IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 1, 2021, accepted May 4, 2021, date of publication May 14, 2021, date of current version May 24, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3080325

Segments Interpolation Extractor for Finding the Best Fit Line in Arabic Offline Handwriting Recognition Words

HAITHAM Q. GHADHBAN[1], MUHAINI OTHMAN[1], NOOR SAMSUDIN[1], SHAHREEN KASIM[1], AISYAH MOHAMED[1], AND YAZAN ALJEROUDI[2]

[1]Software Engineering Department, Computational Intelligence and Data Analytics (CIDA), Uinversiti Tun Hussein Onn Malaysia, Parit Raja 86400, Malaysia
[2]Kulliyyah of Engineering, International Islamic University Malaysia, Kuala Lumpur 50728, Malaysia

Corresponding author: Haitham Q. Ghadhban (haithamq11@gmail.com)

This work was supported by the Uinversiti Tun Hussein Onn Malaysia through the TIER 1 Grant Scheme under Grant Vot H787.

ABSTRACT In the last few years, deep learning-based models have made significant inroads into the field of handwriting recognition. However, deep learning requires the availability of massive labelled data and considerable computation for training or automatic feature extraction. The role of handcrafted features and their significance is still crucial for a specific language type because it is a unique way of writing the characters. These are primitive segments that describe the letter horizontally or vertically distinguish an Arabic letter. This article develops a new type of feature for handwriting using Segments Interpolation (SI) to find the best fitting line in each of the windows and build a model for finding the best operating point window size for SI features. The experimental design was done on two subsets of the Institute for Communications Technology/Ecole Nationale d'Ingénieurs de Tunis (IFN/ENIT) database. The first one contains 10 classes (C10), and the second one has 22 classes (C22). The extracted features were trained with Support Vector Machine (SVM) and Extreme Learning Machine (ELM) with different kernels and activation functions. The evaluation metrics from a classification perspective (Accuracy, Precision, Recall and F-measure) were applied. As a result, SI shows significant results with SVM 90.10% accuracy for C10 and 88.53% accuracy for C22.

INDEX TERMS Arabic handwriting word recognition, classification, ELM, feature extraction, segments interpolation and SVM.

I. BACKGROUND

Handwriting recognition is a dynamic model and simulation environment that is considered a part of pattern recognition. It can contribute an essential benefit to our real life [1]. The diversity of handwriting recognition comes with extensive usage of a massive number of costly computational aspects. Currently, the technology provides an exceptionally smooth technique and, at the same time, hides the bright side of handwriting text. Several applications where handwriting recognition is necessary, such as bank cheques [2], postal addresses [3], and handwritten form processing [4].

Numerous studies on handwriting recognition, especially for the Latin script [2], [3], have been conducted over the last few decades. There are quite good results for machine printed text recognition with over 99% accuracy, for instance. However, in Arabic handwriting recognition as against Latin [7], only minor studies have been carried out. Due to the intricacy of Arabic text and insufficient databases about this language [8]. Recognition of Arabic text is in the initial phases compared to the recognition of Chinese, Latin, and Japanese manuscripts. Furthermore, there are several challenges in Arabic writing recognition practices from the data's cursive form. These challenges arise due to several aspects, like the Arabic writing setup that is cursive, the pen, the writing style, and other elements.

Arabic handwriting recognition consists of two classes, online and offline [9], respectively. First, the characteristics of the handwritten text determines the class of online handwriting recognition, which includes real-time conversion of text as it is produced. In the real-time scenario, the writing medium is typically a Personal Digital Assistant (PDA),

The associate editor coordinating the review of this manuscript and approving it for publication was Zahid Akhtar.

tablet, or a flat panel. Such media record data about the motion, location, and probably the pressure applied on the surface, and the data is subsequently fed to the recognition system. Secondly, offline text recognition comprises the text being produced using pen and paper before being fed into the recognition system [10]. Offline systems require scans of handwritten text, and typically the scanned images are initially enhanced. The next step comprises the extraction of features from the bitmaps using digital image processing methods. Offline handwriting recognition is also referred to as Optical Character Recognition (OCR), encompassing printed text recognition [11]. Offline recognition of Arabic handwriting is a problematic research domain, in contrast to the Latin script, which is the direction of writing in Arabic from right to left. From a presentation standpoint, the form of an Arabic letter varies with its placement in a word, neighboring characters with different shapes of writing based on their position in word or overlap, which is especially important in Arabic. Additionally, there are numerous ligatures; the Arabic script's specified aspects present several challenges to offline handwriting recognition.

The literature studies indicate that the number of published Arabic studies of handwriting has little to do with other text-identifying languages than the number of published Arabic language studies [12]. Most studies on Arabic handwriting recognition tackle an isolated character, word, or digit recognition [7], [8].

Academics have found many kinds of challenges in recognizing offline Arabic handwriting [15], such as the Arabic language is written in a cursive form with overlapping characters [16]. Due to these overlapping characters, the separation of words in Arabic writing is complicated and needs to utilize contextual information in many cases. The overlapping of characters makes the assignment of dots or diacritics a challenging task. Arabic handwriting contains many ligatures. Some of these ligatures are optional. Ligatures are difficult to segment into component characters. They may be treated as separate characters, and writing styles have a terrible effect in some cases. Many Arabic characters have ascenders and descenders, which means that words from different lines touch each other even if they are not on the same baseline. There are also some other issues that the researchers in offline Arabic handwriting recognition have to deal with, for example, difficulties due to the writing process and scanning. The scanning process may introduce noise from the scanner bed-page border.

The number of studies in Figure 1 illustrates the number of published articles in Google scholar using five keywords: Arabic handwriting recognition, Latin handwriting recognition, Chinese handwriting recognition, Japanese handwriting recognition, and English handwriting recognition languages considering the years between 2015 and 2021. We can observe the number of Arabic studies has risen from 2230 studies in 2015 to 2930 studies in 2020. Figures 1 reported in 2021, which is focused on a growing number of published articles, promised a higher publication

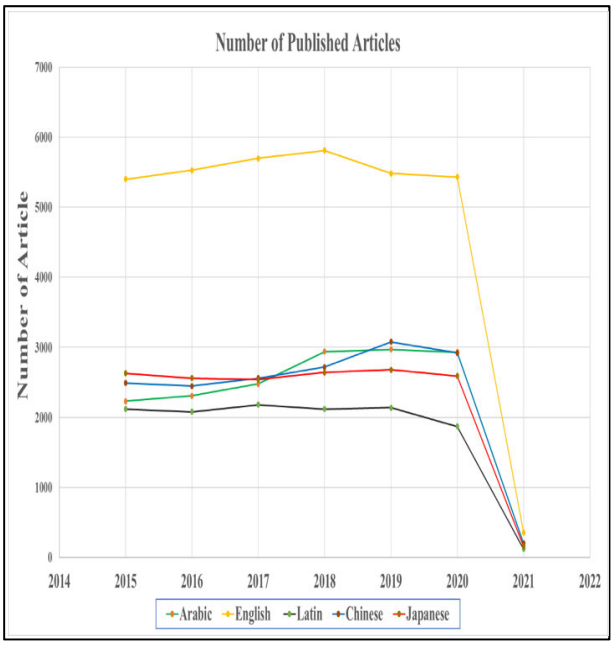

FIGURE 1. Illustrate the number of publication of handwriting recognition.

than in the previous years. The remaining languages can see how the number of studies has fluctuated. In comparison, we can see a substantial increase in handwriting recognition in the number of Arabic studies.

However, these issues and difficulties make Arabic word decomposition into letters a very delicate process and not always ensured due to variability in writing styles from one person to another. It makes the problem of recognition of Arabic words complex and challenging. Many methods and techniques for other languages are not directly applicable to the Arabic script. Methods for recognizing Arabic handwriting should also consider these Arabic handwriting characteristics. Hence, building handwriting recognition of the Arabic language based on multi-type features is helpful to increasing the system's performance. Increasing the number of extracted features is not adequate, but it is needed to provide features with a different nature or perspective of describing the Arabic letters. Reviewing the used features for Arabic handwriting, we find two main categories: one is based on a geometrical assumption about the way of drawing the Arabic letters, which might not be an accurate assumption, and the other is based on mathematical transformation. The issue with the former category is the lack of data-supportive techniques like interpolation or regression. Therefore, this article aims to present the following contributions:

- Investigate the existing feature extractors of handcrafted features with the techniques and implementation in different domains.
- Develop a new geometric feature extraction model that is suitable to capture patterns inside Arabic handwriting images. More specifically, Segment Interpolation (SI) helps approximate the various letters in Arabic.

- Adopting features requires identifying the best operating point or setting for the data considered since dataset images have one or more words in the image. This model required an automatic model to obtain the best window size of the developed SI features.
- A comprehensive evaluation of the developed contributions came from a classification perspective. Four evaluation metrics considered for classification are as follows: Accuracy, Precision, Recall, and F-measure.

The rest of the paper is organized as follows. The related works in Section II and Section III present the proposed method for segment interpolation. Section IV presents the experimental setup, including the dataset, and Section V provides the discussion of our approach with confusion matrix and comparison methods. Finally, Section VI concludes the paper.

II. RELATED WORKS

There are two ways to extract features from an image. Firstly, the most classical way is to design features specially tuned for the task and sometimes even tuned for a specific dataset. These features are referred to as handcrafted features since the algorithm used to extract them was designed manually, incorporating a priori information on the data's specificities. Some of the handcrafted features are very simple, while some more recent feature sets are complex and generally highly dimensional. Secondly, learning features are an image in machine learning, which has been used more since the advent of deep learning. The following section will focus on the first category of the handcrafted feature using techniques and implementation in the Arabic handwriting recognition domain.

A new offline Arabic word recognition technique suggests segmenting words into characters. Metwally *et al.* [17] also adopted the segmentation of training data into individual letters for recognition. The Hidden Markov Model (HMM) is trained for every letter of the alphabet while also considering several writing styles and letter positions. The scenario requires varying features extracted from different dataset images, thereby enhancing the system's recognition accuracy. The method was tested using the IFN/ENIT database with the findings reported. Jayesh *et al.* [18] proposed another approach. A competent offline Arabic handwritten word recognition method. The HMM technique adopted the sliding window mechanism to extract significant features, and the model was examined in two words based on systematic and holistic approaches without word segmentation.

Another system for offline recognition of Arabic handwritten words presented, which was based on HMM, where they consider the contextual character information [19]. The approach is methodical, segmentation-free, and feature-free, based on baseline approximation to integrate peculiarities of both the pixel and text distribution of the word image. This method [20] used a Scale Invariant Feature Transform (SIFT) extractor to help them divide the image into 16 equal-sized frames. It then divided each frame into another 16 sub-frames, achieving directional orientation and gradient measure to facilitate the fit of the entire image. Akram and Khalid [21] introduced this approach to scanning the image from right to left of line text image by sliding window technique. Each window is divided vertically into fixed cells and extracts information from each cell, and then the feature vectors are trained on HMM. Tavoli *et al.* [22] developed a new feature extractor based on straight lines found in the word's geometric and quantitative characteristics, referring to Hough's theory's works. They used the location, length, angle, and number of straight lines for recognition.

This approach [23] introduced a model where they used a geometric feature to fit each character. They divided the image into four frames and extracted pixels with values between 255 and 0. Once the positive pixels are identified, they combine pixels with straight lines and use the line length as an extraction feature. Moreover, the study has used Soble operators to identify the characters against the border and edge accurately. This approach was introduced by Arbain *et al.* [24] to explore triangular geometry as the best fit for handwriting recognition. They split an image using the Zoning technique and assessed each frame individually by applying triangular geometry as feature extraction. They were able to obtain high percentages of fits that lead to recognition once frames were reassembled.

Tamen *et al.* [25] developed Statistical and Contour Features (SCF) to extract local information based on the word images' contour. The SCF method is conducted in several steps to extract information: Firstly, written pixels are colored on the background. Secondly, the ratio between the sum of foreground pixels. Thirdly, the image area and the sliding windows of different sizes. These features allow us to distinguish between globally similar words but whose character content is different. These features can enhance the information provided by moments.

Elleuch *et al.* [26] proposed a handcrafted feature as input with supervised learning, the design feature Histogram of Oriented Gradients (HOG) is used to extract features from textual images. They used a technique to subdivide the image into small connected and equally spaced regions called cells by calculating pixels in each cell.

Hassan and Alawi [27] also implemented a Discrete Wavelet (DWT) technique, which divides each picture into several layers, then evaluates every layer with a high vertical and low vertical coefficient filter, accompanied by a diagonal filter that all combined to give the overall coefficient. Abdalkafor [28] applied a directional filter via zoning technique that split the image into nine grids: zoning techniques that split the image into nine grids that were all extracted individually. The feature was built on Euler no1, which is known for producing good results when applied to binary images with topological structures. Another group of researchers [2] studied stroke direction from which they built a gradient vector for finding local structural similarities with the Gabor filter. Table 1 examines variation

TABLE 1. Demonstrate feature extractor implementation in different domains.

Authors	Techniques	Databases	Domain	Accuracy
Alizadeh et al. [29]	Zernike moments are applied to eliminate the parts of the image that do not have helpful information.	MG63 dataset	Integrative Biology	The recognition rate is 96%.
Wang et al. [30]	The descriptor was applied to extract protein evolutionary information from the Position-Specific Scoring Matrix.	Yeast and H.pylori datasets.	Protein–protein interactions	The recognition of Yeast is 94.48% and H.pylori is 91.25%.
Fredo et al. [31]	The global and local damages that occurred after different impingements such as 5 mm, 6 mm, and 7 mm are classified using shape measures and SVM.	Fabricated by hand layup technique.	Composite materials	The recognition rate is 96.6%.
Li et al. [32]	A novel sphere center projective model was used and improved the sub-pixel edge-locating algorithm.	camera	Vision measurement	Higher accuracy accomplished under complicated illumination terms.
Shu et al. [33]	Efficient computation of Chebyshev applied to binary and grayscale images for binary images using image block representation.	MPEG-7	Image recognition	The computation decreases when blocks are smaller than the image size.
Benouini et al. [34]	Propose a new set of orthogonal moments named (FCM) based on the definition of fractional-order Chebyshev polynomials, which can be essential for image representation.	MPEG-7, Butterfly and Coil-20	Object recognition	MPEG-7=90% Butterfly=57% Coil-20=86%

feature extraction techniques and their application in other domains.

The existing feature extractors ignored the single type of feature that behaves in the same level of discrimination for all handwriting data classes. Due to the writing changes between words in the Arabic letters and types of handwriting, the summary of this analysis, despite the numerous studies proposed for handwriting in general and Arabic handwriting in particular. The geometrical nature of letters in terms of the percentage of segment components (Vertical or Horizontal) in the writing combination was not having enough attention and has been associated with systematic handcrafted feature extraction. The previous studies motivated us to provide a novel type of feature that depends on Arabic word segments.

III. PROPOSED METHODS

This section presents the developed methodology for accomplishing the objectives. It starts with presenting the feature extractor Segment Interpolation (SI) in Sub-Section A. Furthermore, it provides a model for selecting the optimal operating point of this novel feature presents in Sub-Section B. This makes it generalizable to any handwriting data. Afterward, feature normalization is presented in Sub-Section C. This stage is essential in any machine learning application. Next, the individual-based classifier's training concerning features means that each classifier will be trained separately on each SI feature type. We provide two types of classifiers: SVM and ELM in Sub-Section D.

A. SEGMENTS INTERPOLATION (SI)

The proposed feature extractor SI was designed to extract the best fitting line in each window images. Because some images have more than one word written in one image, the widths (W) of the image are not equal to the heights (H). Applying a fixed window size to the entire image with the unequal size is critical.

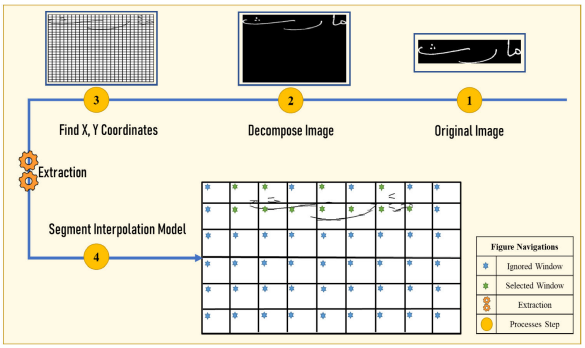

FIGURE 2. Segment Interpolation (SI) Model.

We need to decompose the image into equal size H × W as shown in Figure 2 step 2, and we extracted SI features based on decomposing the image into equal parts called windows in step 3. Next, we run a Least Square Estimation (LSE) to find the best fitting line in each window in step 4. The LSE process is a form of mathematical regression analysis to determine the best fit for several data and to demonstrate the relationship of the data points visually. Every data point represented the relation between a known independent and an unknown variable, as is shown in equation (1).

$$y_t = \beta_0 + \beta_1 x_t + \varepsilon_t \tag{1}$$

The coefficients denote the intercept β_0 and the slope β_1 of the line respectively, the intercept represents the predicted value of when $x = 0$. The slope represents the average predicted change resulting from a one-unit increase. The random error term ε_t does not imply a mistake but a deviation from the underlying straight-line model. It captures anything that may affect others. We use a sample to find the estimated regression line from equation (1). The sample is showing in equation (2).

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 x \tag{2}$$

where $\hat{Y}$ is the predicted value, $\hat{\beta}_0$ is estimated from β_0, and $\hat{\beta}_1$ is estimated from β_1. Then we need to find the residual

distances between the point and line through this equation (3).

$$e_i = Y_i - \hat{Y}_i \tag{3}$$

where Y_i is observed, and $\hat{Y}_i$ is predicted, to minimize the sum of the total vertical distance between point and line through this equation (4).

$$\begin{aligned} \sum e_i^2 &= \sum \left(Y_i - \hat{Y}_i\right)^2 \\ &= \sum \left(Y_i - \hat{Y}_i\right)^2 \\ &= \sum \left(Y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)\right)^2 \end{aligned} \tag{4}$$

To find $\hat{\beta}_0$ and $\hat{\beta}_1$, we use these equations (6) and (7) based on the points provided from the image (X_i, Y_i) in equation (5), Where $X = [X_1\ X_2 \ldots X_n]^T$ and $Y = [Y_1\ Y_2 \ldots Y_n]^T$.

$$\hat{\beta} = [\hat{\beta}_1 \hat{\beta}_2] = (X'X)^{-1} X'Y \tag{5}$$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n} (X_i - \bar{X})^2} \tag{6}$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X} \tag{7}$$

The slops of the lines are concatenated together to combine vectors of features called SI features. Algorithm 1 described the steps of SI.

In Figure 3, it is observed from the two images that the features have the same pattern regardless of the differences between the two images. The features can be discriminative from the perspective of words.

Algorithm 1 Segment Interpolation (SI) Features Extraction

Input: Image, Window Size
Output: SI
Initialization:
1: [no. of Blocks Heights, no. of Blocks Widths] = decompose (Image, window Size) LooP Process
2: **for** i = no. of Blocks Heights **do**
3: **for** j = no. of Blocks Widths **do**
4: XY = get XY of window (Image, i, j)
5: [slop, intercept] = LSE(X Y)
6: Add to SI
7: **end for**
8: **end for**

We present a conceptual example in Figure 4 that includes the letter (Kaaf). As we observe, the window in part shown contains a straight-line with an offset of 3 pixels and a slope of 45 degrees, which generates a pair of features (3,1).

B. OPERATING POINT SELECTOR FOR THE SI FEATURES

We visualized three values of features for three different configurations. Window size equals 20, 30, and 40, as shown in Figure 5, which indicates that feature values are different. This implies that the determination of the best configuration is a crucial part of the features' performance.

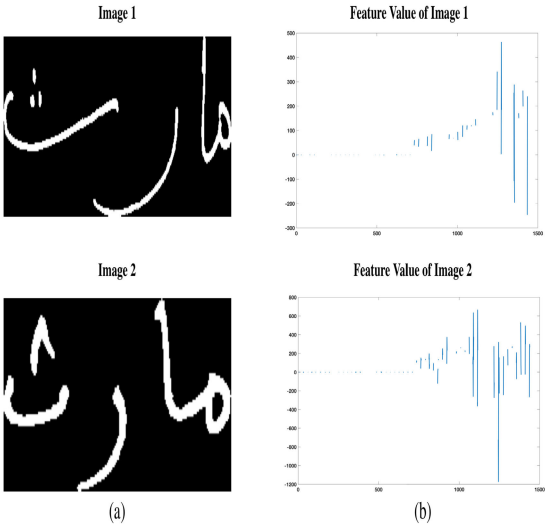

FIGURE 3. Visualize two images for same word: (a) The same word was written twice for SI; (b) SI features for the two images that represents the same word (window size = 20).

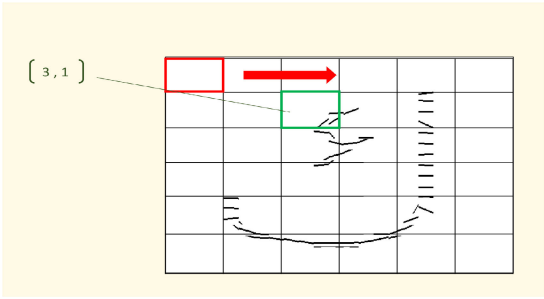

FIGURE 4. Visualize the straight-line in window with offset.

The proposed model aims to build a point selector model for the SI feature. The model is based on a greedy algorithm. It starts with the minimum possible window size. It goes gradually from one candidate size to the next and compares the corresponding training accuracy with the best one obtained so far to select the best among them. The greedy approach used for determining the best window size and awareness of falling in local minima is the convexity of the optimization surface. The pseudocode of the model pointer selector is shown in Algorithm 2.

To show the meaning of the window size in the SI feature results, in Figure 6, an example is visualized for extracting SI feature in two cases of window size: the first case is when the window size is 20, and the second case is for window size 12, as we observe the granularity increases with decreasing the window size, the discrimination of the feature changes with the style of writing and the nature of the ink. Hence, it is better to use a model for deciding the best operating point of the SI window.

C. FEATURE NORMALIZATION

The feature normalization is crucial for transforming the features' values to a standard boundary presented to the

Algorithm 2 Greedy Model of Selecting the Best Window Size of SI Features

Input: Data = {Image, Label}, Window Size Min, Window Size Max, Classifier
Output: Best Window
Initialization:
LOOP Process

```
for Window Size = Window Size Min to Window Size Max do
    Best Window = Window Size Min
    for i = Image of Data do
        SI = extract SI (i, Window Size)
        add SI to SI Data
    end for
    [train, test ] = decompose (SI_Data, Label)
    Train classifier on train
    accuracy = Test classifier on test
    if accuracy Old < accuracy then
        Best Window = Window Size
    endif
end for
```

classifiers, as shown in equation (8).

$$X_N = 2\frac{X - \text{Min}(X)}{Max\ (X) - \text{Min}(X)} - 1 \tag{8}$$

X denotes the original features and X_N indicates the features after normalization. This equation will guarantee that the features will be placed within the boundary of $[-1, 1]^{n\times m}$ where represents the number of records and denotes the number of features.

D. CLASSIFICATION

This section presents the classifiers that are applied in the framework. There are no restrictions on the classifier type, and the framework is flexible enough to use any classifier. There are two types of classifiers used. The first classifier is a single hidden layer feed-forward neural network named extreme learning machine, and the second is a kernel variant classifier called support vector machine. The classifiers are denoted as C(P, F). P represents the set of parameters used with the classifier and indicates the type of Kernel or activation function used for the classifier. Assuming that we have k main types of features and have l varieties of classifiers, we train each feature type classifier. Then we will have total $k \times l$ classifiers. Assume that the statistical description of the features/classes varies from one image to another. All the considered classifiers are described in the following.

1) EXTREME LEARNING MACHINE

The main drawback of Feed Forward Neural Networks (FFNN) is the slow processing and adaptability rate. Each iteration takes a long time and is a significant hindrance to the widespread use and scalability of FFNN. The main drivers causing this are gradient-based learning algorithms and iterative fine-tuning of the network parameters that constantly change when FFNN is trained. Extreme learning machines presented by Huang *et al.* [35] for Single Hidden Layer Feed-forward Neural networks (SLFNs) improve learning speed by randomly choosing the input weights and letting SLFN calculate the output weights. The study argues, this method provides optimal generalized results and performance without compromising on learning speed. Comparing ELM with FFNN, the key standout is speed, and ELM's learning speed is very fast [36]. In most models and simulations during ELM learning, the results can achieve in less than a second [37]. With the advantage of speed,

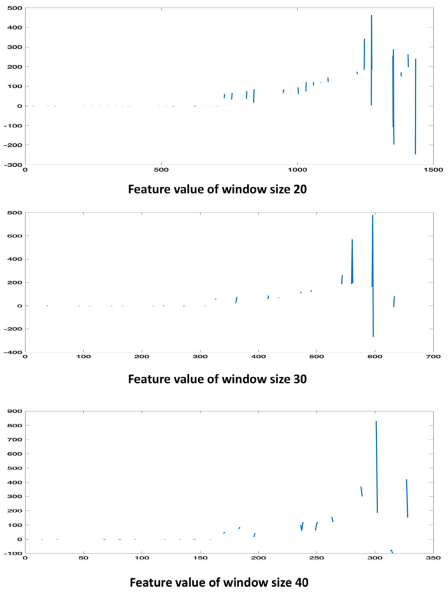

FIGURE 5. Illustrated feature value of one image with different windows size.

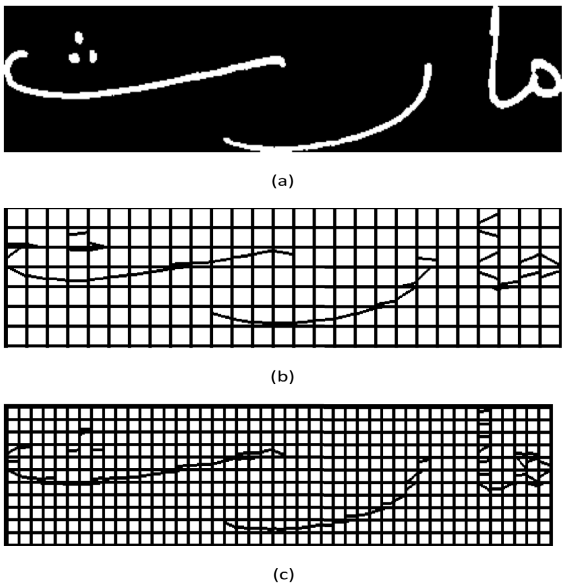

FIGURE 6. Visualize image with different window size:(a) Handwriting image, (b) SI image with window size 20 and (c) SI image with window size 12.

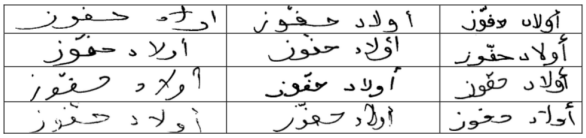

FIGURE 7. Variety of handwriting styles.

ELM has overall better generalization abilities than FFNN and backpropagation learning. Another advantage of ELM is developing SLFNs with various non-differentiable activation roles, compared to traditional gradient-based algorithms that can only be applied to differential activation functions. Here are several ELM studies in the area of recognition that show promising results, such as handwritten character recognition [38], alphabet recognition [39], ELM-based decision rule [40], handwritten numeral extraction [41], and ELM optimized for image food [42].

2) SUPPORT VECTOR MACHINE

Support vector machine is based on statistical learning. It is a form of supervised learning introduced by Vapnik *et al.* [43]. This study was applied for two groups of classification. Typically used for classifying linear and non-linear data, SVM has the predictive capability required for non-linear problem-solving. Hence, for tasks like classification, regression, and clustering. SVM chooses the extreme points that help create the hyperplane, and these extreme cases are called support vectors. The goal of the SVM classifier is to make the best line or decision boundary that can segregate n-dimensional space into classes so that we can quickly put data points in the correct category. Here are several SVM studies in the field of classification that show significant results, such as character recognition [44], [45], character recognition based kernel performance) [46], decision-making [47], and for combination to diagnose transformer faults [48].

IV. EXPERIMENTAL SETUP

We conducted extensive experimental research; the source is written in MATLAB 2018b and runs on a CPU i5-2.4 GHz machine with 8GB RAM. The feature extracted is based on two subsets presented in sub-section (A). Classifiers use ELM and SVM. The SI feature is trained separately on each classifier, and then these features are evaluated based on metrics in sub-Section (B). The summary of the parameters used for feature extraction and classification is showing in Table 2.

A. IFN/ENIT DATABASE

The main issue of Arabic handwriting recognition lacks databases. A few of those databases are freely available to advance Arabic research and enhance word Arabic handwriting recognition. Pechwitz *et al.* [49] proposed the Institute for Communications Technology/Ecole Nationale d'Ingénieurs de Tunis (IFN/ENIT) overcome the lack of Arabic datasets freely available for scholars. IFN/ENIT contains 946 city names. 411 writers have written these cities' names, and each writer requested to fill a form with preselected cities names,

TABLE 2. The parameters that are used for feature extraction and classification.

Parameter	Value
Image size	180 × 180
Window size	20
Activation function	Sigmoid, Sine, Hardlim, Tribas and RBF
Minimum number of neurons	50
Maximum number of neurons	1000
Kernel type	Linear, Polynomial and RBF
MaxIter	400000
Polyorder	3
RBF_sigma	1
Boxconstraint	10

TABLE 3. Statistics of IFN/ENIT.

Sets	Number of images	Number of characters	PAW
A	6538	51984	28298
B	6709	53862	29220
C	6479	52155	28391
D	6732	54166	29511
E	6032	45169	22640

TABLE 4. Summary of dataset.

Dataset	Total	Train	Test	Max per class	Min per class	Number of classes
C10	2945	2066	879	371	129	10
C22	5443	3822	1621	371	53	22

as shown in Figure 7. these forms have been scanned with high resolution and convert image to black and white.

There are 32492 images in IFN/ENIT, distributed into five sets. Each set includes names of cities, Parts of Arabic Words (PAW), and characters, as shown in Table 3.

In the experiment environment and based on the benchmark [25], two subsets of the dataset were used. The first subset consists of 10 classes (C10), and the second subset consists of 22 classes (C22). Each subset is split into 70% training, and the rest are testing, details shown in Table 4.

B. EVALUATION MEASURES

This section presents the evaluation approach of the developed objective. Four evaluation metrics for classification are considered to assume that the image sample is tested to indicate a specific word. The class of a specific word is positive, and the classifier's result can be either positive or negative. Also, for either one of the two decisions, it can be either true or false. This means we have in total as follow: True Positive (TP) represents the data that is positive and correctly classified, True Negative (TN) represents the data that is negative and correctly classified, False Positive (FP) represents data that is negative but incorrectly classified as positive

data, and False Negative (FN) represents data that is positive but incorrectly classified as negative data. We measured the following four types of metrics:

1) ACCURACY

This measure indicates the percentage of misclassification over the total number of tested samples. Another way to express it is by the two types of classifications: the positive ones' P and the negative ones' N and their subsets false F and true T [50], as shown in equation (9). Overall accuracy based on test data has been calculated based on correctly predicted classes divide by the total number of testing classes, as shown in equation (10).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (9)$$

$$\text{Test accuracy} = \left(\frac{Sum\,(\text{TP})}{Total\ test\ number}\right) \cdot 100 \quad (10)$$

2) PRECISION

Precision is metric that calculates the sum of TP for entire classes divided by the sum of TP and FP for all classes [50], as shown in equation (11).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (11)$$

3) RECALL

A recall is a metric that calculates the sum of TP for all classes divided by the sum of TP and FN across entire classes [50], as given in equation (12).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (12)$$

4) F-MEASURE

This measure expresses the fusion of the two measurements, precision, and recall [50]. The formula is given in equation (13).

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (13)$$

V. EXPERIMENTAL RESULTS

This section presents the generated results for the accomplished the proposed methods. It explains the effects of SI features on two sub-set of IFN/ENIT datasets C10 and C22. Then we present individual accuracy for each class, including the confusion matrix and a comparison with previous methods.

A. EVALUATION OF SI FEATURES

The extracted features were tested with an SVM classifier for three kernels: linear, polynomial, and Radial Basis Function (RBF). Besides, the model tested with ELM classifier for five types of activation functions: Sigmoid, Sine, Hardlim, Tribas and RBF, the selection number of neurons to be 400 was not arbitrary as shown in Table 5. The experiments were testing the number of neurons with a range of 50 neurons until 1000, increasing to 50 neurons per experiment but did not show that much difference in results. Hence, we proceed to deliver results with 100 neurons until 1000 neurons in the hidden layer. In both C10 and C22, the corresponding validation accuracy for the five activation functions is generated. The validation results of C10 as shown in Table 5. We have accomplished the best accuracy for 400 neurons in sigmoid function, and the validation results for C22 as shown in Table 6, and it performs the best accuracy for 800 neurons in sigmoid function.

TABLE 5. Validation accuracy for C10 SI features.

Activations	Sig (%)	Sine (%)	Hardlm (%)	Tribas (%)	RBF (%)
100	50	9	49	12	13
200	56	11	55	11	13
300	62	11	62	12	13
400	**68**	10	67	12	15
500	65	10	65	14	15
600	68	11	67	14	16
700	64	11	66	14	16
800	66	10	66	14	15
900	65	10	65	15	15
1000	63	12	65	13	14

TABLE 6. Validation accuracy for C22 SI features.

Activations	Sig (%)	Sine (%)	Hardlm (%)	Tribas (%)	RBF (%)
100	40	4	40	4	8
200	49	5	48	6	9
300	53	5	54	7	9
400	58	4	58	8	10
500	59	5	60	8	10
600	61	5	61	9	10
700	63	4	63	7	11
800	**66**	5	65	8	12
900	64	6	64	9	11
1000	64	5	64	8	11

In Table 7, the experiments were tested for both sub-set C10 and C22 in different kernels. According to the type of kernel, the testing accuracy for C10 ranged from 76% to 90%. The SI has generated more than 90% values for all metrics precision, recall, and F-measure. Hence, SI showed an excellent discriminative power for handwriting images. The testing accuracy for C22 ranged between 72% and 88%, besides the remaining classification metrics were all higher than 70%. In addition, another observation is the best performance achieved with the RBF kernel of SVM compared with the other kernels. This shows the non-linearity aspect of the data features, which have given the RBF kernel higher performance.

TABLE 7. The classification results of SI features based on SVM classifier with three types of kernels.

SI SVM C10 (Number of features 1459)			
Metrics	Linear kernel(%)	Polynomial kernel(%)	RBF kernel(%)
Test accuracy	76.50	89	**90.10**
Precision	76.12	89.52	**90.60**
Recall	76.09	89.07	**90.10**
F_Measure	76.11	89.29	**90.35**
SI SVM C22(Number of features 1459)			
Test accuracy	72	87	**88.53**
Precision	72.14	87.94	**89.16**
Recall	72.01	87.58	**88.53**
F_Measure	72.07	87.76	**88.84**

TABLE 8. The classification results of SI features based on ELM classifier with five activation function.

C10 ELM(Number of features 1459)					
Activations	Sigmoid (%)	Sine (%)	Hardlim (%)	Tribas (%)	RBF (%)
Neurons			400		
Test accuracy	**68**	10	67	12	15
Precision	**68.64**	10.29	68.03	13.05	15.47
Recall	**68.49**	10.42	67.89	12.89	15.71
F_Measure	**68.56**	10.35	67.96	12.97	15.59
C22 ELM (Number of features 1459)					
Neurons			800		
Test accuracy	**66**	5	65	8	12
Precision	NaN	5.54	NaN	8.09	**11.41**
Recall	**66.19**	5.92	65.96	8.18	12.06
F_Measure	NaN	5.72	NaN	8.13	**11.72**

The same experiments were repeated for different classifiers, a neural network type of classifier ELM shown in Table 8. The evaluation was done on different types of activation functions. The number of neurons that were used is 400 for C10 and 800 for C22. C10 achieved 68% accuracy with the Sigmoid function, while C22 achieved 66% accuracy with the Sigmoid function. In some cases, related to precision and F_meature in Table 8, shown NaN value due to TP and FN equal to zero. There are no positive cases in the input data, so any analysis of this case has no information. There is no conclusion about how positive cases are handled due to division by zero.

1) CONFUSION MATRIX

The confusion matrix is about to show in-depth the associate results for each classifier. Two types of classifiers are applied: the first one is ELM and the second one is SVM. To make

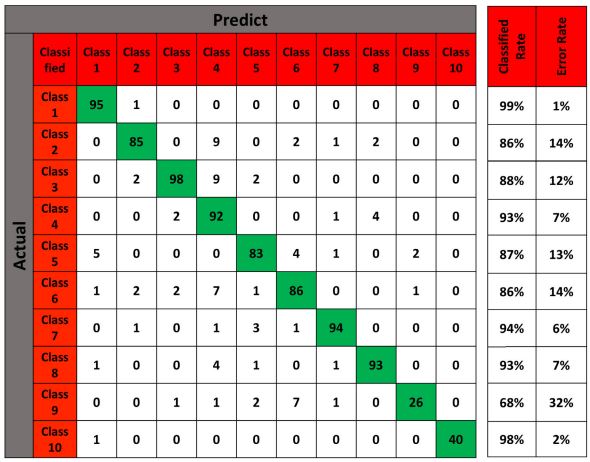

FIGURE 8. SI Confusion matrix of SVM with C10.

the presentation of the classifier-feature association easier. This section provides the confusion matrices of the SI feature with the SVM classifier in Figures 8. A confusion matrix is a viral method for solving classification problems [51], as it can be applied to both binary and multiclass classification problems. A standard confusion matrix is presented to provide detailed information on the type of errors that occur within and between classes. The confusion matrix also lists potential map classes and the totals within each class before summarizing the number of total images considered correct. This matrix helps determine whether a class is over-mapped, under-mapped, or randomly erroneous.

In Figure 8, the row corresponds to classes in the ground truth map (test set), and columns correspond to class classification results (predicted). The green cells on the diagonal elements in the matrix represent the number of correctly classified images of each class. The number of ground truths with a specific class name obtained the same class name during classification. In Figure 8 above, 95 images of "class 1" in the test set were correctly classified as "class 1" in the classified images. The off-diagonal elements (white cells) represent misclassified images or the classification error, the number of the ground truth of images that ended up in another class during classification.

A confusion matrix allows us to assess the accuracy of image classification by understanding the matrix. The classification results are compared to additional ground truth information. The strength of a confusion matrix is that it identifies the nature of classification errors and their quantities.

Overall accuracy essentially tells us, out of all the total numbers, what proportion were mapped correctly. The overall accuracy is usually expressed as a percent, with 100% accuracy being a perfect classification where all total was classified correctly. Accuracy is the easiest to calculate and understand but ultimately only provides the map class and single class with basic accuracy information. To calculate the overall accuracy by adding the number (correctly classified) and dividing it by the total number. In Figure 8, we can calculate the accuracy by dividing 792 (correctly classified

TABLE 9. Statistic accuracy of Segments Interpolation SI ELM-C10.

City name	Class	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	Error rate (%)	TP	FP	FN	TN
الدخّانية	1	95	75	83	79	5	80	27	16	756
شمّاخ	2	92	67	60	63	8	59	29	40	751
الرضّاع	3	93	71	72	71	7	80	33	31	735
نّقة	4	93	70	65	67	7	64	27	35	753
الفايض	5	93	71	63	67	7	60	24	35	760
الخليج	6	91	57	71	63	9	71	54	29	725
مارث	7	93	69	67	68	7	67	30	33	749
زّنوش	8	93	64	79	71	7	79	44	21	735
الشرايع	9	96	62	13	22	4	5	3	33	838
حي الإنطلاقة	10	99	86	90	88	1	37	6	4	832

TABLE 10. Statistic accuracy of Segments Interpolation SI SVM-C10.

City name	Class	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	Error rate (%)	TP	FP	FN	TN
الدخّانية	1	99	92	99	95	1	95	8	1	775
شمّاخ	2	98	93	86	89	2	85	6	14	774
الرضّاع	3	98	95	88	92	2	98	5	13	763
نّقة	4	96	75	93	83	4	92	31	7	749
الفايض	5	98	90	87	89	2	83	9	12	775
الخليج	6	97	86	86	86	3	86	14	14	765
مارث	7	99	95	94	94	1	94	5	6	774
زّنوش	8	99	94	93	93	1	93	6	7	773
الشرايع	9	98	90	68	78	2	26	3	12	838
حي الإنطلاقة	10	100	100	97	99	0	40	0	1	838

TABLE 11. Statistic accuracy of Segments Interpolation SI ELM-C22.

City name	Class ID	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	Error rate (%)	TP	FP	FN	TN
الدخّانية	1	95	57	72	64	5	69	50	27	1475
شمّاخ	2	95	58	48	53	5	48	35	51	1487
الرضّاع	3	95	61	70	65	5	78	49	33	1461
نّقة	4	95	64	58	61	5	58	32	41	1490
الفايض	5	94	53	56	55	6	53	46	42	1480
الخليج	6	96	64	70	67	4	70	40	30	1481
مارث	7	96	63	76	69	4	76	45	24	1476
زّنوش	8	97	71	82	76	3	82	33	18	1488
الشرايع	9	98	62	13	22	2	5	3	33	1580
حي الإنطلاقة	10	99	83	61	70	1	25	5	16	1575
حي النّضامن	11	96	NaN	0	NaN	4	0	0	15	1606
شعّال	12	96	66	78	71	4	81	42	23	1475
الفكّة	13	97	53	41	46	3	20	18	29	1554
شتاوةصحراوي	14	97	66	78	71	3	63	33	18	1507
أوتيك	15	98	71	45	55	2	17	7	21	1576
سبعة آبار	16	98	78	92	84	2	87	25	7	1502
الفحص	17	97	54	14	22	3	6	5	38	1572
المرناقيّة 20 مارس	18	98	80	83	82	2	60	15	12	1534
أكودة	19	97	73	75	74	3	77	28	25	1491
سيدي إبراهيم الزّهار	20	98	78	97	87	2	77	21	2	1521
شوّاط	21	98	45	18	26	2	5	6	23	1587
حي الصلاح	22	98	61	44	52	2	16	10	20	1575

images)/879 (total images) = 90.10% accuracy of SVM with ten classes.

A single class's accuracy is the fraction of correctly classified images concerning all images of that ground truth class. For each class of ground truth image (row), the number correctly classified is divided by the total number of the class's ground truth. In Figure 8, the ''class 1'', the accuracy is 95/96 = 98.95%, meaning that approximately 99%

TABLE 12. Statistic accuracy of Segments Interpolation SI SVM-C22.

City name	Class ID	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	Error rate (%)	TP	FP	FN	TN
الدخّانية	1	99	87	96	91	1	92	14	4	1511
شمّاخ	2	98	90	82	86	2	81	9	18	1513
الرضّاع	3	98	87	86	87	2	96	14	15	1496
نّقة	4	96	66	92	77	4	91	47	8	1475
الفايض	5	98	82	84	83	2	80	17	15	1509
الخليج	6	98	87	84	86	2	84	12	16	1509
مارث	7	99	90	91	90	1	91	10	9	1511
زّنوش	8	99	93	92	92	1	92	7	8	1514
الشرايع	9	99	89	66	76	1	25	3	13	1580
حي الإنطلاقة	10	99	92	90	91	1	37	3	4	1577
حي النّضامن	11	99	71	67	69	1	10	4	5	1602
شعّال	12	99	89	94	91	1	98	12	6	1505
الفكة	13	99	93	77	84	1	38	3	11	1569
شتاوةصحراوي	14	99	92	96	94	1	78	7	3	1533
أوتيك	15	99	93	76	84	1	29	2	9	1581
سبعة آبار	16	99	98	95	96	1	89	2	5	1525
الفحص	17	99	89	77	83	1	34	4	10	1573
المرناقيّة 20 مارس	18	99	94	96	95	1	69	4	3	1545
أكودة	19	99	98	90	94	1	92	2	10	1517
سيدي إبراهيم الزّهار	20	99	97	100	99	1	79	2	0	1540
شوّاط	21	99	84	75	79	1	21	4	7	1589
حي الصلاح	22	99	88	80	84	1	29	4	7	1581

for "class 1" or we can calculate it based on Equation (9) and apply the information provided in Table 10 by dividing 870/879 = 98.97% for "class 1".

In Table 9 -10-11 and 12, we considered the statistic accuracy for each classifier, including the evaluation measures.

B. COMPARISON BETWEEN SI AND OTHER METHODS

In this section, we used the results of our method to compare with three approaches applied to Arabic handwriting recognition words, namely Chebyshev Moments (CM), Zernike moments (ZER), and Statistical Counter Features (SCF) [25]. We applied the same dataset, C10, and C22, to these three approaches, CM, ZER, and SCF, to test the accuracy percentage. Our method demonstrated auspicious and competitive performance compared with other techniques. In Table 13, our proposed method succeeded in finding results with a test accuracy equal to 90.10% for ten classes and 88.53% for 22 classes using the SVM classifier, which was superior to Cm and ZER in test accuracy. On the other hand, we observe that SCF features have only provided competitive accuracy to our method in 10 classes, while its accuracy was inferior in 22 classes. Another comparison with state-of-the-art method [52], [53], dataset C10 And C22 applied to Convolutional Neural Networks (CovNets) with the same architecture and Multi-Dimensional Long Short Term Memory (MDLSTM). As shown in Table 13, CovNets outperforms our method in terms of accuracy. Overall, SVM classifiers give consistently better results for non-linear solution problems and are able to transform with a clear margin of class differentiation.

TABLE 13. Comparison between SI and other methods.

Methods	SVM		ELM	
	C10	C22	C10	C22
Chebyshev Moments (CM)	89%	86%	88%	82%
Zernike Moments (ZER)	84%	77%	61%	52%
Statistical Counter Features (SCF)	91%	88%	87%	83%
	C10		C22	
CovNets	80.82%		80.15%	
MDLSTM	84.35%		82.11%	
Our method	90.10%	88.53%	68%	66%

VI. CONCLUSION

The variation of handwriting recognition is common used for the large majority of expensive computing facets. Several issues have been found in the acknowledgment of offline Arabic. For example, Arabic is cursive written with alternating characters. Because of these issues, feature extractors'

availability overlooked the individual feature sort that discriminates all handwriting data classes at the same level. Our method was capable of handling these issues by segmenting each image into small windows and extracting relevant information. The operating mechanism to find the best window size that handles image width is applied. The mechanism is also aware of falling in local minima due to the optimization surface's convexity. In the classification process, the SVM displays substantial RBF kernel results relative to others due to the non-linearity nature of RBF kernel data features.

In comparison, due to random weights between input and hidden layers in the ELM architecture, ELM accuracy reduced from SVM due to misclassified ''Class 11'' ELM was deficient due to the poorly written samples in ''Class 11''. In future studies, we will extend our model from linear interpolation to parabolic or higher-order in an adaptive according to the nature of the text. In addition, we incorporate ensemble learning to counter the issue of data imbalance in many handwriting datasets.

REFERENCES

[1] P. Yadav, ''Handwriting recognition system–A review,'' *Int. J. Comput. Appl.*, vol. 114, no. 19, pp. 36–40, 2015.

[2] Q. Al-Nuzaili, S. Al-Maadeed, H. Hassen, and A. Hamdi, ''Arabic bank cheque words recognition using Gabor features,'' in *Proc. IEEE 2nd Int. Workshop Arabic Derived Script Anal. Recognit. (ASAR)*, Mar. 2018, pp. 84–89.

[3] M. Gopikrishna and B. Samatha, ''Handwritten digit recognition based on depth neural network,'' *Int. J. Manage., Technol. Eng.*, vol. 8, pp. 1493–1497, Feb. 2018.

[4] J. W. Barrus, E. L. Schwartz, and M. J. Gormish, ''Creating a dashboard for tracking a workflow process involving handwritten forms,'' U.S. Patent 9 870 352, Jan. 16, 2018.

[5] A. Kumar Bhunia, A. Das, A. Kumar Bhunia, P. Sai Raj Kishore, and P. Pratim Roy, ''Handwriting recognition in low-resource scripts using adversarial learning,'' 2018, *arXiv:1811.01396*. [Online]. Available: http://arxiv.org/abs/1811.01396

[6] P. Singh, P. P. Roy, and B. Raman, ''Writer identification using texture features: A comparative study,'' *Comput. Electr. Eng.*, vol. 71, pp. 1–12, Oct. 2018.

[7] A. Lawgali, ''A survey on Arabic character recognition,'' *Int. J. Signal Process., Image Process. Pattern Recognit.*, vol. 8, no. 2, pp. 401–426, Feb. 2015.

[8] J. H. AlKhateeb, ''A database for Arabic handwritten character recognition,'' *Procedia Comput. Sci.*, vol. 65, pp. 556–561, 2015.

[9] U. Tiwari, M. Jain, and S. Mehfuz, ''Handwritten character recognition—An analysis,'' in *Advances in System Optimization and Control*. Singapore: Springer, 2019, pp. 207–212.

[10] S. K. Jemni, Y. Kessentini, S. Kanoun, and J.-M. Ogier, ''Offline Arabic handwriting recognition using BLSTMs combination,'' in *Proc. 13th IAPR Int. Workshop Document Anal. Syst. (DAS)*, Apr. 2018, pp. 31–36.

[11] H. Q. Ghadhban, M. Othman, N. A. Samsudin, M. N. Bin Ismail, and M. R. Hammoodi, ''Survey of offline Arabic handwriting word recognition,'' in *Int. Conf. Soft Comput. Data Mining*, 2020, pp. 358–372.

[12] M. Rabi, M. Amrouch, and Z. Mahani, ''Hybrid HMM/MLP models for recognizing unconstrained cursive Arabic handwritten text,'' in *Int. Conf. Adv. Inf. Technol., Services Syst.*, 2017, pp. 438–448.

[13] K. Younis and A. Khateeb, ''Arabic hand-written character recognition based on deep convolutional neural networks,'' *Jordanian J. Comput. Inf. Technol.*, vol. 3, no. 3, p. 186, 2017.

[14] A. Alani, ''Arabic handwritten digit recognition based on restricted Boltzmann machine and convolutional neural networks,'' *Information*, vol. 8, no. 4, p. 142, Nov. 2017.

[15] S. Aloud, ''Handwriting Arabic words recognition based on structural,'' in *Proc. Book First Conf. Eng. Sci. Technol. (CEST)*, 2018, pp. 62–69.

[16] M. Rabi, M. Amrouch, and Z. Mahani, ''Cursive Arabic handwriting recognition system without explicit segmentation based on hidden Markov models,'' *J. Data Mining Digit. Humanities*, pp. 1–7, Jan. 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01458216/file/VSST%20paper.pdf

[17] A. H. Metwally, M. I. Khalil, and H. M. Abbas, ''Offline Arabic handwriting recognition using hidden Markov models and post-recognition lexicon matching,'' in *Proc. 12th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2017, pp. 238–243.

[18] K. Jayech, M. A. Mahjoub, and N. E. Ben Amara, ''Arabic handwritten word recognition based on dynamic Bayesian network,'' *Int. Arab J. Inf. Technol.*, vol. 13, no. 6B, pp. 1024–1031, 2016.

[19] M. Rabi, M. Amrouch, and Z. Mahani, ''Recognition of cursive Arabic handwritten text using embedded training based on hidden Markov models,'' *Int. J. Pattern Recognit. Artif. Intell.*, vol. 32, no. 1, Jan. 2018, Art. no. 1860007.

[20] A. K. A. Hassan, B. S. Mahdi, and A. A. Mohammed, ''Arabic handwriting word recognition based on scale invariant feature transform and support vector machine,'' *Iraqi J. Sci.*, vol. 60, no. 2, pp. 381–387, 2019.

[21] E. M. Hicham, H. Akram, and S. Khalid, ''Using features of local densities, statistics and HMM toolkit (HTK) for offline Arabic handwriting text recognition,'' *J. Electr. Syst. Inf. Technol.*, vol. 4, no. 3, pp. 387–396, Dec. 2017.

[22] R. Tavoli, M. Keyvanpour, and S. Mozaffari, ''Statistical geometric components of straight lines (SGCSL) feature extraction method for offline Arabic/Persian handwritten words recognition,'' *IET Image Process.*, vol. 12, no. 9, pp. 1606–1616, Sep. 2018.

[23] H. Mohamad, S. A. Hashim, and A. H. Al-Saleh, ''Recognize printed Arabic letter using new geometrical features,'' *Indonesian J. Elect. Eng. Comput. Sci.*, vol. 14, no. 3, pp. 1518–1524, 2019.

[24] N. A. Arbain, M. Sanusi, A. Kamilah, A. Ramzani, and A. Tahir, ''Triangle shape feature based on selected centroid for Arabic subword handwriting,'' *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 12, pp. 1–5, 2018.

[25] Z. Tamen, H. Drias, and D. Boughaci, ''An efficient multiple classifier system for Arabic handwritten words recognition,'' *Pattern Recognit. Lett.*, vol. 93, pp. 123–132, Jul. 2017.

[26] M. Elleuch, A. Hani, and M. Kherallah, ''Arabic handwritten script recognition system based on HOG and Gabor features,'' *Int. Arab J. Inf. Technol.*, vol. 14, no. 4A, pp. 639–646, 2017.

[27] A. K. A. Hassan and M. Alawi, ''Proposed handwriting Arabic words classification based On discrete wavelet transform and support vector machine,'' *Iraqi J. Sci.*, vol. 58, no. 2C, pp. 1159–1168, 2017.

[28] A. S. Abdalkafor, ''Designing offline Arabic handwritten isolated character recognition system using artificial neural network approach,'' *Int. J. Technol.*, vol. 8, no. 3, pp. 528–538, 2017.

[29] E. Alizadeh, S. M. Lyons, J. M. Castle, and A. Prasad, ''Measuring systematic changes in invasive cancer cell shape using zernike moments,'' *Integrative Biol.*, vol. 8, no. 11, pp. 1183–1193, 2016.

[30] Y. Wang, Z. You, X. Li, X. Chen, T. Jiang, and J. Zhang, ''PCVMZM: Using the probabilistic classification vector machines model combined with a Zernike moments descriptor to predict protein–protein interactions from protein sequences,'' *Int. J. Mol. Sci.*, vol. 18, no. 5, p. 1029, 2017.

[31] A. R. Jac Fredo, R. S. Abilash, R. Femi, A. Mythili, and C. S. Kumar, ''Classification of damages in composite images using Zernike moments and support vector machines,'' *Compos. B, Eng.*, vol. 168, pp. 77–86, Jul. 2019.

[32] Y. Li, J. Huo, M. Yang, and G. Zhang, ''Algorithm of locating the sphere center imaging point based on novel edge model and Zernike moments for vision measurement,'' *J. Mod. Opt.*, vol. 66, no. 2, pp. 218–227, Jan. 2019.

[33] H. Shu, H. Zhang, B. Chen, P. Haigron, and L. Luo, ''Fast computation of tchebichef moments for binary and grayscale images,'' *IEEE Trans. Image Process.*, vol. 19, no. 12, pp. 3171–3180, Dec. 2010.

[34] R. Benouini, I. Batioua, K. Zenkouar, A. Zahi, S. Najah, and H. Qjidaa, ''Fractional-order orthogonal chebyshev moments and moment invariants for image representation and pattern recognition,'' *Pattern Recognit.*, vol. 86, pp. 332–343, Feb. 2019.

[35] G. Huang, Q. Zhu, and C. Siew, ''Extreme learning machine: A new learning scheme of feedforward neural networks,'' in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2004, pp. 985–990.

[36] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, ''Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection,'' *IEEE Access*, vol. 6, pp. 33789–33795, 2018.

[37] Z. Wang, M. Li, H. Wang, H. Jiang, Y. Yao, H. Zhang, and J. Xin, "Breast cancer detection using extreme learning machine based on feature fusion with CNN deep features," *IEEE Access*, vol. 7, pp. 105146–105158, 2019.
[38] D. Das, D. R. Nayak, R. Dash, and B. Majhi, "An empirical evaluation of extreme learning machine: Application to handwritten character recognition," *Multimedia Tools Appl.*, vol. 78, pp. 19495–19523, Feb. 2019.
[39] J. Song, Q. Liu, S. Tian, Y. Wei, F. Jin, W. Mo, and K. Dong, "Research on handwritten alphabet recognition system based on extreme learning machine," in *Proc. 37th Chin. Control Conf. (CCC)*, Jul. 2018, pp. 9448–9451.
[40] D. A. Ramli and T. W. Chien, "Extreme learning machine based weighting for decision rule in collaborative representation classifier," *Procedia Comput. Sci.*, vol. 112, pp. 504–513, 2017.
[41] J. Ouyang, "Combining extreme learning machine, RF and HOG for feature extraction," in *Proc. IEEE 3rd Int. Conf. Multimedia Big Data (BigMM)*, Apr. 2017, pp. 419–422.
[42] S. K. Abdulateef, T.-A. N. Abdali, M. D. S. Alroomi, and M. A. A. Altaha, "An optimise ELM by league championship algorithm based on food images," *Indonesian J. Elect. Eng. Comput. Sci.*, vol. 20, no. 1, pp. 132–137, 2020.
[43] C. Cortes and V. Vapnik, "Support vector machines," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.
[44] M. Salam and A. A. Hassan, "Offline isolated Arabic handwriting character recognition system based on SVM," *Int. Arab J. Inf. Technol.*, vol. 16, no. 3, pp. 467–472, 2019.
[45] H. Althobaiti and C. Lu, "Arabic handwritten characters recognition using support vector machine, normalized central moments, and local binary patterns," in *Proc. Int. Conf. Image Process., Comput. Vis., Pattern Recognit. (IPCV)*, 2018, pp. 121–127.
[46] S. Fadel, S. Ghoniemy, M. Abdallah, and H. Abu, "Investigating the effect of different kernel functions on the performance of SVM for recognizing Arabic characters," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 1, pp. 446–450, 2016.
[47] Y. Liu, X. Wang, L. Li, S. Cheng, and Z. Chen, "A novel lane change decision-making model of autonomous vehicle based on support vector machine," *IEEE Access*, vol. 7, pp. 26543–26550, 2019.
[48] J. Liu, Z. Zhao, C. Tang, C. Yao, C. Li, and S. Islam, "Classifying transformer winding deformation fault types and degrees using FRA based on support vector machine," *IEEE Access*, vol. 7, pp. 112494–112504, 2019.
[49] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, and H. Amiri, "IFN/ENIT-database of handwritten Arabic words," in *Proc. CIFED*, vol. 2, 2002, pp. 127–136.
[50] M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, "Disease prediction by machine learning over big data from healthcare communities," *IEEE Access*, vol. 5, pp. 8869–8879, 2017.
[51] M. Hasnain, M. F. Pasha, I. Ghani, M. Imran, M. Y. Alzahrani, and R. Budiarto, "Evaluating trust prediction and confusion matrix measures for Web services ranking," *IEEE Access*, vol. 8, pp. 90847–90861, 2020.
[52] S. B. Ahmed, S. Naz, M. I. Razzak, and R. Yousaf, "Deep learning based isolated Arabic scene character recognition," in *Proc. 1st Int. Workshop Arabic Script Anal. Recognit. (ASAR)*, Apr. 2017, pp. 46–51.
[53] S. B. Ahmed, Z. Malik, M. I. Razzak, and R. Yusof, "Sub-sampling approach for unconstrained Arabic scene text analysis by implicit segmentation based deep learning classifier," *Global J. Comput. Sci. Technol.*, pp. 7–16, Mar. 2019.

HAITHAM Q. GHADHBAN received the bachelor's degree in software engineering from Al Rafidain University College, in 2013, and the master's degree in network technology from Universiti Kebangsaan Malaysia, in 2017. He is currently pursuing the Ph.D. degree with Universiti Tun Hussein Onn Malaysia. His research interests include pattern recognition, artificial intelligence, and machine learning.

MUHAINI OTHMAN received the bachelor's degree in information technology from the Northern University of Malaysia, in 1999, the master's degree in computer science from the University of Malaya, in 2006, and the Ph.D. degree in computer and mathematical science from the Auckland University of Technology, in 2015. She is currently a Senior Lecturer with the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia (UTHM). Her research interests include machine learning, computational intelligence, spiking neural networks, and personalized predictive modeling.

NOOR SAMSUDIN received the B.Sc. degree in computer science from the University of Missouri-Columbia, USA, the master's degree in information technology from the National University of Malaysia, and the Ph.D. degree in information technology from The University of Queensland, Australia. She is currently an Associate Professor with the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia (UTHM). She has received a number of research grants from UTHM and the Ministry of Education for Artificial Intelligence-related projects. Her research interests include machine learning, data mining, and IT applications in technical education.

SHAHREEN KASIM is currently an Associate Professor with the Department of Security Information and Web Technology, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia. Her research interests include bioinformatics, soft computing, data mining, Web, and mobile application.

AISYAH MOHAMED received the bachelor's degree in information technology from University Tun Hussein Onn Malaysia, where she is currently pursuing the master's degree in information technology. She has a few publications regarding temporal data, clustering, and spiking neural networks. She is mainly involved in personalized modeling in temporal data clustering based on Malaysia's real flood case study. Her research interests include exploring knowledge in data mining, deep learning, and artificial intelligent.

YAZAN ALJEROUDI received the bachelor's degree in computer engineering and automation from Damascus University, in 2007, and the master's degree in robotics from the University of Detroit Mercy, in 2011. He is currently pursuing the Ph.D. degree from International Islamic University Malaysia. His research interests include robotics, artificial intelligence, and machine learning.