IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 April 2023, accepted 20 April 2023, date of publication 27 April 2023, date of current version 8 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3270895

RESEARCH ARTICLE

End-To-End Deep-Learning-Based Tamil Handwritten Document Recognition and Classification Model

C. VINOTHENI AND S. LAKSHMANA PANDIAN
Department of Computer Science and Engineering, Puducherry Technological University, Puducherry 605014, India
Corresponding author: C. Vinotheni (vinotheni95@pec.edu)

ABSTRACT Overview: Handwriting recognition (HR) involves converting handwritten text into machine-readable text. Tamil handwritten document recognition remains a challenging process in various real-world applications owing to the differences in the sizes, styles and orientation angles of Tamil alphabets. Prior studies concentrated only on character-level segmentation, and each character was subsequently classified. The recently developed machine learning (ML) and deep learning (DL) approaches can be utilized for Tamil handwritten character recognition (HCR). Objective: This paper attempts to present an end-to-end DL-based Tamil handwritten document recognition (ETEDL-THDR) model. Methods: Segmentation is used, first at the word level and then at the line level. ETEDL-THDR text recognition can be accomplished using two modules: line segmentation and line recognition. Initially, the ETEDL-THDR model targets improving input image quality using the median filtering (MF) technique. To create meaningful regions, more line and character segmentation activities are performed. A deep convolutional neural network (DCNN) based MobileNet approach is also applied to derive feature vectors. Finally, the water strider optimization (WSO) algorithm with a bidirectional gated recurrent unit (BiGRU) model is used to identify the Tamil characters. Results: Extensive experimental analyses of the ETEDL-THDR model have been carried out, and the results show that the ETEDL-THDR model performs better than more recent methodologies, with a maximum accuracy of 98.48%, a precision of 98.38%, a sensitivity of 97.98%, specificity of 98.27% and F-measure of 98.35%. Conclusion: The comparison results show that the proposed model can recognize Tamil handwritten documents in real time.

INDEX TERMS Handwritten character recognition, Tamil language, segmentation, deep learning, machine learning.

I. INTRODUCTION

Tamil is an ancient Dravidian language that has existed for thousands of years, with hundreds of millions of speakers, especially in South India and Sri Lanka. There are 156 characters in this language, including 23 consonants and 12 vowels, written in a non-Latin system [1]. Tamil isolated character recognition (TCR) is more challenging than Latin character recognition, owing to the large category set and possible confusion because of the similarity between handwritten characters. Former methodologies in categorizing Tamil characters have utilized template-matching and different manually created feature characteristics [2], [3]. Handwritten character recognition (HCR) comprises offline as well as online methods. Offline techniques work by considering characters as scanned pictures, whereas online approaches turn the motions (strokes) of digital pens into a set of coordinates [4]. The natural variations seen among different individuals' writing styles are an aspect that challenges the accuracy of HCR. Classical machine learning (ML) techniques have been utilized for offline HCR. A conventional ML method of HCR would include the classification, preprocessing, segmentation and feature extraction of the captured characters [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Wai-Keung Fung.

After training adequately on a sequence of characters with scanned images, an offline HCR system must be able to correctly recognize the characters when given new character images as input [6]. HCR has demonstrated its effectiveness in different applications, such as sorting cheques in banks and mail in post offices, converting handwritten forms/documents and digitizing legacy documents [7]. Digitization of a handwritten document includes noise removal, extraction of foreground texture from background text, separating individual lines, segmentation of characters from all the words, identification of isolated characters, segmentation of words in each line and transformation of grayscale/colour images to binary images [8]. This process involves isolated offline HR and TCR using the deep learning (DL) technique.

Like other languages, the Tamil language also presents complex challenges highly attributable to shape variations, orientation angles, structural style complexity, number of strokes and holes, direction variations, location mismatches, shape similarities, writing style differences and character curves and sliding [9]. It is difficult to distinguish the minute differences in a more extensive writing class because numerous research projects on the offline Tamil HR approach have been carried out in some writing classrooms. DL tends to automatically learn features from the dataset, whereas traditional methodologies take manually created feature sets [10]. Being able to achieve 100% accuracy in character recognition is encouraging since there are wide variances among the handwriting of different people and possibly even in the handwriting of one writer on different occasions.

This paper presents an end-to-end DL-based Tamil handwritten document recognition (ETEDL-THDR) model. The ETEDL-THDR model first employs the median filtering (MF) technique for noise removal. Operations for line and character segmentation are also performed to create meaningful regions. A convolutional neural network (CNN) based MobileNet approach is then applied to derive feature vectors. Finally, the water strider optimization (WSO) algorithm with a bidirectional gated recurrent unit (BiGRU) model is applied to recognize the Tamil characters. Extensive experimental analyses have been conducted to verify the ETEDL-THDR model's improved performance.

The rest of this paper is organized as follows. Section II lists related recent works and a summary of other existing models. Next, Section III introduces the proposed ETEDL-THDR model. The experimental data involved in the proposed model is provided in Section IV. The research results and a comparative study of the proposed with existing methodologies are presented in Section V. The paper is concluded in Section VI.

II. RELATED WORKS

In [11], the researchers utilized existing CNNs to identify Tamil handwritten characters (THCs) offline. The effectiveness of CNNs varies among conventional handwritten TCR (HTCR) methods in automatically deriving the features. The authors in [11] used a complete dataset of THCs created by Hewlett Packard Labs (HPL) India via HP Tablet PC, developing a CNN algorithm from the ground up by training it with Tamil characters in an offline environment and attaining superior identification results. Vinotheni et al. [12] provide a modified CNN (M-CNN) structure for quicker convergence rates and getting the highest detection precision. The M-CNN over distinct prospects, including layers loss function, design, optimization, and activation function, was deliberated.

The researchers in [13] propose a novel Tamil HCR method by following two major recognized processes and image preprocessing. The preprocessing stage encompasses morphological operations, grayscale translation from red, blue, and green (RGB), thresholding of linearisation, binarisation and image complementation. The preprocessed image is then linearised, and recognition is attempted using an optimally designed CNN. In particular, a unique Self-Adaptive Lion Algorithm (SALA), a conceptual improvement over the conventional Lion Algorithm, was used to fine-tune the weights and the fully connected layer. In [14], the researchers define an offline character recognition technique with the help of Residual Network (ResNet) architecture, describing ResNet layers and the challenges of obtaining digital characters from the ResNet layers with the help of DL. It includes data set preprocessing and the training of every Tamil character.

Kowsalya and Periasamy [15] suggest a method that uses efficient TCR. The suggested technique comprises four main proceedings, namely the feature extraction process, preprocessing process, recognition process and segmentation process. In preprocessing, the input image is given to the binarisation process, skew detection technique and Gaussian filter. After the segmentation process, character and line segmentation are made. Once the feature extraction is over, the Tamil characters are identified using the optimal artificial neural network (ANN). In this method, CNNs are altered with the help of optimization techniques. In Neural Networks (NNs), the weights are optimized using Elephant Herding Optimisation.

Ulaganathan et al. [16] recommend a new architecture of CNN for handwritten TCR. The method of utilizing CNN for the detection of handwritten characters varies from other classical methods in feature extraction. Shanmugam and Vanathi [17] suggest an innovative method for improving offline Tamil HCR by using four main levels: classification, preprocessing, feature extraction and segmentation. The Tsallis entropy approach-based atom search (TEAS) optimized strategy is used in this work to segment the Tamil characters as best as possible. The extraction and classification of input images were performed using the deep convolution extreme learning (DELM) approach based on the Newton algorithm.

The authors in [18] develop two zone-wise feature extraction methods, namely zone-wise structural and directional (ZSD) and zone-wise slopes of dominant points (ZSDP), for the recognition of the online handwritten script and word in four major Indic scripts - Devanagari, Bengali, Telugu and Tamil. These features have been utilized individually

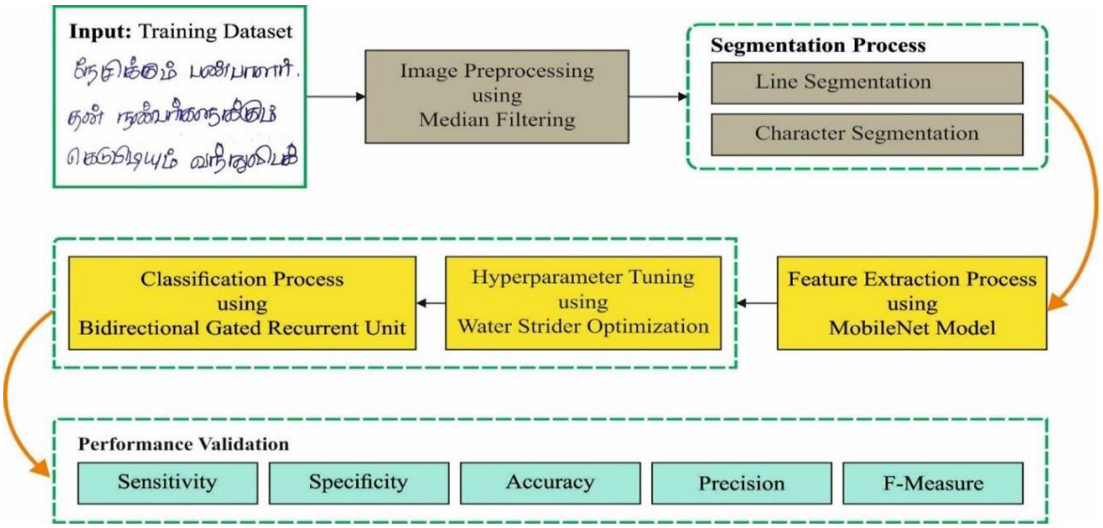

FIGURE 1. Block diagram of the ETEDL-THDR technique.

and combined with an HMM-based platform for recognition purposes.

The authors in [19] present a new method for online handwritten cursive and non-cursive word recognition in Devanagari and Bengali scripts using DL models. Several structural and directional features are then derived from every fundamental stroke of the word separately for each zone. The DL model then examines these zone-wise basic stroke features.

III. THE PROPOSED MODEL

THC identification has been improved in this work by using the new ETEDL-THDR model. To accomplish this, the ETEDL-THDR approach includes a series of procedures such as MF (for preprocessing), segmentation, MobileNet (for feature extraction), BiGRU recognition and WSO hyperparameter optimization. In the initial stage, the MF technique is used to eradicate the presence of noise. Next, the segmentation of characters takes place, and the MobileNet model is used to derive feature vectors. Then, the WSO–BiGRU approach is used for the recognition of THCs. The ETEDL-THDR technique is shown in a block diagram format in Figure 1.

A. PREPROCESSING

The MF function calculates the median of every pixel in the kernel window, and the central pixel is interchanged with this median value. This method can be highly effective in extracting salt-and-pepper noise [21]. Noticeably, during the Gaussian and box filters, the filter values to the central element are values that could not occur from the original images. However, this is different in the MF approach, where the central element is continuously exchanged with any pixel value from the images. This helps decrease the noise efficiently. The kernel size is a positive odd integer.

The MF computation is shown in Eq. (1).

$$MF\,(X) = \begin{cases} X\left[\frac{n}{2}\right] \\ \frac{\left(\frac{X[n-1]}{2} + \frac{X[n+1]}{2}\right)}{2} \end{cases} \quad (1)$$

where X refers to the orderly list of values from the dataset, and n signifies the number of values from the dataset.

B. SEGMENTATION PROCESS

Segmentation is a crucial step in a recognition system that eliminates relevant portions to allow further examination. It has been widely used in images for object verification and boundaries, i.e., lines, curves, etc. The scanned images are separated into paragraphs using the spatial space recognition method, lines using a horizontal histogram, and paragraphs into lines using a vertical histogram. The accuracy efficiency of character identification is heavily dependent on segmentation performances. The procedure of segmentation primarily comprises the following steps:

- Identifying a page's text lines.
- Recognizing individual characters in all the words.

A thorough description of the segmentation procedure is provided in the following text.

1) LINE SEGMENTATION

The projection profile procedure is a fairly common method for line segmenting grayscale images [20]. The gaps between text lines are recognized by summarising the projection profile along the horizontal direction on documents and identifying the projected values. It is then necessary to utilize a horizontal projection profile study since the text is often aligned along the horizontal line in documented image alignments as the algorithm counts the black pixels to create a raster image of all the columns for computing the horizontal projection histogram. The horizontal projection profiles are only used on M × N images, i.e., with M rows and N columns, after which a column vector of size M × 1 is produced. The total of all the row's pixel values from the document images makes up the column vectors component. The output is then subjected to character segmentation after line segmentation.

2) CHARACTER SEGMENTATION

Characters are eventually segmented from the retrieved lines. To identify the boundaries between the characters, a threshold value is applied to the length of the space between the characters [13], [15]. After determining the positions of the spaces between characters, the parts of the line segment get eliminated. The line and characters are segmented from the preprocessed document based on the above process. After the segmentation process, the resultant output is fed to the feature extraction.

C. FEATURE EXTRACTION USING MobileNet MODEL

MobileNet is a CNN-related approach that is broadly utilized for classifying images. An important benefit of utilizing the MobileNet structure is that it requires comparatively lesser computational power than the typical CNN technique, making it deployable on mobile devices and low-end computers [22]. The MobileNet architecture is a streamlined design that effectively combines a convolution layer based upon two global hyperparameters switching between parameter accuracy and latency. It uses depth-wise separable convolutions for reducing network sizes. The fundamental architecture depends on abstraction layers, a component of various convolutional that performs the quantized configuration to evaluate a common issue's difficulty in depth. To reduce the dimensionality of the input images and all internal representations of the layers with identical variables, the resolution multiplier variable ω was included.

The filter has a size of $F_s \times F_s$, and the feature vector map measures $F_m \times F_m$. The variable c_e shows the total computational effort to the basic abstract levels of structure and evaluates as shown in Eq. (2):

$$c_e = F_s \cdot F_s \cdot \omega \cdot \alpha F_m \cdot \alpha F_m + \omega \cdot \rho \cdot \alpha F_m \cdot \alpha F_m \quad (2)$$

where ω and ρ represent the number of input and output channels, respectively, in order to conduct an experimental study for handwritten document recognition, it was hypothesized that the multiplier value ω has context-specific and would range from 1 to n. The value of the variable resolution multiplier recognized by α assumed as 1. The computing effort is calculated using the variable *cost* measured in Eq. (3) as

$$cost_e = F_s \cdot F_s \cdot \omega \cdot \rho \cdot F_m \cdot F_m \quad (3)$$

The depletion variable recognized by variable d, which is computed in Eq. (4), defines the depth-wise and point-wise convolutional methods to that of the projected approach joins.

$$d = \frac{F_s \cdot F_s \cdot \omega \cdot \alpha F_m \cdot \alpha F_m + \omega . \cdot \rho \cdot \alpha F_m \cdot \alpha F_m}{F_s \cdot F_s \cdot \omega \cdot \rho \cdot F_m F_m} \quad (4)$$

Depending on the circumstance, the two hyper features' width and resolution multipliers assist in altering the accuracy and predictability of its appropriate size window. In the presented method, the input size of images is $224 \times 224 \times 3$. The primary values (224×224) refer to the height and width of images. This value is continuously superior to 32.

D. DOCUMENT RECOGNITION USING BiGRU MODEL

To recognize the THCs, the BiGRU model is utilized. Recurrent neural network (RNN) distinguishes itself from other NN primarily by permitting data to survive through a circular infrastructure [23]. The network recalls the features of the preprocessed data, ensures that NN has a memory, and connects the prior data to the current task. Thus, it can be utilized highly well in time-series data analyses. However, the traditional RNN entails the risk of gradient explosion and gradient disappearance. It cannot control long-time dependent data. To resolve this issue, Long short-term memory (LSTM) is presented. The novelty of LSTM is that it establishes a model of cells. To elaborate, the LSTM structure consists of three gates from the cells, namely the output, input, and forget gates. In order to handle data, LSTM has developed a data pass that alternates between the memorization of some data and forgetting of other data with these three gates. The Gated Recurrent Unit (GRU) was developed to improve the LSTM; it incorporates input and ''forgets gates into a single update gate. Besides, the cell, as well as hidden states, were varied, and so on. The last method was easier than the typical LSTM method, and the parameters were lesser than 33%; however, the result is close. The previous output of the hidden layer and the current input of cells initially served as the reset and update gates, respectively.

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right) \quad (5)$$

$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right) \quad (6)$$

where r_t implies the reset gate, z_t stands for the update gate, h_{t-1} denotes the preceding result, the hidden layer, x_t refers to the present input of cells, and σ signifies the sigmoid function.

Second, after receiving the gate signal, the reset gate is employed for processing data, and the memory of data is recognized under this step. The tanh function is used for immediate scaling the data to zero and one for their later use. The computation equation is as follows:

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right) \quad (7)$$

Eventually, a similar gate was utilized for data forgetting and selected memory. Besides, $h_{t-1} * (1 - z_t)$ remembers the choice while forgetting the initial hidden state. $\tilde{h}_t * z_t$ represents the selected memory of the current cell. This is demonstrated in Eq. (8):

$$h_t = h_{t-1} * (1 - z_t) + \tilde{h}_t * z_t \quad (8)$$

where z_t goes to zero and one and signifies the weight of forgetting. Figure 2 displays the infrastructure of GRU.

Similar to LSTM, GRU can handle long-time-series data. However, its structural unit is smaller than LSTM, and thus the convergence speed of GRUs is quicker. While GRU only attains the forward context data and ignores the backward context data, it can utilize the bidirectional GRU (BiGRU) technique. The BiGRU layer adopts two kinds of forwarding GRUs and reverses GRU infrastructures to the fundamental

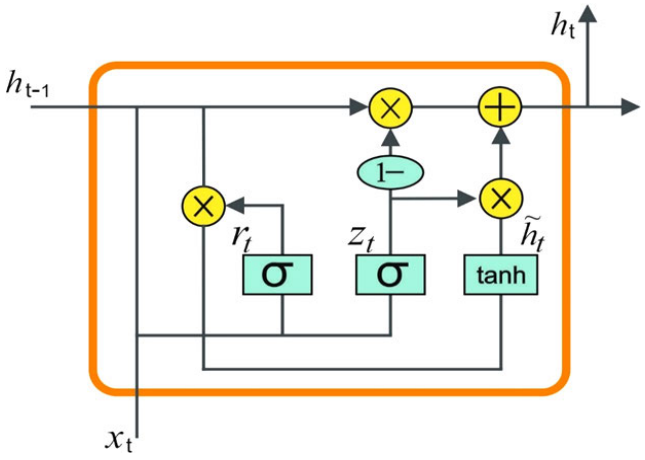

FIGURE 2. Framework of GRU.

GRU framework. BiGRU is a sequence processing model that consists of two GRUs. One GRU takes the input in a forward direction and the other in a backwards direction. Hence, the BiGRU technique accurately extracts text data features and is superior in terms of accuracy in computational outcomes.

E. HYPERPARAMETER TUNING

To effectually modify the hyperparameters related to the BiGRU model, the WSO algorithm is utilized. The WSO algorithm, whose formulation was inspired by the characteristics of insects known as *water striders*(WS), simulates their traits such as ripple communication, territorial life, foraging behaviour, mating process, succession and death. Kaveh and Eslamlou [24] thoroughly utilized WSO to find the technique that manifests the best trade-off between exploration (exploring the novel area of the searching space) and exploitation (intensifying the search in the potential region of the searching space). The six major phases of the WSO algorithm are listed as follows.

- Initial birth
- Territory establishment
- Mating
- Feeding
- Death and succession
- Termination of WSA

During birth, the WSs (solution) are initialized randomly in the lake (searching space) as follows:

$$WS_i^0 = Lb + rand_i.\,(Ub - Lb)\,, i = 1, 2, \ldots, nws, \quad (9)$$

In Eq. (8), WS_i^0 refers to the primary location of ith WSs; Lb and Ub indicate the lower and upper bounds corresponding to the variable's maximal and minimal values, respectively; $rand_i$ indicates a uniformly distributed arbitrary number that lies between zero and one produced for the WS_j; and nws denotes the amount of WS_s.

On establishing territory, the solution is separated into a predetermined number of territories (nt) based on rank. First, the WS is arranged according to the cost value and is situated in $\frac{nws}{nt}$ groups. Finally, they are chosen according to their rank in the groups, establishing territory. All the territories are generally occupied by one male strider and a pair of female striders ('keystone'). The major advantage of the keystone is mating. Here, a targeted female is sent a courtship calling ripple signal from a male, and the female responds by sending back an attraction or repulsion ripple signal. Nevertheless, whatever the female response, the male strider generally mounts the female. However, females can avoid effective mating via unique mechanisms [25]. The following formula, Eq. (10), is presented for updating the position of the keystone to decide mate or repel:

$$= \begin{cases} WS_i^{t+1} \\ WS_i^t + R.rand_i \quad if\ mating\ happens \\ (with\ probability\ of\ 50\%) \\ WS_i^t + R.(1 + rand_i) \quad otherwise \end{cases} \quad (10)$$

In Eq. (6), WS_i^t denotes the location of the ith WS in the tth cycle, and $rand_i$ indicates the ith vectors using arbitrary values that lie between zero and one. R indicates a vector that initiates at the keystone location (WS_i^{t-1}) and ends at the location of the targeted females (WS_F^{t-1}).

As compared to males, a major advantage of female WSs is their ability to find food. Thus, entomologists call them 'optimal foraging-habitat users', which implies that they typically occupy a space using food. After the mating procedure, the keystone needs to replenish the consumed energy. Therefore, when the keystone's new position in the mating technique runs out of food, the keystone moves to the area using one of the meals in the manner described as follows:

$$WS_i^{t+1} = WS_i^t + 2rand.\left(WS_{BL}^t - WS_i^t\right), \quad (11)$$

In Eq. (11), WS_{BL}^t denotes where the WS is located by applying its optimal cost value. Notably, it is assumed that the WS cannot seek food since it is unable to increase the previous cost values.

Due to the resident's propensity for aggressive territorial behaviour, entering an unfamiliar area might be perilous. Aggression between territorial residents and intruders is so severe that this might result in murder. Here, the keystone perishes when it is unable to find enough food to restore its energy level, and a new WS inside its region is chosen to replace it as a novel keystone in the manner described in Eq. (12):

$$WS_i^{t+1} = Lb_j^t + 2rand\left(Ub_j^t - Lb_j^t\right), \quad (12)$$

In Eq. (12), Lb_j and Ub_j indicate the maximal and minimal values of WSs^1 location inside *thejth* territories, respectively.

Finally, when the end criteria are satisfied, the process ends with reporting the optimal location found so far. Or else it returns to the mating phase for a novel loop.

For enhanced classification efficiency, the WSO system resolves a fitness function (FF). For indicating the higher performance of candidate outcomes, it solves a positive integer. In this instance, it has been assumed that FF is delivering

Algorithm 1 Pseudo-Code of the WSO Algorithm

Input: Maximal number of analyses (Max NA), population size (*nws*) and number of territories (*nt*)
Output: Effective WS production location at the lowest cost and with the highest value
Arbitrary population initialization
Compute the fitness value of WS
while (stopping criteria is unsatisfied) do
    Create *nt* number of territories and assign WSs
    for (every territory) do
    The male keystone causes the elected female to respond by sending mating ripples
    Upgrade keystone location
    Determine a new location for finding food to make up for energy lost during mating
    If (keystone fails to recognize food) then
    Scavenge for food, then head toward an area with plenty of food
        if (keystone fails to recognize food again), then
        Keystone's hunger will result in death
        A mature larva substitute killed keystone
        End
    end
end
Return $WS_{optimum}$

a classifier error rate that is as low as possible, as shown in Eq. (13).

$$fitness\,(x_i) = Classifier\ Error\ Rate\,(x_i) = \frac{\text{number of misclassified samples}}{\text{Total number of samples}} * 100 \quad (13)$$

IV. PERFORMANCE VALIDATION

This work uses the proposed model simulated via Google Colab and the Python 3.6.5 tool.

We have collected our own dataset from schools in and around Puducherry and Tamil Nadu, India. Each paper is scanned and saved in image format as JPG. Using conventional ball-point pens, gel pens and ink pens, samples were gathered on notebooks and ordinary writing materials. We also gathered samples of sketch pens, which generate thick strokes. All document pages were scanned at 500 dpi and converted to a digitized image format as input image see figure a with an HP Scanjet 2500 Pro Flatbed Scanner, and each was assigned a three-digit number name (from 001 to 250) with the extension '.jpg'. This dataset is a real-time dataset that contains 250 pages after augmentation techniques contains 590 pages, which are segmented into 8350 lines. For experimental validation, ten-fold cross validation technique is used.

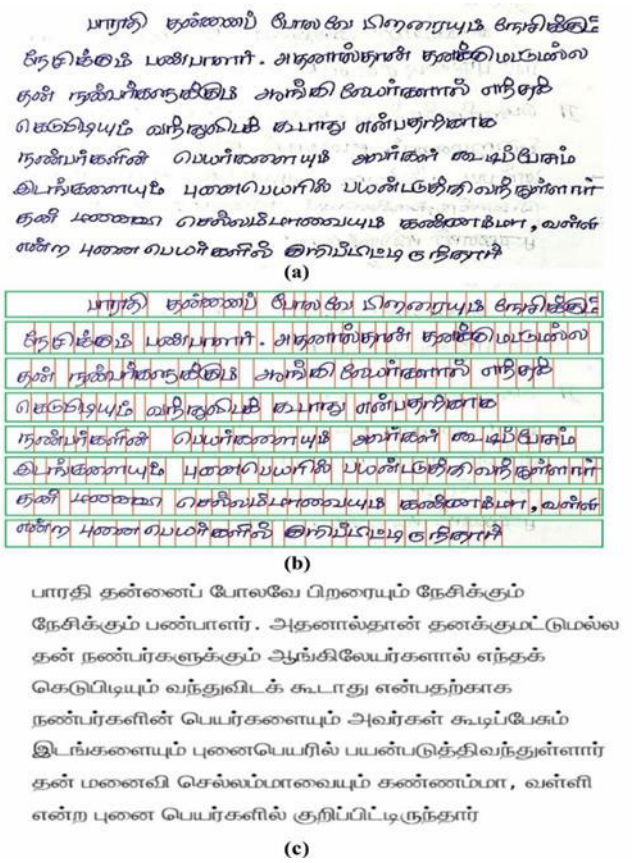

FIGURE 3. Sample results: (a) input text; (b) segmented text; (c) recognized output.

Figure 3 shows the sample visualization output of the ETEDL-THDR model. Figure 3a shows the input handwritten text as a scanned document, while Figure 3b indicates the segmented lines and segmented characters in the lines. Finally, Figure 3c exhibits the recognized output by the ETEDL-THDR model.

Table 1 and Figure 4 report the overall HCR performance of the ETEDL-THDR model via different epochs. The outcome implies that the ETEDL-THDR model produces effective results in all epochs. Using 500 epochs as an instance, the ETEDL-THDR model offers an accuracy of 98.08%, precision of 97.32%, sensitivity of 96.25%, specificity of 96.85% and F-measure of 97.20%.

Also, on 1000 epochs, the ETEDL-THDR approach produces an accuracy of 98.61%, precision of 99.14%, sensitivity of 98.75%, specificity of 99.21% and F-measure of 99.27%. Moreover, on 1500 epochs, the ETEDL-THDR methodology has an accuracy of 98.83%, precision of 98.19%, sensitivity of 97.89%, specificity of 97.88% and F-measure of 98.26%. Finally, on 2000 epochs, the ETEDL-THDR system presented an accuracy of 98.40%, precision of 98.87%, sensitivity of 99.01%, specificity of 99.15% and F-measure of 98.68%.

The comprehensive average result of the ETEDL-THDR model under distinct epochs is analyzed in Figure 5.

TABLE 1. Result analysis of ETEDL-THDR approach by different epochs and measures.

No. of epochs	Accuracy	Precision	Sensitivity	Specificity	F-measure
Epoch-500	98.08	97.32	96.25	96.85	97.20
Epoch-1000	98.61	99.14	98.75	99.21	99.27
Epoch-1500	98.83	98.19	97.89	97.88	98.26
Epoch-2000	98.40	98.87	99.01	99.15	98.68
Average	**98.48**	**98.38**	**97.98**	**98.27**	**98.35**

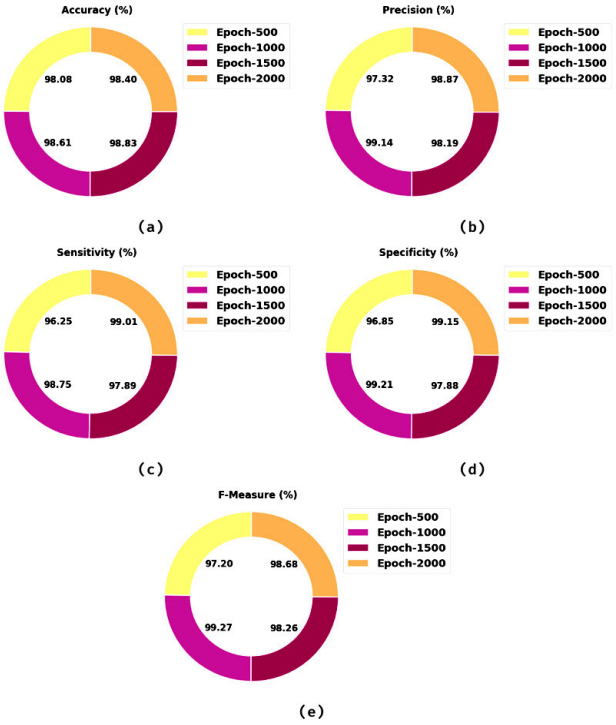

FIGURE 4. Result analysis of the ETEDL-THDR approach: (a) accuracy, (b) precision, (c) sensitivity, (d) specificity, (e) F-measure.

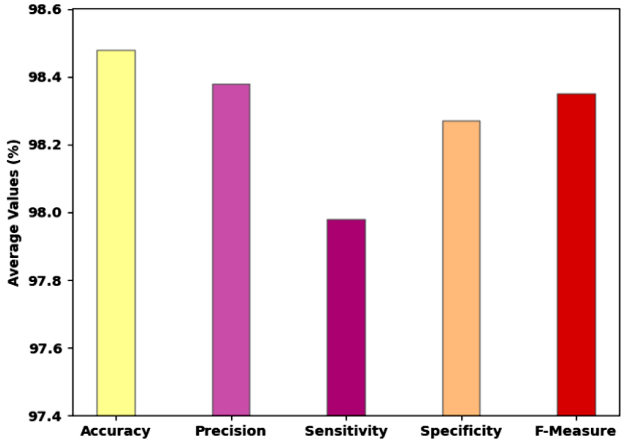

FIGURE 5. Average result analysis of the ETEDL-THDR approach.

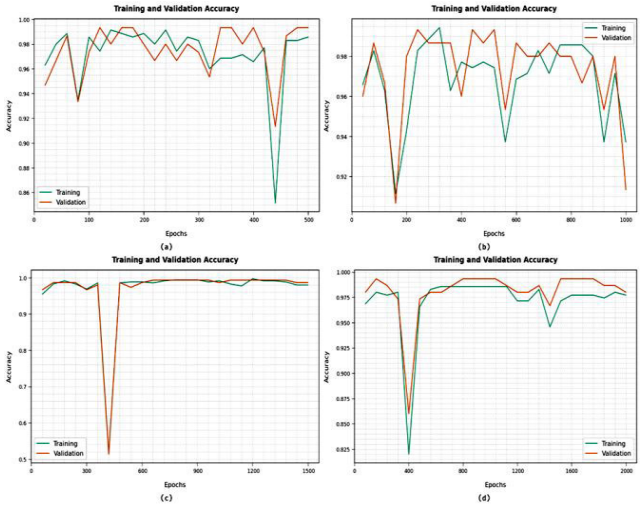

FIGURE 6. Analysis of the ETEDL-THDR TA and VA approach: (a) epoch 500, (b) epoch 1000, (c) epoch 1500, (d) epoch 2000.

According to the findings, the ETEDL-THDR model has improved performance with a maximum average accuracy of 98.48%, precision of 98.38%, sensitivity of 97.98%, specificity of 98.27% and F-measure of 98.35%.

Figure 6 shows the training accuracy (TA) and validation accuracy (VA) that the ETEDL-THDR technique was able to achieve under various epochs. The results of the experiment demonstrated that the TA and VA maxima obtained by the ETEDL-THDR approach were high. Particularly, the VA has specifically been greater than the TA.

Figure 7 illustrates the training loss (TL) and validation loss (VL) that the ETEDL-THDR method obtained under various epochs. The results of the trial showed that the ETEDL-THDR system attained the TL and VL minimum values. Notably, the VL appeared to be lesser than the TL.

Figure 8 exhibits the AUC analysis of the ETEDL-THDR method under several epochs. The experimental results stated that the ETEDL-THDR method reaches increasing AUC values under all epochs.

Table 2 presents a thorough examination of how the ETEDL-THDR model compares to other existing models [11], [13], [15]. Figure 9 showcases the sensitivity and specificity inspection of the ETEDL-THDR model to other existing models. According to the graph, the Support Vector Machine (SVM) and Elephant Herd Optimization (EHO) models led to lower values in sensitivity and specificity. The SALA and NN models have simultaneously obtained marginally improved sensitivity and specificity values. Also, the clustering-groupwise and modified NN models have accomplished moderately closer values of sensitivity and specificity. Though the CNN, DNN-VGG 16, DNN-Inception and DNN-DenseNet 169 models have attained reasonable sensitivity and specificity values, the presented ETEDL-THDR model has demonstrated competitive sensitivity and specificity of 97.98% and 98.27%, respectively.

Figure 10 illustrates the precision and F-measure examination of the ETEDL-THDR approach against existing

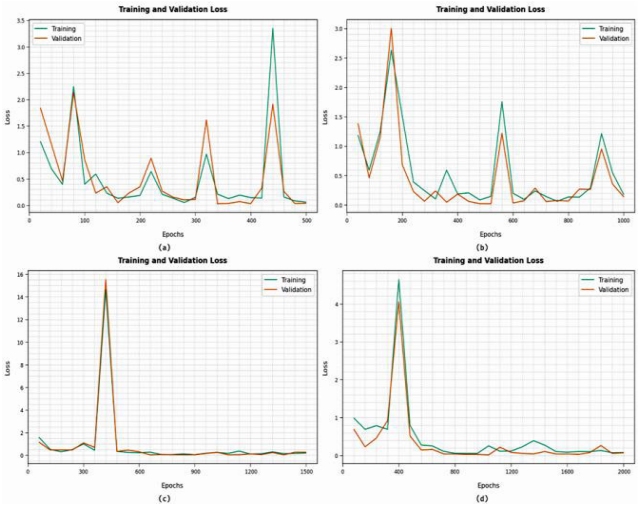

FIGURE 7. Analysis of the ETEDL-THDR TL and VL approach: (a) epoch 500, (b) epoch 1000, (c) epoch 1500, (d) epoch 2000.

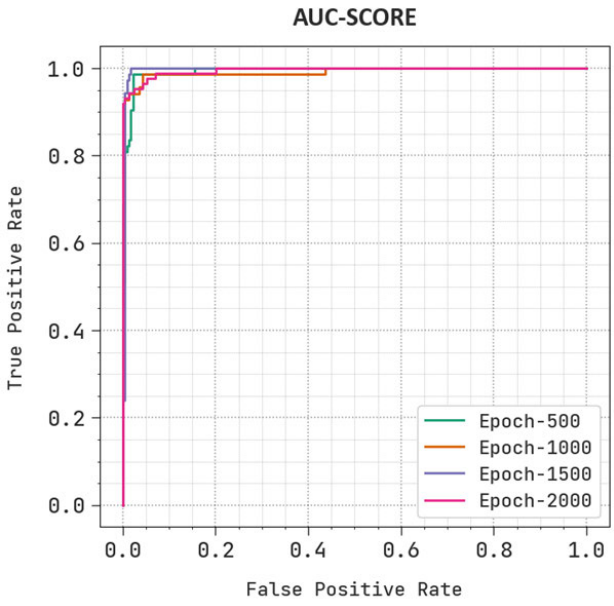

FIGURE 8. AUC Analysis of the ETEDL-THDR under varying epochs.

methods. The figure reveals that the SVM and EHO systems have resulted in lesser values of precision and F-measure, while the SALA and NN algorithms have reached somewhat enhanced values of precision and F-measure. Also, the clustering-groupwise and modified NN approaches have accomplished moderately closer values of precision and F-measure.

However, while the CNN, DNN-VGG 16, DNN-Inception and DNN-DenseNet 169 models have attained reasonable precision and F-measure values, the presented ETEDL-THDR system has outperformed them in competitive precision and F-measure, with values of 98.38% and 98.35%, respectively.

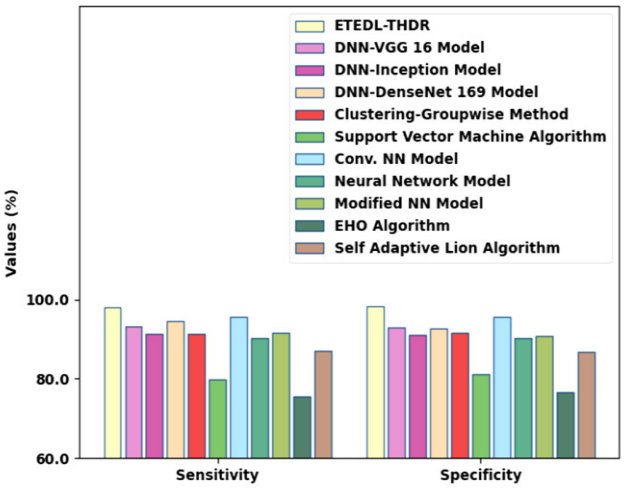

FIGURE 9. Sensitivity and specificity analysis of the ETEDL-THDR approach against existing methodologies.

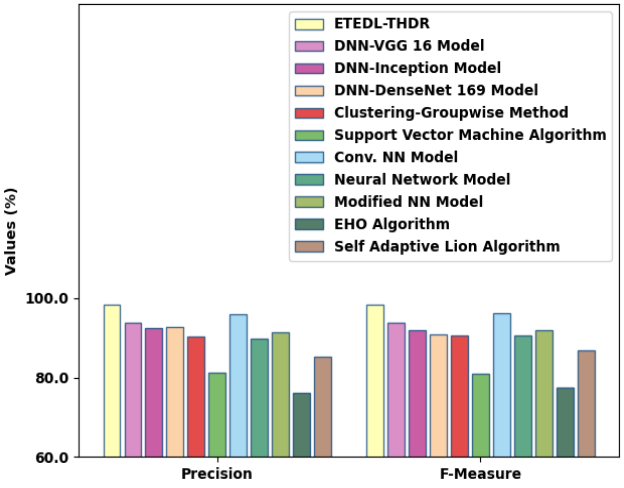

FIGURE 10. Precision and F-measure analysis of the ETEDL-THDR approach against existing methodologies.

Figure 11 provides an accurate examination of the ETEDL-THDR approach against other recent models. The graph shows that the SVM and EHO models produced inferior results, with the least accuracy of 82.04% and 76.84%, respectively. In line with this, the clustering-groupwise, NN and SALA models have reached moderately closer accuracy values of 89.66%, 88.78% and 86.39%, respectively. Next, the modified NN and CNN models have shown acceptable accuracy values of 92.86% and 89.66%, respectively. Finally, the DNN-VGG 16, DNN-Inception and DNN-DenseNet 169 models have reported closer accuracy values of 92.93%, 94.99% and 91.04%, respectively. However, the proposed model has produced better results with a maximum accuracy of 98.48%.

Finally, a detailed computation time (CT) analysis of the ETEDL-THDR model against recent models is carried out in Table 3 and Figure 12. The results reveal that the SVM, EHO and SALA models have shown poor performance with

TABLE 2. Comparative analysis of the ETEDL-THDR approach with existing methodologies.

Methods	Accuracy	Precision	Sensitivity	Specificity	F-measure
ETEDL-THDR	**98.48**	**98.38**	**97.98**	**98.27**	**98.35**
DNN-VGG 16 model	92.93	93.76	93.29	92.97	93.88
DNN-Inception model	94.99	92.42	91.32	91.08	92.00
DNN-DenseNet 169 model	91.04	92.76	94.48	92.76	90.75
Clustering-groupwise method	89.66	90.21	91.40	91.59	90.73
Support vector machine algorithm	82.04	81.17	79.90	81.22	80.96
CNN model	94.70	96.02	95.68	95.49	96.23
Neural network model	88.78	89.92	90.39	90.22	90.69
Modified NN model	92.86	91.48	91.66	90.78	91.95
EHO algorithm	76.84	76.06	75.65	76.63	77.38
Self-Adaptive Lion Algorithm	86.39	85.11	87.03	86.69	86.92

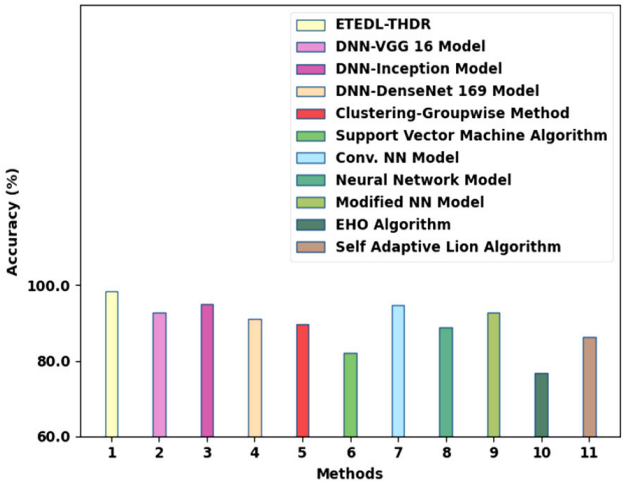

FIGURE 11. Comparative accuracy analysis of the ETEDL-THDR approach against existing methodologies.

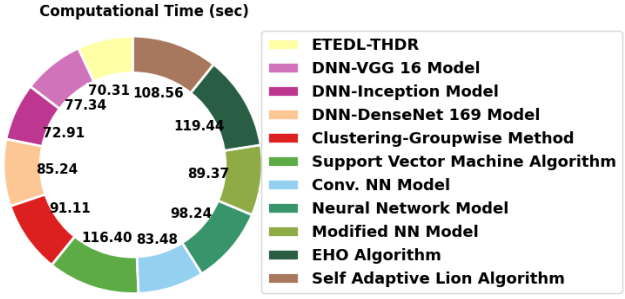

FIGURE 12. CT analysis of the ETEDL-THDR algorithm against existing methodologies.

higher CTs of 116.4s, 119.44s and 108.56s, respectively. Also, the clustering-groupwise, NN and modified NN models have demonstrated closer CTs of 91.11s, 98.24s, and 89.37s, respectively.

It is seen that the CNN, DNN-VGG 16, DNN-Inception and DNN-DenseNet 169 models have resulted in reasonable

TABLE 3. CT analysis of the ETEDL-THDR algorithm against existing methodologies.

Methods	Computational time (sec)
ETEDL-THDR	**70.31**
DNN-VGG 16 model	77.34
DNN-Inception model	72.91
DNN-DenseNet 169 model	85.24
Clustering-Groupwise method	91.11
Support vector machine algorithm	116.4
CNN model	83.48
Neural network model	98.24
Modified NN model	89.37
EHO algorithm	119.44
Self-Adaptive Lion Algorithm	108.56

CTs of 83.48s, 77.34s, 72.91s and 85.24s, respectively. However, the ETEDL-THDR model has reached a minimal CT of 70.31s.

These findings and the accompanying discussion demonstrate that the ETEDL-THDR model outperforms all others. As a result, the proposed model may be used to recognize THCs in real-world settings. The enhanced performance of the proposed model is due to the incorporation of WSO algorithm for hyperparameter tuning process, which identifies the optimal combination of hyperparameters that leads to the highest accuracy on the applied dataset.

V. CONCLUSION

In this work, a novel ETEDL-THDR technique for the identification of THCs was developed. To accomplish this, the ETEDL-THDR approach incorporated a series of processes such as MF-based preprocessing, segmentation, MobileNet feature extraction, BiGRU recognition and WSO hyperparameter optimization. To effectively modify the hyperparameters related to the BiGRU model, the WSO algorithm was utilized, which helps in improving the recognition performance. A broad experimental investigation was conducted

to verify the improved performance of the ETEDL-THDR model. Extensive comparative results reported the enhanced performance of the ETEDL-THDR approach as compared to other recent methodologies. In future, an ensemble of three DL-based fusion models can be introduced further to enhance the recognition results of the ETEDL-THDR model.

DATA AVAILABILITY

The training and testing trajectory datasets are available on request at the following ResearchGate URL: https://www.researchgate.net/publication/362490821_Tamil_Handwritten_Documents_Dataset

DECLARATIONS

Conflict of interest: The authors declare no competing interests.

REFERENCES

[1] M. A. Pragathi, K. Priyadarshini, S. Saveetha, A. S. Banu, and K. M. Aarif, "Handwritten Tamil character recognition using deep learning," in *Proc. Int. Conf. Vis. Towards Emerg. Trends Commun. Netw. (ViTECoN)*, 2019, pp. 1–5.

[2] P. Banumathi and G. M. Nasira, "Handwritten Tamil character recognition using artificial neural networks," in *Proc. Int. Conf. Process Autom., Control Comput.*, Jul. 2011, pp. 1–5.

[3] A. A. Prakash and S. Preethi, "Isolated offline Tamil handwritten character recognition using deep convolutional neural network," in *Proc. Int. Conf. Intell. Comput. Commun. Smart World (I2C2SW)*, Dec. 2018, pp. 278–281.

[4] N. Shanthi and K. Duraiswamy, "A novel SVM-based handwritten Tamil character recognition system," *Pattern Anal. Appl.*, vol. 13, no. 2, pp. 173–180, May 2010.

[5] A. Bhardwaj, "Handwritten Devanagari character recognition using deep learning-convolutional neural network (CNN) model," *PalArch's J. Archaeol. Egypt/Egyptol.*, vol. 17, no. 6, p. 7965, 2020.

[6] N. Sasipriyaa, P. Natesan, R. Anand, P. Arvindkumar, R. S. A. Prakadis, and K. A. Surya, "Recognizing handwritten offline Tamil character by using cGAN & CNN," in *Proc. Int. Conf. Sustain. Comput. Data Commun. Syst. (ICSCDS)*, Apr. 2022, pp. 509–515, doi: 10.1109/ICSCDS53736.2022.9760808.

[7] B. Rajyagor and R. Rakhlia, "Handwritten character recognition using deep learning," *Int. J. Recent Technol. Eng. (IJRTE)*, vol. 8, no. 6, p. 5815, 2020.

[8] R. Vaidya, D. Trivedi, S. Satra, and P. M. Pimpale, "Handwritten character recognition using deep-learning," in *Proc. 2nd Int. Conf. Inventive Commun. Comput. Technol. (ICICCT)*, Apr. 2018, pp. 772–775.

[9] S. Akashkumar, A. N. Dyaram, and M. Anand, "Identification of Tamil characters using deep learning," in *Proc. Mach. Learn. Auton. Syst. (ICMLAS)*. Singapore: Springer, 2022, pp. 223–237.

[10] P. Gnanasivam, G. Bharath, V. Karthikeyan, and V. Dhivya, "Handwritten Tamil character recognition using convolutional neural network," in *Proc. 6th Int. Conf. Wireless Commun., Signal Process. Netw. (WiSPNET)*, Mar. 2021, p. 84, doi: 10.1109/WiSPNET51692.2021.9419451.

[11] B. R. Kavitha and C. Srimathi, "Benchmarking on offline handwritten Tamil character recognition using convolutional neural networks," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 4, pp. 1183–1190, Apr. 2022.

[12] C. Vinotheni, S. L. Pandian, and G. Lakshmi, "Modified convolutional neural network of Tamil character recognition," in *Proc. Adv. Distrib. Comput. Mach. Learn. (ICADCML)*. Singapore: Springer, 2020, pp. 469–480.

[13] R. B. Lincy and R. Gayathri, "Optimally configured convolutional neural network for Tamil handwritten character recognition by improved lion optimization model," *Multimedia Tools Appl.*, vol. 80, no. 4, pp. 5917–5943, Feb. 2021.

[14] R. Jayakanthan, A. H. Kumar, N. Sankarram, B. S. Charulatha, and A. Ramesh, "Handwritten Tamil character recognition using ResNet," *Int. J. Res. Eng., Sci. Manage.*, vol. 3, no. 3, p. 133, 2020.

[15] S. Kowsalya and P. S. Periasamy, "Recognition of Tamil handwritten character using modified neural network with aid of elephant herding optimization," *Multimedia Tools Appl.*, vol. 78, no. 17, pp. 25043–25061, Sep. 2019.

[16] N. Ulaganathan, J. Rohith, A. S. Abhinav, V. Vijayakumar, and L. Ramanathan, "Isolated handwritten Tamil character recognition using convolutional neural networks," in *Proc. 3rd Int. Conf. Intell. Sustain. Syst. (ICISS)*, Dec. 2020, pp. 383–390.

[17] K. Shanmugam and B. Vanathi, "Newton algorithm based DELM for enhancing offline Tamil handwritten character recognition," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 36, no. 5, Apr. 2022, Art. no. 2250020.

[18] R. Ghosh, P. P. Roy, and P. Kumar, "Smart device authentication based on online handwritten script identification and word recognition in indic scripts using zone-wise features," *Int. J. Inf. Syst. Model. Design*, vol. 9, no. 1, pp. 21–55, Jan. 2018.

[19] R. Ghosh, C. Vamshi, and P. Kumar, "RNN based online handwritten word recognition in devanagari and Bengali scripts using horizontal zoning," *Pattern Recognit.*, vol. 92, pp. 203–218, Aug. 2019.

[20] C. Sureshkumar and T. Ravichandran, "Handwritten Tamil character recognition and conversion using neural network," *Int. J. Comput. Sci. Eng.*, vol. 2, no. 7, p. 2261, 2010.

[21] A. Shah, J. I. Bangash, A. W. Khan, I. Ahmed, A. Khan, A. Khan, and A. Khan, "Comparative analysis of median filter and its variants for removal of impulse noise from gray scale images," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 3, pp. 505–519, Mar. 2022.

[22] J. Su, J. Faraone, J. Liu, Y. Zhao, D. B. Thomas, P. H. Leong, and P. Y. Cheung, "Redundancy-reduced MobileNet acceleration on reconfigurable logic for ImageNet classification," in *Proc. Int. Symp. Appl. Reconfigurable Comput.*, vol. 16, 2018, pp. 16–28, doi: 10.1007/978-3-319-78890-6_2.

[23] S. Behera, R. Misra, and A. Sillitti, "Multiscale deep bidirectional gated recurrent neural networks based prognostic method for complex non-linear degradation systems," *Inf. Sci.*, vol. 554, pp. 120–144, Apr. 2021.

[24] A. Kaveh and A. D. Eslamlou, "Water strider algorithm: A new metaheuristic and applications," *Structures*, vol. 25, pp. 520–541, Jun. 2020.

[25] B. Liu and S. Pouramini, "Multi-objective optimization for thermal comfort enhancement and greenhouse gas emission reduction in residential buildings applying retrofitting measures by an enhanced water strider optimization algorithm: A case study," *Energy Rep.*, vol. 7, pp. 1915–1929, Nov. 2021.

C. VINOTHENI received the B.Tech. degree from the Sri Manakula Vinayagar Engineering College, in 2016, and the M.Tech. degree from the Pondicherry Engineering College, in 2018. She is currently a Researcher with Puducherry Technological University. She has authored more than six papers published in international conferences and professional journals. Her research interests include natural language processing, deep learning, image processing, neural networks, and artificial intelligence.

S. LAKSHMANA PANDIAN received the B.Eng. degree in electrical and electronics engineering and the M.Eng. degree in computer science and engineering from the Government College of Engineering, Tirunelveli, in 1993 and 1998, respectively, and the Ph.D. degree from the Department of CSE, Anna University, Chennai, in 2011. He is currently an Associate Professors with the Department of CSE, Puducherry Technological University (formerly Pondicherry Engineering College), India. He has more than 30 publications in international journals. He has presented more than 20 papers in international conferences. He has guided both UG and PG candidate's projects. His research interests include language technology (natural language processing, compilers, automata theory, and computations), embedded systems, and data analytics.