{\max} - x_k^{\min}\right)$;
  end if
end for
```

3 Results and Discussion

The proposed model is simulated using Python 3.6.5 tool. The proposed model experiments on PC i5-8600k, GeForce 1050Ti 4 GB, 16 GB RAM, 250 GB SSD, and 1 TB HDD. The parameter settings are learning rate: 0.01, dropout: 0.5, batch size: 5, epoch count: 50, and activation: ReLU. In this section, the performance validation of the FWODL-AHCR model is tested using 28 Arabic characters, as given in Table 1.

Table 1: List of Arabic characters

Class name	Description	Class name	Description	Class name	Description
Chr-1	Alif	Chr-11	Zaay	Chr-21	Qaf
Chr-2	Baa	Chr-12	Seen	Chr-22	Kaf
Chr-3	Tea	Chr-13	Sheen	Chr-23	Lam
Chr-4	Thea	Chr-14	Sad	Chr-24	Meem
Chr-5	Jam	Chr-15	Dhad	Chr-25	Noon
Chr-6	Haa	Chr-16	Tah	Chr-26	Ha
Chr-7	Khaa	Chr-17	Dha	Chr-27	Waw
Chr-8	Daal	Chr-18	Ain	Chr-28	Yaa
Chr-9	Thaal	Chr-19	Ghen		
Chr-10	Raa	Chr-20	Fa		

Table 2 and Fig. 3 report the overall recognition outcomes of the FWODL-AHCR model on the test AHCD dataset. The results implied that the FWODL-AHCR model improved performance under all characters. For instance, on Chr-1, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.93%, 99.95%, 99.98%, and 99.89%, respectively. At the same time, on Chr-2, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.94%, 99.89%, 99.88%, and 99.91%, correspondingly; additionally, on Chr-3, the FWODL-AHCR model has presented $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.92%, 99.92%, 99.97%, and 99.90%, correspondingly, Finally, on Chr-4, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.84%, 99.93%, 99.98%, and 99.93%, correspondingly. Regarding, Chr-5, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.82%, 99.89%, 99.87%, and 99.93%, correspondingly.

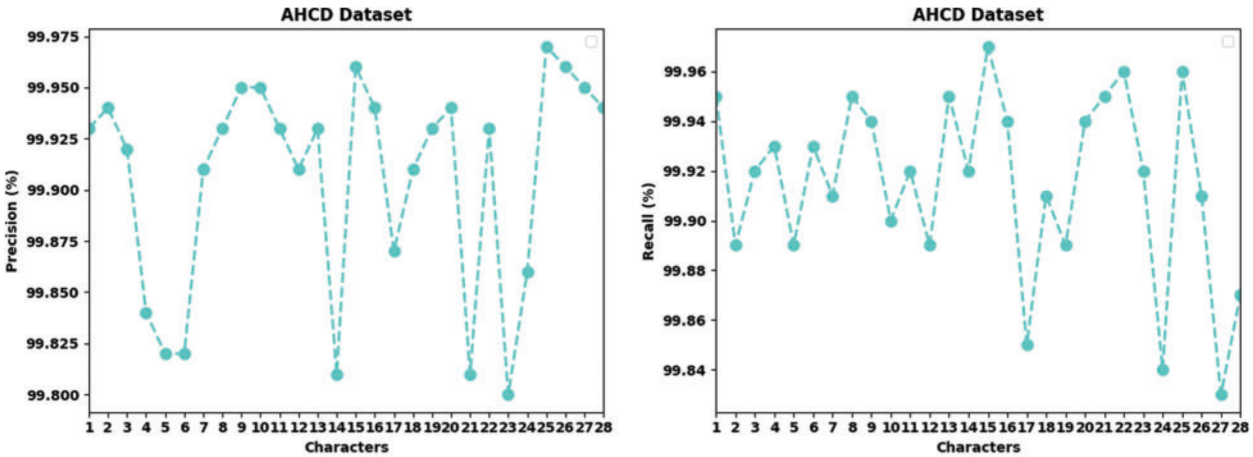
Table 2: Result analysis of FWODL-AHCR approach with distinct characters under the AHCD dataset

	AHCD dataset			
Characters	Precision	Recall	Accuracy	F1-score
Chr-1	99.93	99.95	99.98	99.87
Chr-2	99.94	99.89	99.88	99.91
Chr-3	99.92	99.92	99.97	99.90
Chr-4	99.84	99.93	99.98	99.93
Chr-5	99.82	99.89	99.87	99.93
Chr-6	99.82	99.93	99.92	99.89
Chr-7	99.91	99.91	99.92	99.90
Chr-8	99.93	99.95	99.87	99.89
Chr-9	99.95	99.94	99.90	99.87
Chr-10	99.95	99.90	99.93	99.94
Chr-11	99.93	99.92	99.90	99.96
Chr-12	99.91	99.89	99.91	99.92

(Continued)

Table 2 (continued)

AHCD dataset				
Characters	Precision	Recall	Accuracy	F1-score
Chr-13	99.93	99.95	99.98	99.92
Chr-14	99.81	99.92	99.92	99.87
Chr-15	99.96	99.97	99.88	99.88
Chr-16	99.94	99.94	99.93	99.90
Chr-17	99.87	99.85	99.93	99.90
Chr-18	99.91	99.91	99.91	99.86
Chr-19	99.93	99.89	99.94	99.91
Chr-20	99.94	99.94	99.93	99.88
Chr-21	99.81	99.95	99.95	99.92
Chr-22	99.93	99.96	99.90	99.89
Chr-23	99.80	99.92	99.97	99.94
Chr-24	99.86	99.84	99.88	99.97
Chr-25	99.97	99.96	99.96	99.95
Chr-26	99.96	99.91	99.94	99.94
Chr-27	99.95	99.83	99.95	99.95
Chr-28	99.94	99.87	99.88	99.97
Average	99.91	99.92	99.92	99.91



(a) (b)

Figure 3: (Continued)

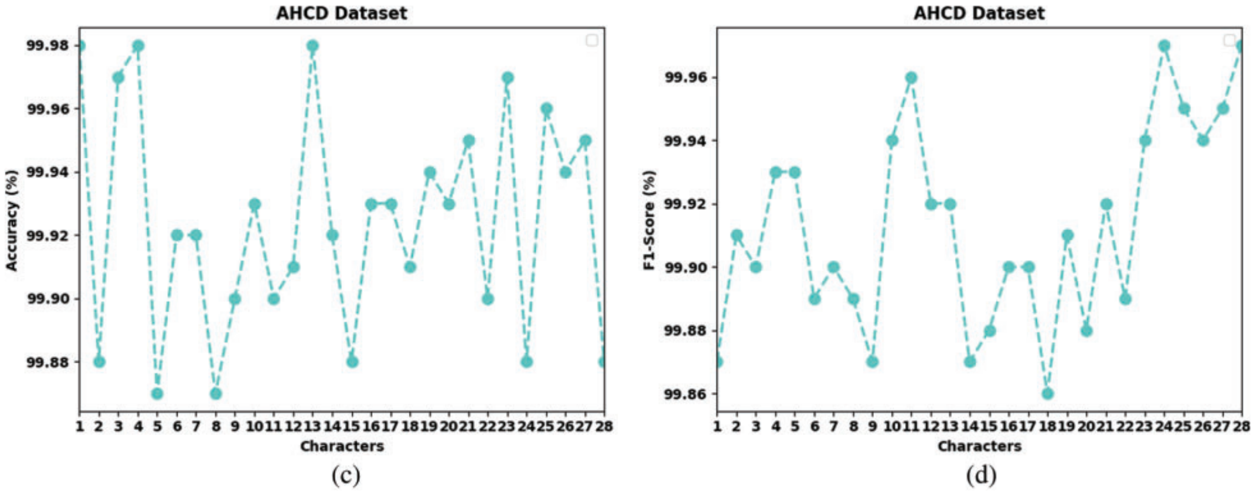


Figure 3: Result analysis of FWODL-AHCR approach under AHCD dataset (a) $Prec_n$, (b) $Reca_l$, (c) $Accu_y$, and (d) $F1_{score}$

The training accuracy (TA) and validation accuracy (VA) attained by the FWODL-AHCR method on the AHCD dataset is shown in Fig. 4. The experimental outcome implicit in the FWODL-AHCR technique has reached maximal values of TA and VA. In Particular, the VA is greater than TA.

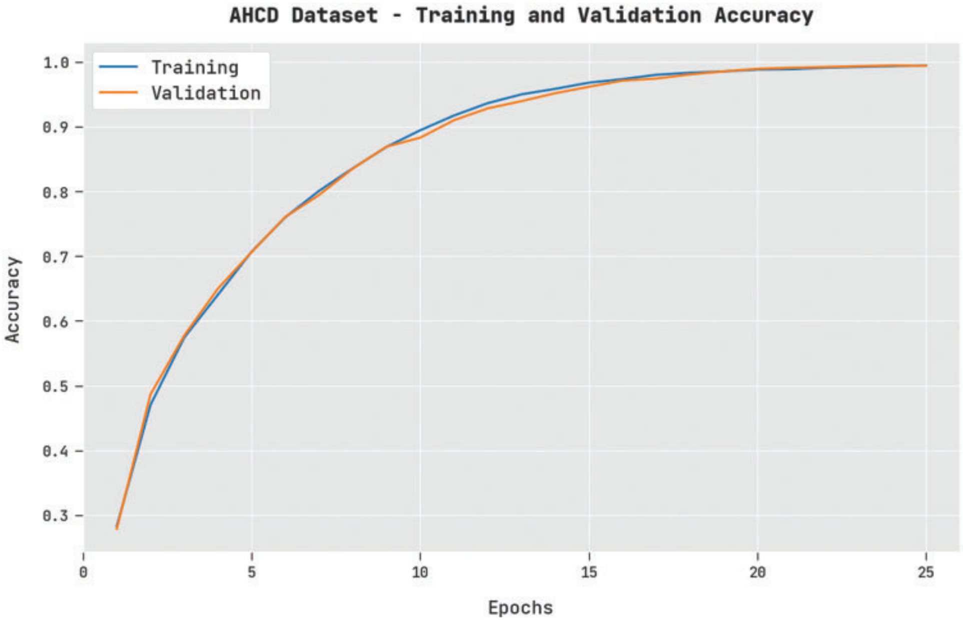


Figure 4: TA and VA analysis of FWODL-AHCR approach under the AHCD dataset

The training loss (TL) and validation loss (VL) reached by the FWODL-AHCR approach on the AHCD dataset are exhibited in Fig. 5. The experimental outcome implied that the FWODL-AHCR algorithm had accomplished the least values of TL and VL. Specifically, the VL is lesser than TL.

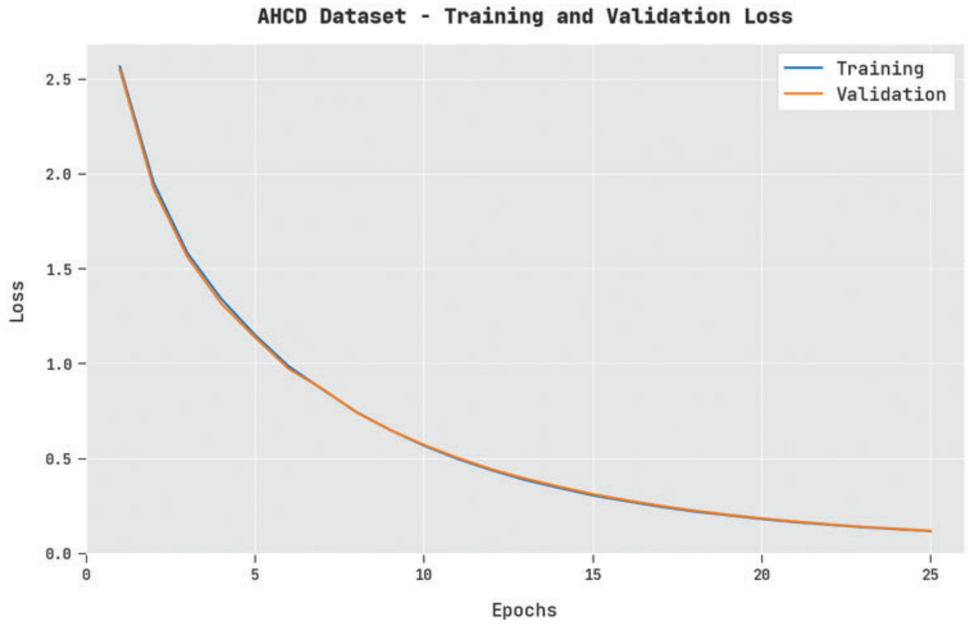


Figure 5: TL and VL analysis of FWODL-AHCR approach under the AHCD dataset

Table 3 and Fig. 6 report the overall recognition outcomes of the FWODL-AHCR model on the test Hijja dataset. The results implied that the FWODL-AHCR model improved performance under all characters. For example, on Chr-1, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.21%, 98.76%, 98.81%, and 99.39%, correspondingly. Additionally, on Chr-2, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.10%, 99.08%, 99.29%, and 98.60%, correspondingly. Also, on Chr-3, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.18%, 98.85%, 98.77%, and 99.04%, correspondingly. After, on Chr-4, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 99.09%, 98.69%, 98.80%, and 99.11%, respectively. Further, on Chr-5, the FWODL-AHCR model has offered $prec_n$, $reca_l$, $accu_y$, and $F1_{score}$ of 98.61%, 99.05%, 99.14%, and 99.12%, correspondingly.

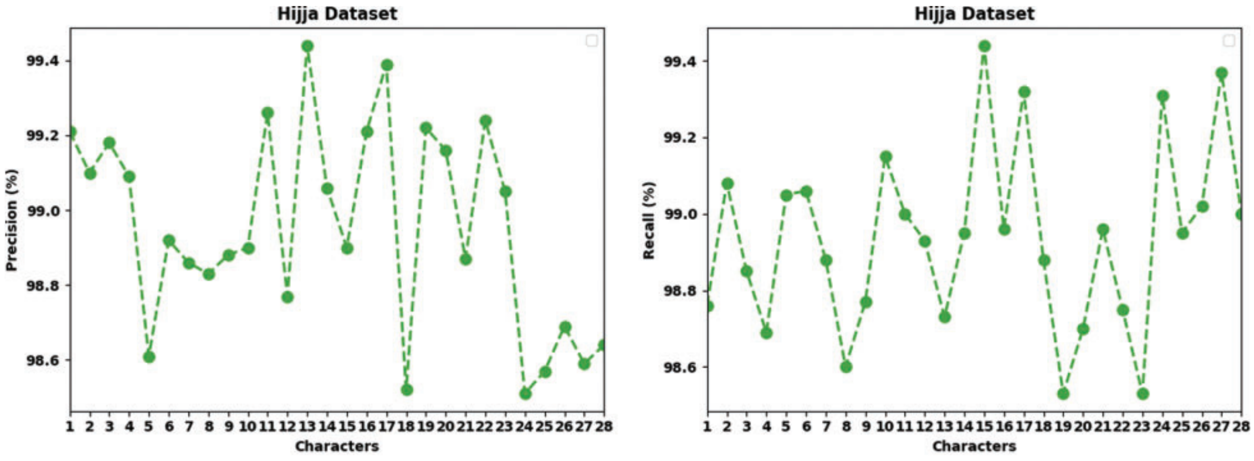
Table 3: Result analysis of FWODL-AHCR approach with distinct characters under the Hijja dataset

	Hijja dataset			
Characters	Precision	Recall	Accuracy	F1-score
Chr-1	99.21	98.76	98.81	99.39
Chr-2	99.10	99.08	99.29	98.60
Chr-3	99.18	98.85	98.77	99.04
Chr-4	99.09	98.69	98.80	99.11
Chr-5	98.61	99.05	99.14	99.12
Chr-6	98.92	99.06	98.88	99.16
Chr-7	98.86	98.88	99.12	98.71
Chr-8	98.83	98.60	99.06	99.18

(Continued)

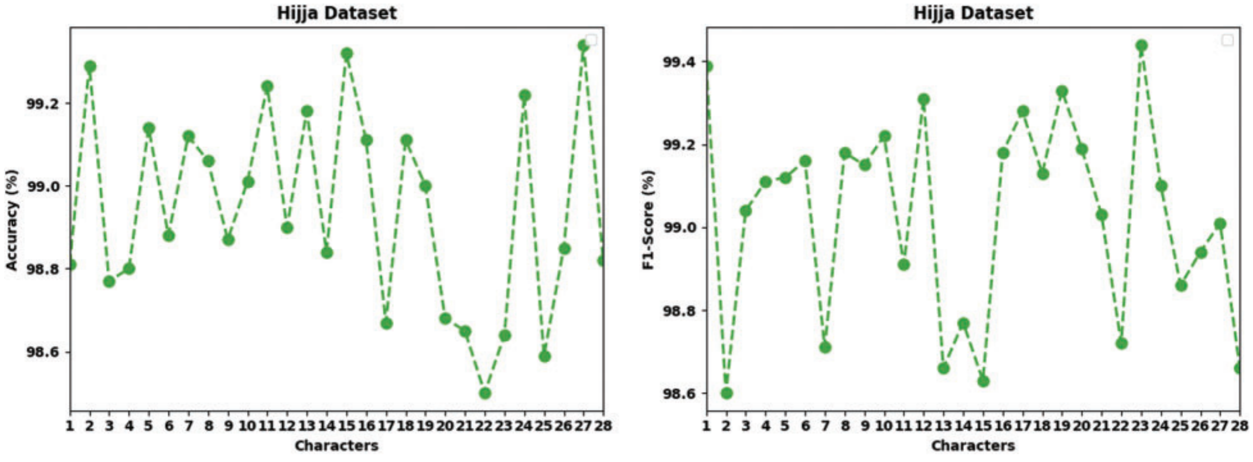
Table 3 (continued)

	Hijja dataset			
Characters	Precision	Recall	Accuracy	F1-score
Chr-9	98.88	98.77	98.87	99.15
Chr-10	98.90	99.15	99.01	99.22
Chr-11	99.26	99.00	99.24	98.91
Chr-12	98.77	98.93	98.90	99.31
Chr-13	99.44	98.73	99.18	98.66
Chr-14	99.06	98.95	98.84	98.77
Chr-15	98.90	99.44	99.32	98.63
Chr-16	99.21	98.96	99.11	99.18
Chr-17	99.39	99.32	98.67	99.28
Chr-18	98.52	98.88	99.11	99.13
Chr-19	99.22	98.53	99.00	99.33
Chr-20	99.16	98.70	98.68	99.19
Chr-21	98.87	98.96	98.65	99.03
Chr-22	99.24	98.75	98.50	98.72
Chr-23	99.05	98.53	98.64	99.44
Chr-24	98.51	99.31	99.22	99.10
Chr-25	98.57	98.95	98.59	98.86
Chr-26	98.69	99.02	98.85	98.94
Chr-27	98.59	99.37	99.34	99.01
Chr-28	98.64	99.00	98.82	98.66
Average	98.95	98.94	98.94	99.03



(a) (b)

Figure 6: (Continued)



(c) (d)

Figure 6: Result analysis of FWODL-AHCR approach under Hijja dataset (a) $Prec_n$, (b) $Reca_l$, (c) $Accu_y$, and (d) $F1_{score}$

The TA and VA obtained by the FWODL-AHCR method on the Hijja dataset are shown in Fig. 7. The experimental outcome implicit in the FWODL-AHCR technique has obtained maximal values of TA and VA. In specific, the VA is greater than TA.

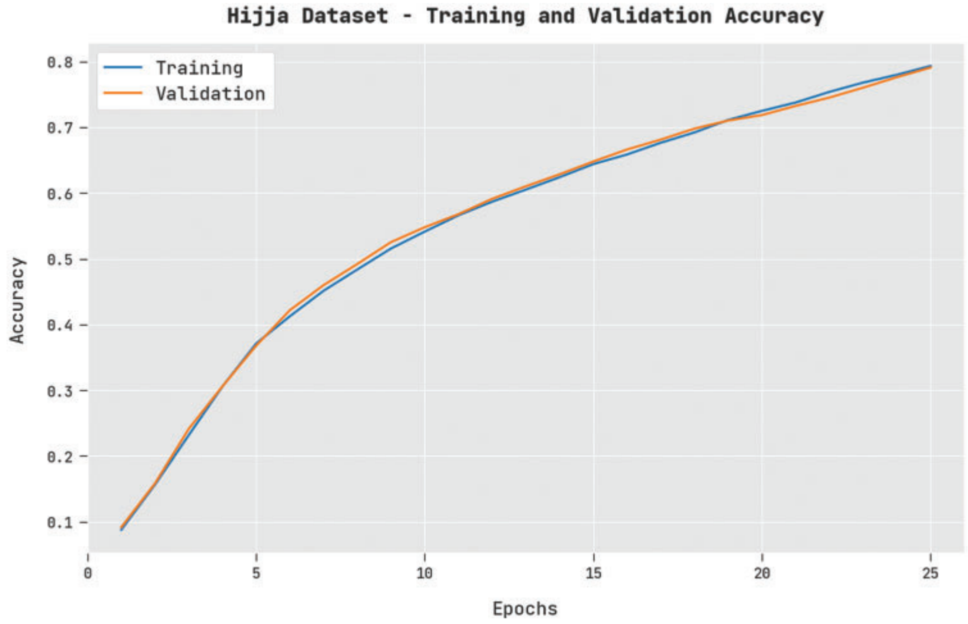


Figure 7: TA and VA analysis of FWODL-AHCR approach under the Hijja dataset

The TL and VL achieved by the FWODL-AHCR approach on the Hijja dataset are established in Fig. 8. The experimental outcome implied that the FWODL-AHCR method had accomplished the least values of TL and VL. Particularly, the VL is lesser than TL.

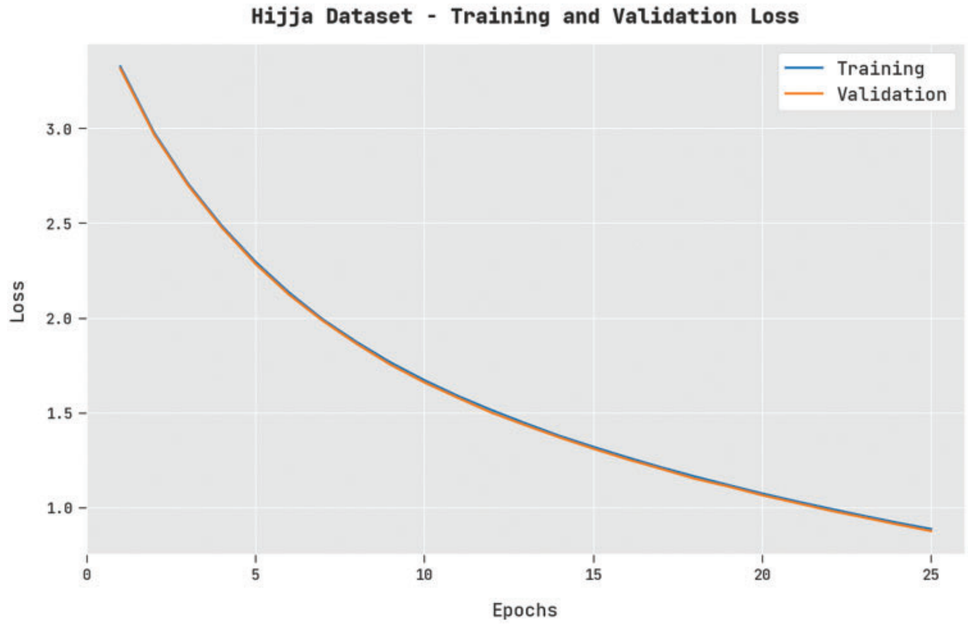


Figure 8: TL and VL analysis of FWODL-AHCR approach under the Hijja dataset

Table 4 exhibits the outcomes of the FWODL-AHCR model with other models on the AHCD dataset [23]. The experimental outcomes indicated that the FWODL-AHCR model had enhanced performance over other models. Based on $prec_n$, the FWODL-AHCR model has obtained an increased $prec_n$ of 99.91%. In contrast, the Adam, stochastic gradient descent (SGD), root mean square propagation (RMSProp), Nadam, Momentum, and Adamax models have attained reduced $prec_n$ of 96.70%, 95.69%, 96.01%, 96.40%, 95.90%, and 96.56%, respectively. Meanwhile, based on $reca_l$, the FWODL-AHCR model has reached an increased $reca_l$ of 99.92%. In contrast, the Adam, SGD, RMSProp, Nadam, Momentum, and Adamax models have attained reduced $reca_l$ of 96.40%, 95.87%, 96.24%, 96.40%, 95.37%, and 96.56%, respectively, Parallelly, based on $accu_y$, the FWODL-AHCR model has obtained increased $accu_y$ of 99.92%. In contrast, the Adam, SGD, RMSProp, Nadam, Momentum, and Adamax models have attained reduced $accu_y$ of 96.62%, 95.85%, 96.28%, 96.71%, 95.78%, and 96.35%, correspondingly. Also, based on the $F1_{score}$, the FWODL-AHCR model has gained an increased $F1_{score}$ of 99.91%, whereas the Adam, SGD, RMSProp, Nadam, Momentum, and Adamax models have attained a reduced $F1_{score}$ of 96.32%, 95.83%, 96.13%, 96.70%, 95.23%, and 96.29%, correspondingly.

Table 4: Comparative analysis of FWODL-AHCR approach with existing methods under the AHCD dataset

		AHCD dataset		
Methods	Precision	Recall	Accuracy	F1-score
FWODL-AHCR	99.91	99.92	99.92	99.91
Adam algorithm	96.70	96.40	96.62	96.32
SGD algorithm	95.69	95.87	95.85	95.83

(Continued)

Table 4 (continued)

AHCD dataset				
Methods	Precision	Recall	Accuracy	F1-score
RMSprop algorithm	96.01	96.24	96.28	96.13
Nadam algorithm	96.40	96.40	96.71	96.70
Momentum algorithm	95.90	95.37	95.78	95.23
AdaMax algorithm	96.56	96.56	96.35	96.29

Table 5 exhibits the outcomes of the FWODL-AHCR model with other models on the Hijja dataset. The experimental outcomes indicated that the FWODL-AHCR model had enhanced performance over other models. Based on $prec_n$, the FWODL-AHCR model has obtained an increased $prec_n$ of 98.95%, whereas the Adam, SGD, RMSProp, Nadam, Momentum, and Adamax models have attained reduced $prec_n$ of 88.81%, 87.94%, 87.78%, 88.91%, 88.01%, and 87.63%, correspondingly. In addition, based on *the* $reca_l$, the FWODL-AHCR model has obtained an increased $reca_l$ of 98.94%. In contrast, the Adam, SGD, RMSProp, Nadam, Momentum, and Adamax models have reached reduced $reca_l$ of 87.94%, 87.83%, 87.82%, 88.06%, 87.89%, and 87.66%, correspondingly. In the end, based on $accu_y$, the FWODL-AHCR model has obtained an increased $accu_y$ of 98.94%. In contrast, the Adam, SGD, RMSProp, Nadam, Momentum, and Adamax models have attained reduced $accu_y$ of 88.59%, 87.60%, 88.25%, 88.03%, 87.89%, and 88.16%, correspondingly. Finally, based on the $F1_{score}$, the FWODL-AHCR model has obtained an increased $F1_{score}$ of 99.03%, whereas the Adam, SGD, RMSProp, Nadam, Momentum, and Adamax models have attained a reduced $F1_{score}$ of 88.81%, 87.91%, 87.73%, 88.19%, 87.89%, and 87.98%, correspondingly.

Table 5: Comparative analysis of FWODL-AHCR approach with existing methods under the Hijja dataset

Hijja dataset				
Methods	Precision	Recall	Accuracy	F1-score
FWODL-AHCR	98.95	98.94	98.94	99.03
Adam algorithm	88.81	87.94	88.59	88.81
SGD algorithm	87.94	87.83	87.60	87.91
RMSprop algorithm	87.78	87.82	88.25	87.73
Nadam algorithm	88.91	88.06	88.03	88.19
Momentum algorithm	88.01	87.89	87.49	87.89
AdaMax algorithm	87.63	87.66	88.16	87.98

4 Conclusion

In this study, a new FWODL-AHCR technique has been developed to recognize the distinct handwritten characters in the Arabic language. It initially pre-processes the handwritten images to improve their quality of them. Then, the RetinaNet-based deep convolutional neural network is applied as a feature extractor to produce feature vectors. Next, the DESN model is utilized for the classification of handwritten characters. Finally, the FWO algorithm is exploited as a hyperparameter tuning strategy to boost recognition performance. A series of simulations have been performed to exhibit the enhanced performance of the FWODL-AHCR technique, and the comparison study portrayed the supremacy of the FWODL-AHCR technique over other approaches. In the future, the FWODL-AHCR technique's performance can be boosted using the fusion models.

Acknowledgement: None.

Funding Statement: Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2022R263), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: 22UQU4340237DSR39.

Author Contributions: The manuscript was written through contributions of all authors. All authors have given approval to the final version of the manuscript. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data sharing not applicable to this article as no datasets were generated during the current study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] H. M. Balaha, H. A. Ali, M. Saraya, and M. Badawy, "A new Arabic handwritten character recognition deep learning system (AHCR-DLS)," *Neural Comput. Appl.*, vol. 33, no. 11, pp. 6325–6367, 2021. doi: 10.1007/s00521-020-05397-2.

[2] R. Ahmed *et al.*, "Novel deep convolutional neural network-based contextual recognition of Arabic handwritten scripts," *Entropy*, vol. 23, no. 3, 2021, Art. no. 340. doi: 10.3390/e23030340.

[3] Y. S. Can and M. E. Kabadayı, "Automatic cnn-based Arabic numeral spotting and handwritten digit recognition by using deep transfer learning in ottoman population registers," *Appl. Sci.*, vol. 10, no. 16, 2020, Art no. 5430. doi: 10.3390/app10165430.

[4] A. A. A. Ali, M. Suresha, and H. A. M. Ahmed, "A survey onArabichandwrittencharacterrecognition, SN Computer Science," *SN Comput. Sci.*, vol. 1, no. 3, p. 152, 2020. doi: 10.1007/s42979-020-00168-1.

[5] F. N. Al-Wesabi, "Entropy-based watermarking approach for sensitive tamper detection of Arabic text," *Comput. Mater. Contin.*, vol. 67, no. 3, pp. 3635–3648, 2021. doi: 10.32604/cmc.2021.015865.

[6] B. H. Nayef, S. N. H. S. Abdullah, R. Sulaiman, and Z. A. A. Alyasseri, "Optimized leaky ReLU for handwritten Arabic character recognition using convolution neural networks," *Multimed. Tools Appl.*, vol. 81, no. 2, pp. 2065–2094, 2022. doi: 10.1007/s11042-021-11593-6.

[7] F. N. Al-Wesabi, "A hybrid intelligent approach for content authentication and tampering detection of Arabic text transmitted viainternet," *Comput. Mater. Contin.*, vol. 66, no. 1, pp. 195–211, 2021. doi: 10.32604/cmc.2020.012088.

[8] M. Elkhayati and Y. Elkettani, "UnCNN: A new directed CNN model for isolated Arabic handwritten characters recognition," *Arab. J. Sci. Eng.*, vol. 47, no. 8, pp. 10667–10688, 2022. doi: 10.1007/s13369-022-06652-5.

[9] M. Haddad, D. Gaceb, and S. Faouci, "Offline Arabic handwritten character recognition: From conventional machine learning system to deep learning approaches," *Int. J. Comput. Sci. Eng.*, vol. 25, no. 4, pp. 385–398, 2022. doi: 10.1504/IJCSE.2022.124562.

[10] S. Benbakreti, M. Benouis, A. Roumaneand, and S. Benbakreti, "Stacked autoencoderforArabichandwriting word recognition," *Int. J. Comput. Sci. Eng.*, vol. 24, no. 6, pp. 629–638, 2021. doi: 10.1504/IJCSE.2021.119988.

[11] A. Khémiri, A. K. Echi, and M. Elloumi, "Bayesianversusconvolutional networksforArabichandwriting recognition," *Arab. J. Sci. Eng.*, vol. 44, no. 11, pp. 9301–9319, 2019. doi: 10.1007/s13369-019-03939-y.

[12] M. Salam and A. A. Hassan, "OfflineisolatedArabichandwriting characterrecognitionsystembased on SVM," *Int. Arab J. Inf. Technol.*, vol. 16, no. 3, pp. 467–472, 2019.

[13] N. Altwaijry and I. Al-Turaiki, "Arabic handwriting recognition system using convolutional neural network," *Neural Comput. Appl.*, vol. 33, no. 7, pp. 2249–2261, 2021. doi: 10.1007/s00521-020-05070-8.

[14] H. Alyahya, M. M. B. Ismail, and A. Al-Salman, "Deep ensembleneural networksforrecognizingisolated Arabic handwritten characters," *ACCENTS Trans. Image Process. Comput. Vis.*, vol. 6, no. 21, pp. 68–79, 2020. doi: 10.19101/TIPCV.2020.618051.

[15] J. H. Alkhateeb, "Aneffectivedeeplearning approachforimproving off-lineArabichandwrittencharacter recognition," *Int. J. Softw. Eng. Comput. Syst.*, vol. 6, pp. 53–61, 2020. doi: 10.15282/ijsecs.6.2.2020.7.0076.

[16] S. I. Saleem, A. M. Abdulazeez, and Z. Orman, "A new segmentation framework for Arabic handwritten text using machine learning techniques," *Comput. Mater. Contin.*, vol. 68, no. 2, pp. 2727–2754, 2021. doi: 10.32604/cmc.2021.016447.

[17] M. Elleuch and M. Kherallah, "Boosting of deep convolutional architectures for Arabic handwriting recognition," *Int. J. Multimed. Data Eng. Manag.*, vol. 10, no. 4, pp. 26–45, 2019. doi: 10.4018/IJMDEM.

[18] H. M. Balaha *et al.*, "RecognizingArabichandwritten characters usingdeeplearning andgenetic algorithms,Multimedia Tools and Applications," *Multimed. Tools Appl.*, vol. 80, no. 21– 23, pp. 32473–32509, 2021. doi: 10.1007/s11042-021-11185-4.

[19] K. M. Chang and S. H. Liu, "GaussiannoisefilteringfromECGby wienerfilterand ensembleempirical mode decomposition," *J. Signal Process. Syst.*, vol. 64, no. 2, pp. 249–264, 2011. doi: 10.1007/s11265-009-0447-z.

[20] Y. Wang, C. Wang, H. Zhang, Y. Dong, and S. Wei, "Automatic ship detection based on retinanet using multi-resolution Gaofen-3 imagery," *Remote Sens.*, vol. 11, no. 5, 2019, Art. no. 531. doi: 10.3390/rs11050531.

[21] J. Long, S. Zhang, and C. Li, "Evolving deep echo state networks for intelligent fault diagnosis," *IEEE Trans. Ind. Inform.*, vol. 16, no. 7, pp. 4928–4937, 2020. doi: 10.1109/TII.2019.2938884.

[22] Y. J. Zheng, Q. Song, and S. Y. Chen, "Multiobjective fireworks optimization for variable-rate fertilization in oil crop production," *Appl. Soft Comput.*, vol. 13, no. 11, pp. 4253–4263, 2013. doi: 10.1016/j.asoc.2013.07.004.

[23] N. Wagaa, H. Kallel, and N. Mellouli, "Improved Arabic alphabet characters classification using convolutional neural networks (CNN)," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–16, 2022. doi: 10.1155/2022/9965426.