Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 12, Issue. 11, November 2023, pg.53 – 68

Writer Dependent Offline Signature Verification Based on Cluster-Specific Classifier

Annapurna H[1]; Manjunatha K S[2]; Guru D S[3]

[1,3]Department of Studies in Computer Science, Manasagangothri, University of Mysore, Mysuru, Karnataka, 570006, India
[2]Department of Computer Science, Maharani's Science College for Women, Mysuru, Karnataka, 570005, India
[1] annapurnah2015@gmail.com; [2] kowshik.manjunath@gmail.com; [3] dsg@compsci.uni-mysore.ac.in

DOI: https://doi.org/10.47760/ijcsmc.2023.v12i11.005

Abstract— In this paper, we propose a writer dependent approach for offline signature verification based on writer-specific features and cluster-specific classifiers. In this work, writer-dependency is exploited at three levels: features, classifiers, and clusters. Initially, a template signature is selected for each writer from the training samples of that writer. This template signature serves as a representative signature of the respective writer. The relevant features for each writer are chosen using a filter-based feature selection method. The writers are then clustered based on their similar characteristics using the k-means algorithm. After clustering, a cluster-specific classifier is identified. This classifier is then set as the default classifier for all the writers in that cluster. During verification, writer-specific features and cluster-specific classifiers of the claimed writer are used to verify the authenticity of the given test signature. The approach is verified on three benchmarking offline signature datasets: CEDAR, MCYT, and GPDS-960.

Keywords— Offline signature, Template signature, Writer-specific features, Cluster-specific classifier, Clustering.

I. INTRODUCTION

Biometric authentication refers to the identification of an individual through their physiological traits such as fingerprints, facial features, hand geometry, retina, iris, etc., or behavioral traits such as handwritten signature, voice, and gait. A handwritten signature is one of the most widely used behavioral biometric traits to verify a person's identity due to its ease of capture and the familiarity of individuals with affixing their signature in daily life applications such as banking transactions, legal document validation, security systems, etc.[16].

The primary objective of a signature verification system is to automatically distinguish between genuine signatures and forgeries. Signature verification systems can be classified as offline or online depending on the acquisition mode. In an offline mode, cameras or scanners are used to acquire images of the signatures on paper documents. On the other hand, in the online mode, signatures are captured using specialized tools like pressure-sensitive tablets, smart pens, and other tools that can record dynamic aspects of a writer such as velocity, pressure, and acceleration during the signing [27]. Even though a lot of research has been done on signature verification in the past four decades, verification of the offline signature remains a more challenging task than online signature verification. This is primarily due to the absence of dynamic information and the relative ease with which offline signatures can be imitated when compared to online signatures [25].

A. *Preamble*

In this paper, a writer dependent approach for offline signature verification is proposed. This approach is based on writer-specific parameters, viz., features, classifiers, and cluster-specific classifiers. This is an extension of our earlier works [6],[36]. To improve the efficacy of the model, we have implemented this work with a different set of features. The writer-dependency is exploited at the feature level, feature dimension level, classifier level, and cluster level. All the experiments are conducted on three benchmarking offline signature datasets namely, CEDAR, MCYT, and GPDS-960.

We have conducted the experiments in four different ways:

- With a common set of features for all the writers with writer-specific classifiers.
- With writer-specific classifiers under varying feature dimensions.
- With writer-specific features and a common classifier across all writers.
- With writer-specific features and cluster-specific classifiers by varying the number of clusters.

B. *Related works*

Three distinct methods can be used for offline signature verification: 1. writer-independent method; 2. writer-dependent method; 3. Hybrid method [25]. In the writer-independent method, the model is trained using the same classifier with a set of features common across all writers. Whereas in the writer-dependent method, the model is trained separately for each writer using writer-specific parameters, viz., features, threshold, classifier, etc. The computational cost of this strategy is high, and it needs to be retrained when a new writer is added. The concepts of both the writer-dependent and writer-independent approaches have been combined in the hybrid method [17]. In this method, writer-dependent features are used in conjunction with writer-independent classifiers, and vice versa [16].

The majority of the models proposed in the literature for offline signature verification are writer-independent models. To represent offline signatures, three different types of features are used, viz., local [2]-[3],[19],[55], global[1],[8],[14],[19],[39], and deep features [23]. Compared to global and local features, the accuracy of the system has been enhanced by models trained with deep features [23].

The test signature is compared to the reference signatures during verification using the appropriate classification techniques. In literature, we can observe the application of various classifiers such as Naïve Bayes classifier [50],[53], support vector machine (SVM) [9],[10],[39],[55], Hidden Markov model (HMM)[14],[19],[32], Neural network (NN) [18],[23], Probabilistic neural network (PNN) [40], k-nearest neighbour (knn) [1],[53], symbolic classifier[42],[44], distance-based [2],[8], fuzzy-similarity measure [3] etc. SVM is the most popular classifier among these traditional machine learning techniques because it can deal with both linearly and nonlinearly separable data. Further, fusion-based [9],[45],[49],[59] methods have also been investigated for offline signature verification. Fusion-based machine learning methods have made the models more accurate than conventional machine learning methods. In recent models, deep architectures [2],[20],[28],[57]-[58] are being employed for classification purposes. Even though the use of deep architectures makes the models more accurate, it needs enough number of training samples and takes a long time to train them.

In the works listed above, it can be seen that the use of a common set of features and a single classifier or a common set of classifiers is used to determine whether a test signature is genuine or a forgery. So, these models are called writer-independent models. Although writer-independent models are computationally efficient, they do not take into account the characteristics of individual writers. Hence, few researchers have proposed writer dependent models for offline signature verification. In writer dependent models writer-dependency has been exploited at various levels, such as feature [36], threshold [22], and classifier [7],[11]-[12],[51]. Few researchers have proposed hybrid approaches for offline signature verification [17],[23],[60]. The models performed better with hybrid approaches, but they are computationally expensive.

C. *Findings*

However, in reality, there will be variations in the training signatures of different writers. It will lead to different distributions for different writers because some writers are more consistent in their signing than others. As a result, the use of common features and the same classifier may not be appropriate for all writers. Hence, the usage of writer-specific parameters at various stages like features, threshold, and classifier is necessary rather than employing a common set of features and classifiers for all writers. According to the literature, no attempt has been made to verify offline signatures using writer-specific parameters at multiple levels. This motivated us to design a writer-dependent model for offline signature verification based on multiple writer-specific parameters such as features, feature dimension, classifier, and cluster-specific classifier. Considering these issues, in this work, we propose an approach for offline signature verification by clustering different writers with similar characteristics and selecting a suitable classifier for each cluster.

The primary contributions of this paper are as follows:

- Offline signature verification employs writer-specific parameters at multiple levels.
- Exploration of writer-dependency at various levels, including feature level, features dimension, classifier, and cluster level.
- Template signature is selected to represent the training samples of each writer.
- Clustering is done using the template signatures of the writers.
- Exploration of writer-dependent features and adaption of the writer-dependent classifier.
- A lower error rate has been attained in comparison to several other works already in existence.

There are five sections in the paper. The introduction is presented in section I. The various steps involved in the proposed model are explained in section II. The details of the experiment, along with the results, are given in section III. A comparative analysis of the proposed approach is provided in section IV. Conclusions are drawn in section V.

II. PROPOSED MODEL

The distinct stages in the proposed model are:

- Pre-processing
- Feature extraction
- Template signature selection
- Writer-specific feature selection
- Writer-specific classifier selection
- Clustering
- Cluster-specific classifier selection
- Signature verification

The block schematic of the proposed approach is shown in Fig. 1.

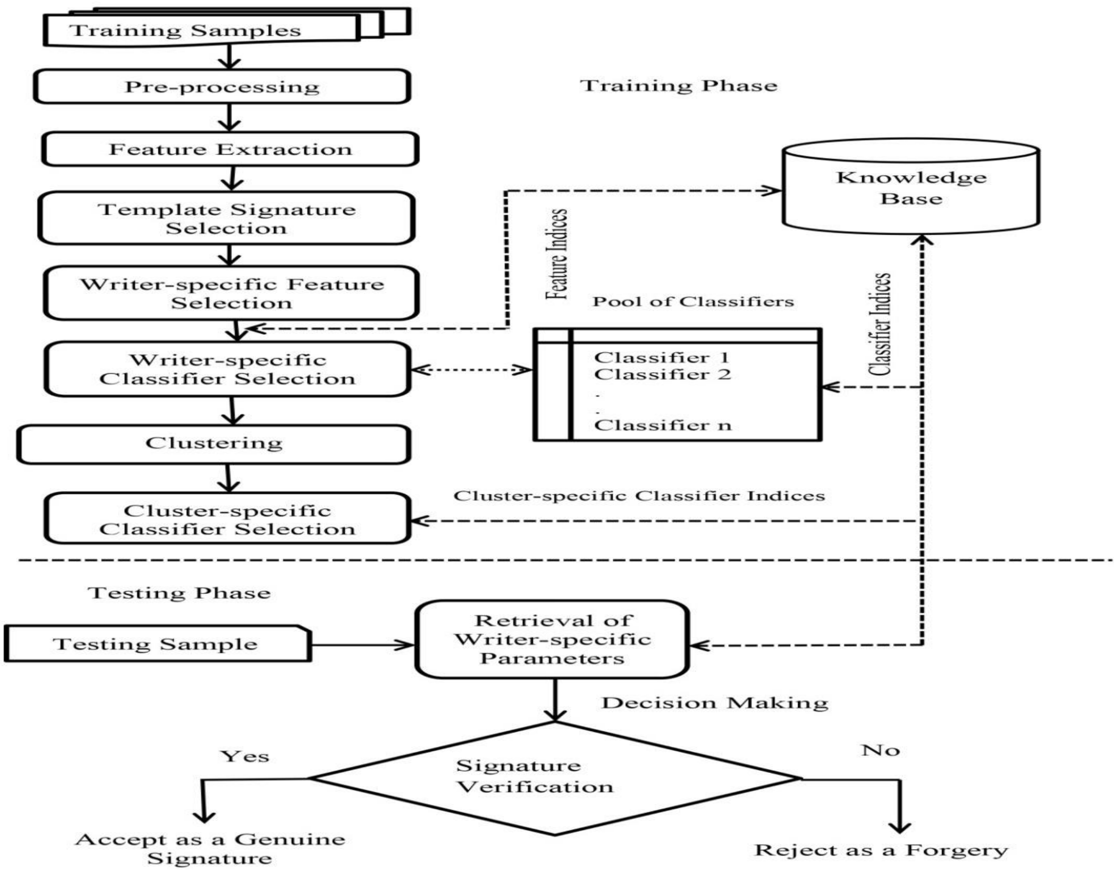

Fig. 1 The block schematic of the proposed writer dependent offline signature verification approach based on cluster-specific classifier

A. *Pre-processing*

In the first step, the original scanned input signature images are pre-processed. Pre-processing operations are typically carried out to enhance the quality of the signature image without losing relevant information. The pre-processing steps used in this work are binarization, filtering, thinning, cropping, and resizing. The threshold technique is used to first convert the signature image into a binary image. The median filter is then used to reduce noise from the binarized image. An image is thinned to reduce its thickness to a single pixel. To make computations simpler, the signature image area has been cropped. The image is then scaled to 256×256. The pre-processing stage is used for both the training and testing signatures. From the pre-processed signature images, various features were subsequently extracted.

B. *Feature Extraction*

In this work, we considered both global and grid features. In the proposed model, global features such as signature area, normalized area, baseline shift, global centroid, etc. are utilized. From each signature sample, 25 global features are computed. To calculate grid features, nine distinct 3×3 masks are slid over the pre-processed image. Five global statistical features are collected from the sample signature when each mask is used: standard deviation, skewness, variance, contrast, and homogeneity. Consequently, a total of 45 grid features are obtained. A total of 70 features are computed from each signature sample. In Table I, the details of the computed features are listed. Let N be the total number of genuine signature samples are collected from each writer during enrolment. For training purposes, m samples are randomly selected from N genuine signature samples.

Let $\omega = \{\omega_1, \omega_2, \omega_3, \ldots, \omega_M\}$, where M is the total number of writers in the system; and $w_k = \{Gs_1^k, Gs_2^k, Gs_3^k, \ldots, Gs_m^k\}$ be the set of m number of genuine signature samples of k^{th} writer $\omega_k (1 \leq k \leq M)$ used for training purpose.

Let $\{F_1^k, F_2^k, F_3^k, \ldots, F_\rho^k\}$ be the feature vector representing the writer w_k where ρ is the total number of features computed for each writer. The feature matrix of each writer ω_k is m× ρ. The feature matrix dimension of the dataset is $M \times \text{m} \times \rho$.

C. *Template signature selection*

In this stage, one signature is selected from the training samples of each writer, which serves as a representative signature sample of the respective writer and is referred to as the template signature. Here, we intend to have effective representation for each writer by providing a single template signature for each class. The selected template signature has the highest degree of resemblance to the other signatures. That is, it has the lowest average distance with other training samples of the same writer.

We used the following steps to select a representative signature for each writer, called a template signature.

- Calculate the pair-wise distance between all the training samples of writer ω_k, that is from Gs_i^k to Gs_j^k where $i = 1,2,3, \ldots m$ and $j = 1,2,3, \ldots m$.

  i.e., Dist $(Gs_i^k, Gs_j^k), i \neq j$
- Compute the average distance for each signature sample $Gs_i^k (i = 1,2,3, \ldots m)$.

$$Avg_{Dist(Gs_i^k)} = \frac{1}{m-1} \sum_{i,j=1}^{m} Dist(Gs_i^k, Gs_j^k), i = 1,2,3, \ldots m, j = 1,2,3, \ldots m.$$

The signature Gs_i^k which has a minimum average distance is selected as the template signature of k^{th} writer denoted by τ_s^k.

i.e., $\tau_s^k = \min(Avg\_Dist(Gs_i^k)), i = 1,2,3, \ldots m$

This results in selecting a template signature for every writer in the database.

TABLE I

DETAILS OF FEATURES EXTRACTED

Feature description	Feature description
Signature area [30]	Baseline shift [46]
Center feature	Variance of row vector [37]
Normalized area [33]	Energy
Minor axis length	Outline signature area [26]
Major axis plus minor axis length	Horizontal center of the signature [47]
Maximum horizontal projection plus maximum vertical projection	Vertical center of the signature [47]
Diameter	Convex area
Radii	Total number of background pixels
Major axis length	Perimeter
Number of edge-points [47]	Equi-diameter
Global centroid	Mean of row vector [37]
Homogeneity [56]	Correlation [56]
Grid Features	

In Table II, the pair-wise distance between randomly selected training samples (when 50% genuine samples are considered for training purpose) of writer1 from the CEDAR dataset, along with their indices, is given as an example.

TABLE II
THE PAIR-WISE DISTANCE BETWEEN ALL THE TRAINING SAMPLES OF WRITER1 FROM THE CEDAR DATASET

Sample #	**21**	**13**	**7**	**24**	**8**	**23**	**9**	**19**	**10**	**14**	**16**	**20**
21	0.00	6.67	5.42	8.50	3.42	3.23	6.80	2.85	6.73	8.19	6.02	8.17
13	6.67	0.00	6.22	9.01	5.28	4.48	3.62	5.25	7.96	5.26	1.84	7.67
7	5.42	6.22	0.00	12.45	7.40	4.94	8.78	3.14	1.84	3.78	4.52	2.84
24	8.50	9.01	12.45	0.00	6.13	9.17	6.15	10.22	14.06	13.62	10.14	15.01
8	3.42	5.28	7.40	6.13	0.00	3.35	3.85	4.60	9.09	8.89	5.60	9.95
23	3.23	4.48	4.94	9.17	3.35	0.00	5.30	2.06	6.58	6.33	3.80	7.14
9	6.80	3.62	8.78	6.15	3.85	5.30	0.00	6.74	10.60	8.70	5.10	10.77
19	2.85	5.25	3.14	10.22	4.60	2.06	6.74	0.00	4.69	5.53	4.01	5.63
10	6.73	7.96	1.84	14.06	9.09	6.58	10.60	4.69	0.00	4.59	6.23	2.25
14	8.19	5.26	3.78	13.62	8.89	6.33	8.70	5.53	4.59	0.00	3.71	3.08
16	6.02	1.84	4.52	10.14	5.60	3.80	5.10	4.01	6.23	3.71	0.00	5.91
20	8.17	7.67	2.84	15.01	9.95	7.14	10.77	5.63	2.25	3.08	5.91	0.00

The average distance between each signature sample and other samples from the same writer is then calculated and given.

Sample #	**21**	**13**	**7**	**24**	**8**	**23**	**9**	**19**	**10**	**14**	**16**	**20**
Average Distance	5.50	5.27	5.11	9.54	5.63	4.70	6.37	4.56	6.22	5.97	4.74	6.53

Here, the signature sample 19 has a minimum average distance and is selected as the template signature of writer1.

Let $\tau^1_{s\,System} = \{\tau_s^1, \tau_s^2, \tau_s^3, \ldots\ldots \tau_s^M\}$ be the set of template signatures selected for all M writers in the system.

For instance, the indices of the sample signatures selected as template signatures of first fifteen writers of CEDAR dataset is given.

τ_s^1	τ_s^2	τ_s^3	τ_s^4	τ_s^5	τ_s^6	τ_s^7	τ_s^8	τ_s^9	τ_s^{10}	τ_s^{11}	τ_s^{12}	τ_s^{13}	τ_s^{14}	τ_s^{15}
19	9	7	11	5	12	1	6	12	1	5	10	5	3	8

Since a single signature is selected as a template signature among each writer's training samples, the feature matrix's dimension will be reduced to M×ρ. After selecting template signatures for each writer, we select writer-specific features for each writer based on their training samples.

D. Writer-specific feature selection

Due to intra-class variations, the signatures of a writer usually generate some natural clusters. Selecting features that can preserve the structure of the cluster represents the signature more effectively. Hence, we use an unsupervised feature selection method suitable for multi-clustered data like offline signatures to select relevant features for each writer called multi-cluster feature selection (MCFS) [13]. For each feature, a relevancy score is determined based on its contribution to preserving the cluster structure of the respective writer. The relevancy score indicates the importance of a feature. We sort the selected features in descending order according to their relevancy score. The features with the highest relevance score are chosen for each writer.

Let $\{F_1^k, F_2^k, F_3^k, \ldots, F_{þ}^k\}$ be the feature vector representing the writer-specific features of writer w_k where þ is the number of relevant features selected out of ρ features. It reduces the dimension of the feature vector of a writer to m× þ(þ $< \rho$) . The features chosen for each writer differ from one writer to another. Writer-specific feature indices are stored in the knowledge base to use during verification. As a result, writer-specific features are selected. As an example, in Table III the feature indices for the top 15 features selected by the MCFS method for the first 20 writers in the CEDAR dataset with 50% training samples.

E. Writer-specific classifier selection

Once the features appropriate for each writer have been determined, the suitable classifier is selected for each writer. Employing a single classifier for all writers may not be appropriate because their signature samples do not fit the same distribution. Hence, we have explored the possibility of adopting a writer-specific classifier for each individual writer.

The appropriate classifier for each writer is selected using the following criteria:

The m training samples are subdivided into two sets for validation: 1. training (m_t) and 2.Validation (m_v). For validation purposes, we have also considered the m_r random number of samples of forgery. Random forgeries are generated for a specific writer by selecting a genuine signature from every other writer.

Let $\mathrm{C} = \{\mathrm{C}_{\mathbf{1}}, \mathrm{C}_{\mathbf{2}}, \mathrm{C}_{\mathbf{3}}, \ldots\ldots, \mathrm{C}_{\propto}\}$, where $\propto$ be a number of classifiers and a writer ω_k is described with a feature vector of dimension $\mathrm{m} \times þ$. Each of the available classifier is trained using the feature matrix $m_t \times þ$ and validated using $m_v \times þ$ and $m_r \times þ$ feature matrices. That is The false acceptance rate (FAR) and false rejection rate (FRR) are computed using m_v and m_r. Finally, an equal error rate (EER) is determined for each classifier. The obtained EER from all the available classifiers C for a writer ω_k can be written as,

$\mathcal{E}_{\mathrm{C}} = \{\mathcal{E}\mathcal{E}\mathrm{R}_1^k, \mathcal{E}\mathcal{E}\mathrm{R}_2^k, \mathcal{E}\mathcal{E}\mathrm{R}_3^k, \ldots\ldots, \mathcal{E}\mathcal{E}\mathrm{R}_\propto^k\}$.

For instance, $\mathcal{E}\mathcal{E}\mathrm{R}_1^k$ denotes the EER obtained from classifier, $\mathcal{E}\mathcal{E}\mathrm{R}_2^k$ denotes the EER obtained from classifier_2 and so on. By varying the training and validation samples, the experiment is repeated T number of trials. The training and validation samples are chosen randomly, with no overlap in each T trail. A classifier with the lowest error rate is identified for each trial.

TABLE III

THE INDICES OF THE TOP FIFTEEN FEATURES SELECTED BY THE MCFS ALGORITHM FOR THE FIRST 20 WRITERS IN THE CEDAR DATASET WITH 50% TRAINING SAMPLES

Write r#	**Indices of the top-ranked fifteen features**														
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	8	46	36	57	2	20	31	14	16	3	53	15	26	40	10
2	8	14	3	57	31	26	21	40	15	46	10	16	53	6	36
3	6	26	14	31	60	57	40	3	16	8	53	46	36	20	10
4	3	40	26	8	10	53	46	20	16	6	31	57	14	36	21
5	2	40	26	46	6	3	14	10	36	57	8	31	15	20	53
6	8	3	26	53	40	20	57	31	2	46	16	6	21	10	15
7	5	57	3	16	40	26	36	11	14	2	6	53	46	60	31
8	60	15	20	31	16	19	14	18	57	40	6	26	36	46	11
9	3	6	15	57	60	14	40	5	20	16	18	21	36	11	46
10	15	20	14	8	3	40	6	16	26	31	10	36	57	53	46
11	15	16	60	8	10	18	2	57	40	14	31	36	3	12	19
12	3	8	6	57	10	53	31	46	26	21	20	14	36	5	16
13	6	10	3	57	46	16	26	15	14	8	36	31	53	5	19
14	8	14	26	20	60	6	40	3	16	46	2	53	36	18	57
15	3	8	10	14	6	15	26	57	19	46	36	21	53	60	20
16	10	6	60	40	14	8	15	26	16	57	53	31	46	3	21
17	3	8	6	26	60	31	20	53	40	57	14	36	15	46	10
18	57	53	3	6	46	40	10	8	31	36	14	26	21	20	16
19	10	57	8	3	46	2	53	20	16	40	31	36	14	21	26
20	3	8	57	60	14	20	53	15	46	26	16	2	19	40	36

For instance, $\mathcal{E}\mathcal{E}\mathrm{R}_1^k$ denotes the EER obtained from classifier, $\mathcal{E}\mathcal{E}\mathrm{R}_2^k$ denotes the EER obtained from classifier_2 and so on. By varying the training and validation samples, the experiment is repeated T number of trials. The training and validation samples are chosen randomly, with no overlap in each T trail. A classifier with the lowest error rate is identified for each trial.

$$\mathrm{C}^{\mathrm{T}}_{Selected} = min\{\mathrm{C}\}$$

Let $\mathrm{C}_{Selected} = \{\mathrm{C}^1_{Selected}, \mathrm{C}^2_{Selected}, \mathrm{C}^3_{Selected}, \ldots\ldots, \mathrm{C}^{\mathrm{T}}_{Selected}\}$ represent the list of classifiers that each of the T trials identified

for the writer ω_k. To select the most suitable classifier from the list, we rank them in descending order based on the total number of times with which they are chosen for a specific writer, as determined by equation 1.

$$Total_{numberoftimes}(\mathrm{C}_{\boldsymbol{k}}, \omega_k) = \frac{Number\ of\ times\ the\ classifier\ \mathrm{C}_k\ selected\ for\ a\ writer\ \omega_k}{Total\ number\ of\ trails\ conducted} \quad (1)$$

The classifier that has been chosen the highest number of times is selected as a writer-specific classifier for the respective writer. In some cases, more than one classifier may be chosen the same number of times for a particular writer. To address such issues, we prioritized the order of the classifiers based on the ease of their implementation, taking into account of the complexity of different classifiers. As a result, whenever there is a tie among the classifiers, the classifier that comes first in the list is selected. Similarly, the classifier is selected for each writer in the database, as discussed above. The classifier chosen in the most trials is given the highest ranking for each writer.

In Table IV, the classifiers ranked by the frequency of selection for the first fifteen writers of the CEDAR dataset are given as examples when the model is trained with 50% training samples. In this work, we have considered six classifiers and the labels C1 to C6 denote the classifiers Naive Bayesian (C1), Nearest Neighbor (C2), Support Vector Machine (C3), Probabilistic Neural Network (C4), Linear Discriminant Analysis (C5), and Principal Component Analysis (C6), respectively.

TABLE IV
CLASSIFIER RANKING FOR THE FIRST 15 WRITERS OF THE CEDAR DATASET

Writer #	Classifier ranking					
1	C_3	C_5	C_1	C_6	C_2	C_4
2	C_3	C_5	C_6	C_1	C_2	C_4
3	C_5	C_3	C_1	C_6	C_2	C_4
4	C_2	C_6	C_3	C_5	C_1	C_4
5	C_5	C_3	C_2	C_6	C_1	C_4
6	C_2	C_6	C_5	C_3	C_1	C_4
7	C_2	C_6	C_3	C_5	C_1	C_4
8	C_3	C_5	C_6	C_2	C_1	C_4
9	C_3	C_5	C_1	C_2	C_6	C_4
10	C_3	C_5	C_1	C_6	C_2	C_4
11	C_3	C_5	C_1	C_6	C_2	C_4
12	C_3	C_5	C_2	C_6	C_4	C_1
13	C_3	C_2	C_6	C_5	C_1	C_4
14	C_3	C_5	C_6	C_1	C_2	C_4
15	C_2	C_6	C_5	C_3	C_1	C_4

F. *Clustering*

Based on the template signatures of each writer in the system, writers are clustered. Clustering based on the template signatures selected has an advantage. Suppose all of the training samples are utilized for clustering, the training samples of the same writer may fall into multiple clusters, making a decision on selecting the classifier for a writer is difficult. We applied the K-means clustering method in this work due to its ease of use and computational efficiency. The value of K is determined empirically.

As a result of clustering, let K number of clusters are formed and denoted as

$C_\gamma = \{C_{\gamma 1}, C_{\gamma 2}, C_{\gamma 3}, \ldots\ldots, C_{\gamma K}\}$, where each cluster $C_{\gamma_p} = \{\tau_s^x, \tau_s^y, \tau_s^z, \ldots\ldots \tau_s^p\}$ be the set of template signatures of different writers say x,y,z..etc. For instance, an example of clustering the writers from the CEDAR dataset is presented in Table V when the number of clusters is equal to 6, and the training set is 50%.

G. *Cluster-specific classifier selection*

Clustering reduces the number of classifiers that need to be trained and also identifies writers with similar characteristics. In this work, we have selected a classifier based on the clustering of writers because it is computationally expensive to create a verification model by training a suitable classifier for each writer.

TABLE V
CLUSTERING THE WRITERS FROM THE CEDAR DATASET

Cluster # 1	**Cluster # 2**	**Cluster # 3**	**Cluster # 4**	**Cluster # 5**	**Cluster # 6**
Writer #					
4	1	6	16	19	2
7	3	26	21	20	9
8	5	38	24	27	10
11	12	46	25	28	34
13	14	48	29	30	41
15	17	49	32	33	47

18	31	50	39	35	51
22		54	44	42	
23			52	43	
36			53		
37			55		
40					
45					

The selection of an appropriate classifier for each cluster is determined as follows:

For instance, the classifier, which has been selected for most of the writers in cluster $C_{\gamma K}$ is C_{l_1} and it is decided as a cluster-specific classifier. Once the cluster classifier is identified, the cluster-specific classifier is set as the default classifier for all writers in the respective cluster. This results in selecting a cluster-specific classifier. Similarly, the classifier is identified for all the other clusters.

After the cluster classifier is identified for all the clusters, we store the indices of the cluster-specific classifiers in the knowledge base. Then, before the verification step, we train the system using writer-specific features and a cluster-specific classifier that is unique to each cluster. In Table VI, an example of the cluster-specific classifiers chosen for writers in each cluster is given along with the writer indices from the CEDAR dataset.

TABLE VI
DETAILS OF CLUSTER-SPECIFIC CLASSIFIERS SELECTED FOR WRITERS IN EACH CLUSTER OF WRITERS FROM THE CEDAR DATASET

Cluster #	**Writer #**	**Cluster-dependent classifier**
1	4,7,8,11,13,15,18,22,23,36,37,40,45	C_3
2	1,3,5,12,14,17,31	C_1
3	6,26,38,46,48,49,50,54	C_2
4	16,21,24,25,29,32,39,44,52,53,55	C_3
5	19,20,27,28,30,33,35,42,43	C_6
6	2,9,10,34,41,47,51	C_2

H. *Signature verification*

Verification is carried out after the writer-specific parameters have been chosen. We retrieve the cluster number and cluster-specific classifier from the knowledge base to verify whether the given test signature is genuine or a forgery. Then the test signature is fed to the respective cluster-specific classifier, which decides the authenticity of the test signature.

Let s_m^k bet the given test signature sample of the writer ω_k. First we retrieve the label of the cluster $C_{\gamma K}$ to which the writer ω_k belongs and the cluster-specific classifier C_{l_1}. To verify the test signature s_m^k is fed into the cluster-specific classifier C_{l_1} which determines the genuinity of the signature.

III.EXPERIMENTATION DETAILS AND RESULTS

In this section, information on the datasets used, the training and testing set details, and the results of the experiment are presented.

A. *Dataset used*

We have conducted experiments on three benchmarking offline signature datasets: CEDAR, MCYT, and GPDS-960 (a sub-corpus of GPDS-4000 Synthetic). Each dataset has a different number of genuine and forged signature samples. For offline signature datasets, skilled forgeries are created by professionals who have practiced and replicated genuine signatures.

The CEDAR [29] dataset consists of 2,640 signature samples collected from 55 writers. There are 48 signature samples from each individual, out of which 24 are genuine, and 24 are skilled forgeries.

The MCYT [41] consists of the signatures of 75 writers. From each writer, 15 genuine and 15 skilled forgeries are collected. Thus, it contains 2,250 samples.

The GPDS-960 is a subset of the GPDS-4000 synthetic dataset [18]. It has 24 genuine signatures and 30 skilled forgeries from 960 writers. Thus, in total, it contains 51,840 signature samples.

B. *Experimental setup*

We have used 70 global features (Table.I) and three offline signature datasets, viz., CEDAR, MCYT, and GPDS-960, for our experiments. Each dataset is split into two sets: training sets and testing sets. The model is trained with varying the percentage of genuine samples from 30% to 70%. Only genuine samples are used for training purposes. The remaining genuine signatures and all the forgery samples (both skilled and random forgeries) are considered for testing.

In this work, for each writer, we have created random forgeries by taking one genuine sample from every other writer. We used a varying percentage of genuine signature samples (70% to 30%) and all the skilled forgeries to test the model with skilled forgeries. Similarly, for testing with random forgeries, we used all the random forgeries with different numbers of genuine signature samples (70% to 30%). Initially, we conducted experiments using a common set of features for all writers as well as a writer-specific classifier. The results achieved on CEDAR, MCYT, and GPDS-960 datasets without feature selection are shown in the last row of Tables VII, VIII, and IX, respectively.

In the second stage, experiments are carried out using writer-specific classifiers under varying feature dimensions from 5 to 50 in step 5. In this work, the MCFS method is used to choose writer-specific features for each writer, as detailed in section 2.4. Once the suitable features for each writer have been identified, the appropriate classifier for each writer is determined. For our experiments, we considered six different classifiers: Naive Bayesian (C1), Nearest Neighbor (C2), Support Vector Machine (C3), Probabilistic Neural Network (C4), Linear Discriminant Analysis (C5), and Principal Component Analysis (C6). The training set is further divided into a training set and a validation set. The k-fold validation technique is used to determine the writer-specific classifier. The value of k is determined empirically. For validation, we considered an equal number of random forgeries and genuine samples. We estimate FAR and FRR for each writer using writer-specific features in each classifier. We calculate EER by taking the average of FAR and FRR.

We carried out the experiments with ten trials. The training samples are selected randomly in each trial. The classifier with the minimum EER is identified in each trial. The decision regarding a writer-specific classifier is arrived at as discussed in section 2.5. As explained in section 2.5, a writer-specific classifier is chosen for each value of the feature dimension. The obtained EER with writer-specific classifiers for different feature dimensions on the CEDAR, MCYT, and GPDS-960 datasets is shown in Tables VII, VIII, and IX, respectively.

TABLE VII
THE EER WITH WRITER-SPECIFIC CLASSIFIER UNDER VARYING FEATURE DIMENSION ON CEDAR DATASET

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
Feature dimension	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
5	12.81	11.23	10.36	8.19	7.64	8.78	7.05	6.58	6.01	5.19
10	12.30	11.23	10.36	8.04	7.25	8.24	7.69	6.45	6.01	5.08
15	12.45	11.21	10.64	8.27	7.43	8.67	7.34	6.03	5.91	4.88
20	12.37	10.76	10.27	8.19	7.60	8.81	7.75	6.70	5.87	4.48
25	11.40	10.45	10.27	8.45	7.53	8.12	7.78	6.52	5.19	4.48
30	11.24	10.68	10.36	8.45	6.62	8.64	7.71	6.30	5.19	3.58
35	11.55	10.30	9.82	7.81	6.17	7.02	7.65	6.52	4.91	3.45
40	10.44	9.92	9.55	7.42	6.30	7.02	6.94	6.21	4.91	3.45
45	10.31	9.24	8.82	7.22	6.36	7.16	6.94	5.91	4.85	3.12
50	10.94	8.45	8.26	7.64	6.62	7.03	6.03	5.09	4.48	3.12
Without Feature Selection	11.05	8.92	8.02	7.89	6.06	7.69	6.91	5.97	4.89	3.73

TABLE VIII
THE EER WITH WRITER-SPECIFIC CLASSIFIER UNDER VARYING FEATURE DIMENSION ON MCYT DATASET

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
Feature dimension	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
5	18.91	17.70	15.61	14.94	13.85	12.28	10.63	9.45	9.36	7.91
10	18.82	17.43	15.44	14.89	13.77	12.19	10.12	9.18	9.32	7.81
15	18.22	16.84	15.53	14.80	13.56	12.08	9.92	8.91	9.01	7.55
20	18.09	16.39	15.39	14.73	13.39	11.64	9.12	8.88	8.72	7.43
25	17.82	16.40	15.32	14.25	12.97	11.64	8.99	8.65	8.52	7.27
30	17.68	16.29	14.18	14.06	12.97	10.73	8.73	8.58	7.97	7.18
35	17.64	15.48	14.33	13.91	12.77	10.45	9.09	7.72	7.95	7.18
40	7.64	15.46	14.11	13.78	12.50	10.36	8.18	7.41	7.79	6.91
45	17.56	15.37	14.11	13.28	12.44	9.91	7.27	6.94	7.54	6.27
50	17.48	15.05	14.01	13.25	12.31	9.27	7.24	6.83	6.67	5.91
Without Feature Selection	18.02	16.33	15.61	14.27	13.12	9.57	8.32	7.45	6.65	5.12

TABLE IX
THE EER WITH WRITER-SPECIFIC CLASSIFIER UNDER VARYING FEATURE DIMENSION ON GPDS-960 DATASET

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
Feature dimension	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
5	20.91	18.53	13.91	12.86	11.41	13.91	12.68	12.62	10.35	9.51
10	20.91	18.14	13.91	12.45	11.37	13.82	12.03	12.35	9.86	9.36
15	20.02	17.88	13.85	11.94	11.37	13.82	12.36	12.14	9.19	9.65
20	20.45	16.74	13.76	10.85	11.29	13.45	11.82	11.20	8.64	9.44
25	20.45	16.02	13.08	10.57	10.78	13.09	11.19	10.91	8.55	9.27
30	19.55	15.55	12.70	10.48	10.05	12.91	10.96	10.00	8.45	8.86
35	19.18	14.09	12.70	9.79	9.14	12.91	10.70	9.32	8.45	8.81
40	19.09	14.09	11.96	9.62	8.63	12.67	10.86	8.71	8.27	8.51
45	19.09	13.96	11.56	9.30	8.78	12.55	10.12	8.94	8.18	7.70
50	20.11	14.27	11.89	10.21	9.33	13.41	11.23	9.21	8.98	7.96
Without Feature Selection	20.91	18.53	13.91	12.86	11.41	13.91	12.68	12.62	10.35	9.51

C. *Results*

To show the efficacy of writer-specific classifier selection, we also carried out experiments using writer-specific features along with a common classifier. The EER obtained on the CEDAR, MCYT, and GPDS-960 datasets when a common classifier is employed for all writers is shown in Tables X, XI, and XII respectively.

TABLE X
THE EER OBTAINED USING WRITER-SPECIFIC FEATURES AND A COMMON CLASSIFIER ACROSS ALL WRITERS IN THE CEDAR DATASET

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
Classifier Label	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
C1	8.51	7.84	6.97	6.75	6.36	5.65	4.97	4.31	3.92	3.23
C2	10.02	9.81	8.45	8.13	7.96	7.21	8.96	6.25	5.63	4.87
C3	6.43	5.72	5.59	6.23	5.61	5.74	5.58	4.67	3.84	3.23
C4	7.96	7.35	6.28	6.16	6.12	5.73	4.87	4.83	3.81	3.55
C5	9.45	9.22	8.08	6.73	6.01	5.51	4.72	4.01	3.63	3.27
C6	10.78	10.02	9.78	7.72	7.28	6.62	6.07	5.74	4.45	3.94

TABLE XI
THE EER OBTAINED USING WRITER-SPECIFIC FEATURES AND A COMMON CLASSIFIER ACROSS ALL WRITERS IN THE MCYT DATASET

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
Classifier Label	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
C1	21.34	19.86	17.64	15.91	11.45	10.8	8.78	7.74	7.12	6.81
C2	19.85	18.78	17.96	15.82	12.86	12.07	9.93	7.43	6.89	7.78
C3	16.37	14.91	12.35	9.66	7.48	8.11	7.06	6.41	5.69	4.77
C4	18.4	17.55	15.2	14.36	12.97	9.08	8.43	7.91	7.83	6.42
C5	20.85	19.36	17.54	15.57	13.51	11.57	9.92	9.77	8.51	7.84
C6	19.34	18.14	16.89	14.27	12.27	10.76	8.91	8.61	7.91	6.95

TABLE XII
THE EER OBTAINED USING WRITER-SPECIFIC FEATURES AND A COMMON CLASSIFIER ACROSS ALL WRITERS IN THE GPDS-960 DATASET

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
Classifier Label	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
C1	17.86	15.82	13.85	12.21	10.51	9.95	8.55	6.59	5.95	5.5
C2	18.82	17.79	15.27	13.94	12.18	10.07	9.93	7.4	6.89	6.38
C3	11.91	11.36	10.82	8.36	7.65	8.82	7.06	5.99	4.97	4.77
C4	16.69	15.91	14.71	12.36	10.21	9.96	7.12	6.27	5.92	4.95
C5	18.79	16.04	15.99	13.51	13.02	9.94	7.92	6.41	5.69	4.87
C6	17.99	17.31	15.48	14.81	12.77	11.77	9.41	7.13	5.56	5.31

The experiment has been carried out by varying the number of clusters from 2 to 10, which is fixed to be the same across all writers. The EER obtained using writer-specific features and cluster-specific classifiers for varying percentages of training samples from 30% to 70% on the CEDAR, MCYT, and GPDS-960 datasets is presented in Tables XIII, XIV, and XV, respectively.

TABLE XIII
THE EER OBTAINED BY VARYING NUMBER OF CLUSTERS ON CEDAR DATASET USING WRITER-SPECIFIC FEATURES AND CLUSTER-SPECIFIC CLASSIFIER

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
No. of clusters	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
2	7.47	7.2	6.95	6.84	6.44	5.74	4.07	3.84	2.89	2.07
3	7.38	7.22	6.83	6.1	6.26	6.45	5.94	4.91	3.45	2.36
4	7.49	7.16	6.42	6.17	5.22	5.55	4.82	3.36	2.91	2.22
5	6.92	6.86	5.63	5.2	4.64	4.82	3.69	3.31	3.02	2.77
6	6.26	6.12	6.05	5.86	5.71	4.12	3.82	3.49	2.69	2.6
7	7.62	7.21	6.72	6.54	6.66	4.99	4.51	4.02	3.94	2.92
8	7.54	7.4	6.97	6.91	6.88	3.79	3.32	3.04	2.58	2.5
9	6.53	6.21	6.05	5.69	5.36	4.82	4.79	3.27	2.95	2.64
10	7.74	6.74	6.07	5.98	5.83	3.71	2.95	2.91	2.63	2.55

TABLE XIV
THE EER OBTAINED BY VARYING NUMBER OF CLUSTERS ON MCYT DATASET USING WRITER-SPECIFIC FEATURES AND CLUSTER-SPECIFIC CLASSIFIER

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
No. of clusters	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
2	16.65	14.54	13.26	12.66	11.21	8.03	6.99	5.69	5.37	4.12
3	16.87	14.41	12.23	11.35	10.44	8.2	6.25	5.47	5.44	4.37
4	16.21	14.33	11.96	9.52	7.12	7.94	6.78	5.47	5.12	4.11
5	15.11	13.62	11.25	8.64	6.45	7.88	5.87	4.98	4.76	3.23
6	16.36	14.47	11.98	8.44	7.12	8.02	6.05	5.01	4.98	3.51
7	16.34	14.32	12.22	8.57	7.23	8.12	6.32	5.36	5.31	4.12
8	16.94	14.81	12.14	8.95	7.14	7.95	6.32	5.25	5.31	4.12
9	16.32	14.12	12.02	8.67	7.13	7.95	6.37	5.36	5.17	4.32
10	16.47	14.36	12.19	8.98	7.11	8.05	6.24	5.15	5.73	4.01

TABLE XV
THE EER OBTAINED BY VARYING NUMBER OF CLUSTERS ON GPDS-960 DATASET USING WRITER-SPECIFIC FEATURES AND CLUSTER-SPECIFIC CLASSIFIER

Percentage of training samples	**30**	**40**	**50**	**60**	**70**	**30**	**40**	**50**	**60**	**70**
No. of clusters	**Testing with Skilled forgeries**					**Testing with Random forgeries**				
2	11.28	9.98	8.22	6.87	6.33	7.42	6.75	4.96	4.72	3.25
3	11.98	9.54	8.23	6.84	6.75	7.69	6.54	4.74	4.51	3.54
4	11.02	9.36	8.02	6.84	6.81	7.58	6.82	4.81	4.21	3.81
5	10.47	9.31	7.96	6.59	6.03	7.02	6.06	4.51	3.98	3.21
6	10.99	9.57	8.09	6.62	6.32	7.12	6.89	4.63	4.02	3.66
7	10.97	9.68	7.99	6.62	6.23	7.21	6.89	4.63	4.25	3.66
8	10.56	9.81	7.99	6.97	6.32	7.21	6.48	4.52	4.25	3.32
9	10.97	9.34	7.84	6.97	6.37	7.54	6.46	4.59	5.17	3.32
10	10.97	9.34	7.89	6.98	6.37	7.54	6.46	4.87	5.73	3.37

It shall be observed that the proposed method performs better when the number of clusters is 5. The proposed technique performs better for random forgeries.

IV. COMPARATIVE STUDY

In this section, the comparative study with state-of-the-art offline signature verification models is presented. Comparing the performance of different offline signature verification models is difficult due to the variations in the number of signature samples considered for training and testing the model. To compare the performance of our model with other models, EER has been used as a measure. In this work, we have compared the EER of the proposed model with other existing models reported in the literature on CEDAR, MCYT, and GPDS datasets. Tables XVI, XVII, and XVII shows the EER of various models tested on the CEDAR, MCYT, and GPDS datasets, respectively. Some table entries are marked with a '-' because the authors have not reported the number of training samples used to obtain those results.

TABLE XVI
COMPARISON OF THE EER/AER OF THE STATE-OF-THE-ART MODELS ON THE CEDAR DATASET

Model	**No. of training samples**	**EER/AER**
[29]	-	21.90
[57]	-	08.50
[24]	04	08.27
[10]	12	7.84
[48]	-	6.75
[35]	10	5.91
[21]	12	5.60
[11]	12	5.50
[20]	-	4.69
[54]	12	4.67
Proposed Method	17	4.64

From Table XVI, it is clear that the error rate we achieved is the lowest when compared to all the models listed.

TABLE XVII
COMPARISON OF THE EER/AER OF THE STATE-OF-THE-ART MODELS ON THE MCYT DATASET

Model	**No. of training samples**	**EER/AER**
[4]	10	22.13
[43]	-	17.61
[24]	10	12.44
[40]	10	9.87
[11]	10	9.26
[56]	10	7.08
[54]	10	5.96
[20]	-	5.58
Proposed Method	09	6.45

In Table XVII, it can be observed that the proposed model performs better than all the existing approaches listed except for two models [20],[54]. This is because the model proposed by [20] is based on a convolutional network, and [54] have computed 156 features from each signature sample whereas, in our work, we have considered 50 top-ranked features and a conventional approach to verify the test signature.

Table XVIII shows that our proposed model achieved the lowest error when compared to all the models listed. The obtained results demonstrated that the proposed model performs better than most state-of-the-art models. This shows the efficiency of writer-specific parameters in offline signature verification systems. Compared to the EER obtained from the CEDAR and GPDS datasets. This is because there are less genuine signatures in the MCYT dataset than in the CEDAR and GPDS datasets that can be used for training.

TABLE XVIII
COMPARISON OF THE EER/AER OF THE STATE-OF-THE-ART MODELS ON THE GPDS-960 DATASET

Model	No. of training samples	EER/AER
[52]	54	18.32
[60]	12	15.41
[15]	-	14.58
[61]	14	12.57
[38]	12	10.42
[34]	-	9.95
[20]	-	6.76
Proposed Method	17	6.03

V. CONCLUSION

In this paper, a writer dependent approach for offline signature verification based on writer-specific parameters has been proposed. In this work, writer-dependency is exploited at three levels: feature, classifier, and cluster. The effectiveness of the model has been tested using six different classifiers that are used in the field of offline signature verification. The validation technique is used to choose writer-specific classifiers. The clustering of writers is accomplished through template signatures, which reduces computational complexity. Verification is done using writer-specific features and cluster-specific classifiers. Extensive experimentation is carried on three benchmark offline signature datasets: CEDAR, MCYT, and GPDS-960. The experimental results of the proposed model based on writer clustering demonstrate the significance of writer clustering.

Conflicts of Interest

The authors have no conflicts of interest to declare.

Funding

No funding is provided for the preparation of manuscripts

REFERENCES

[1] Abdelrahaman, A. A. A., & Abdallah, M. E. A. (2013). K-nearest neighbor classifier for signature verification system. 2013 International Conference on Computing, Electrical and Electronic Engineering (ICCEEE), 58–62.

[2] Al-Maqaleh, B. M., & Musleh, A. M. Q. (2015). An efficient offline signature verification system using local features. International Journal of Computer Applications, 131(10).

[3] Alaei, A., Pal, S., Pal, U., & Blumenstein, M. (2017). An Efficient Signature Verification Method Based on an Interval Symbolic Representation and a Fuzzy Similarity Measure. IEEE Transactions on Information Forensics and Security, 12(10), 2360–2372.

[4] Alonso-Fernandez, F., Fairhurst, M. C., Fierrez, J., & Ortega-Garcia, J. (2007). Automatic measures for predicting performance in off-line signature. 2007 IEEE International Conference on Image Processing, 1, I–369.

[5] Alsuhimat, F. M., & Mohamad, F. S. (2023). Offline signature verification using long short-term memory and histogram orientation gradient. Bulletin of Electrical Engineering and Informatics, 12(1), 283–292.

[6] Annapurna, H., Manjunatha, K. S., & Guru, D. S. (2021). Clustering of Writers and Cluster Dependent Classifier Selection for Offline Signature Verification. Communications in Computer and Information Science, 1376 CCIS, 407–419.

[7] Avola, D., Bigdello, M. J., Cinque, L., Fagioli, A., & Marini, M. R. (2021). R-SigNet: Reduced space writer-independent feature learning for offline writer-dependent signature verification. Pattern Recognition Letters, 150, 189–196.

[8] Baltzakis, H., & Papamarkos, N. (2001). New signature verification technique based on a two-stage neural network classifier. Engineering Applications of Artificial Intelligence, 14(1), 95–103.

[9] Batool, F. E., Attique, M., Sharif, M., Javed, K., Nazir, M., Abbasi, A. A., Iqbal, Z., & Riaz, N. (2020). Offline signature verification system: a novel technique of fusion of GLCM and geometric features using SVM. Multimedia Tools and Applications.

[10] Bharathi, R. K., & Shekar, B. H. (2013). Off-line signature verification based on chain code histogram and Support Vector Machine. Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2013, 2063–2068. https://doi.org/10.1109/ICACCI.2013.6637499

[11] Bhunia, A. K., Alaei, A., & Roy, P. P. (2019). Signature verification approach using fusion of hybrid texture features. Neural Computing and Applications, 31(12), 8737–8748.

[12] Bouamra, W., Djeddi, C., Nini, B., Diaz, M., & Siddiqi, I. (2018). Towards the design of an offline signature verifier based on a small number of genuine samples for training. Expert Systems with Applications, 107, 182–195. https://doi.org/10.1016/j.eswa.2018.04.035

[13] Cai, D., Zhang, C., & He, X. (2010). Unsupervised feature selection for multi-cluster data. Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 333–342.

[14] Coetzer, J., Herbst, B. M., & Du Preez, J. A. (2006). Off-line signature verification: A comparison between human and machine performance. Tenth International Workshop on Frontiers in Handwriting Recognition.

[15] DIaz, M., Ferrer, M. A., Eskander, G. S., & Sabourin, R. (2017). Generation of Duplicated Off-Line Signature Images for Verification Systems. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(5), 951–964.

[16] Diaz, M., Ferrer, M. A., Impedovo, D., Malik, M. I., Pirlo, G., & Plamondon, R. (2019). A perspective analysis of handwritten signature technology. ACM Computing Surveys, 51(6). https://doi.org/10.1145/3274658

[17] Eskander, G. S., Sabourin, R., & Granger, E. (2013). Hybrid writer-independent writer-dependent offline signature verification system. IET Biometrics, 2(4), 169–181.

[18] Ferrer, M. A., Diaz-Cabrera, M., & Morales, A. (2014). Static signature synthesis: A neuromotor inspired approach for biometrics. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(3), 667–680. https://doi.org/10.1109/TPAMI.2014.2343981

[19] Fierrez-Aguilar, J., Alonso-Hermira, N., Moreno-Marquez, G., & Ortega-Garcia, J. (2004). An off-line signature verification system based on fusion of local and global information. Biometric Authentication: ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15th, 2004. Proceedings, 295–306.

[20] Ghosh, R. (2021). A Recurrent Neural Network based deep learning model for offline signature verification and recognition system. Expert Systems with Applications, 168, 114249. https://doi.org/10.1016/j.eswa.2020.114249

[21] Guerbai, Y., Chibani, Y., & Hadjadji, B. (2015). The effective use of the one-class SVM classifier for handwritten signature verification based on writer-independent parameters. Pattern Recognition, 48(1), 103–113. https://doi.org/10.1016/j.patcog.2014.07.016

[22] Hafemann, L. G., Oliveira, L. S., & Sabourin, R. (2018). Fixed-sized representation learning from offline handwritten signatures of different sizes. International Journal on Document Analysis and Recognition, 21(3), 219–232. https://doi.org/10.1007/s10032-018-0301-6

[23] Hafemann, L. G., Sabourin, R., & Oliveira, L. S. (2016). Writer-independent feature learning for Offline Signature Verification using Deep Convolutional Neural Networks. Proceedings of the International Joint Conference on Neural Networks, 2016-Octob, 2576–2583. https://doi.org/10.1109/IJCNN.2016.7727521

[24] Hafemann, L. G., Sabourin, R., & Oliveira, L. S. (2019). Meta-learning for fast classifier adaptation to new users of signature verification systems. IEEE Transactions on Information Forensics and Security, 15, 1735–1745.

[25] Hameed, M. M., Ahmad, R., Mat, L., & Murtaza, G. (2021). Signal Processing : Image Communication Machine learning-based offline signature verification systems : A systematic review. Signal Processing: Image Communication, 93(April 2020), 116139.

[26] Huang, K., & Yan, H. (1997). Off-line signature verification based on geometric feature extraction and neural network classification. Pattern Recognition, 30(1), 9–17.

[27] Impedovo, D., & Pirlo, G. (2008). Automatic signature verification: The state of the art. IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, 38(5), 609–635. https://doi.org/10.1109/TSMCC.2008.923866

[28] Jain, A., Singh, S. K., & Singh, K. P. (2020). Handwritten signature verification using shallow convolutional neural network. Multimedia Tools and Applications, 79(27–28), 19993–20018. https://doi.org/10.1007/s11042-020-08728-6

[29] Kalera, M. K., Srihari, S., & Xu, A. (2004). Offline signature verification and identification using distance statistics. International Journal of Pattern Recognition and Artificial Intelligence, 18(7), 1339–1360. https://doi.org/10.1142/S0218001404003630

[30] Karouni, A., Daya, B., & Bahlak, S. (2011). Offline signature recognition using neural networks approach. Procedia Computer Science, 3, 155–161.

[31] Kaur, H., & Kumar, M. (2021). Signature identification and verification techniques: state-of-the-art work. Journal of Ambient Intelligence and Humanized Computing, 0123456789.

[32] Kaur, R., & Choudhary, P. (2015). Offline signature verification in Punjabi based on SURF features and critical point matching using HMM. International Journal of Computer Applications, 111(16), 4–11.

[33] Kruthi, C., & Shet, D. C. (2014). Offline signature verification using support vector machine. Proceedings - 2014 5th International Conference on Signal and Image Processing, ICSIP 2014, 3–8. https://doi.org/10.1109/ICSIP.2014.5

[34] Li, C., Lin, F., Wang, Z., Yu, G., Yuan, L., & Wang, H. (2019). DeepHSV: User-independent offline signature verification using two-channel CNN. Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 166–171.

[35] Maergner, P., Pondenkandath, V., Alberti, M., Liwicki, M., Riesen, K., Ingold, R., & Fischer, A. (2019). Combining graph edit distance and triplet networks for offline signature verification. Pattern Recognition Letters, 125, 527–533.

[36] Manjunatha, K. S., Annapurna, H., & Guru, D. S. (2019). Offline signature verification: An approach based on user-dependent features and classifiers. In Lecture Notes in Networks and Systems (Vol. 43). Springer Singapore.

[37] Mhatre, P. M., & Maniroja, M. (2011). Offline signature verification based on statistical features. International Conference and Workshop on Emerging Trends in Technology 2011, ICWET 2011 - Conference Proceedings, Icwet, 59–62.

[38] Narwade, P. N., Sawant, R. R., & Bonde, S. V. (2018). Offline signature verification using shape correspondence. International Journal of Biometrics, 10(3), 272–289.

[39] Nguyen, V., Blumenstein, M., & Leedham, G. (2009). Global features for the off-line signature verification problem. Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 1300–1304. https://doi.org/10.1109/ICDAR.2009.123

[40] Ooi, S. Y., Teoh, A. B. J., Pang, Y. H., & Hiew, B. Y. (2016). Image-based handwritten signature verification using hybrid methods of discrete Radon transform, principal component analysis and probabilistic neural network. Applied Soft Computing Journal, 40, 274–282. https://doi.org/10.1016/j.asoc.2015.11.039

[41] Ortega-Garcia, J., Fierrez-Aguilar, J., Simon, D., Gonzalez, J., Faundez-Zanuy, M., Espinosa, V., Satue, A., Hernaez, I., Igarza, J.-J., & Vivaracho, C. (2003). MCYT baseline corpus: a bimodal biometric database. IEE Proceedings-Vision, Image and Signal Processing, 150(6), 395–401.

[42] Pal, S., Alaei, A., Pal, U., & Blumenstein, M. (2015). Interval-valued symbolic representation based method for off-line signature verification. Proceedings of the International Joint Conference on Neural Networks, 2015-Septe.

[43] Prakash, H. N., & Guru, D. S. (2009a). Bi-Interval Valued Symbolic Feature Vector for Off-line Signature Verification: A Fusion Based Approach. IICAI, 1887–1900.

[44] Prakash, H. N., & Guru, D. S. (2009b). Relative orientations of geometric centroids for off-line signature verification. Proceedings of the 7th International Conference on Advances in Pattern Recognition, ICAPR 2009, 201–204. https://doi.org/10.1109/ICAPR.2009.28

[45] Prakash, H. N., & Guru, D. S. (2010). Offline Signature Verification: An Approach Based on Score Level Fusion. International Journal of Computer Applications, 1(18), 55–61.

[46] Qi, Y., & Hunt, B. R. (1994). Signature verification using global and grid features. Pattern Recognition, 27(12), 1621–1629.

[47] Ramachandra, A. C., Rao, J. S., Raja, K. B., Venugopla, K. R., & Patnaik, L. M. (2009). Robust offline signature verification based On global features. 2009 IEEE International Advance Computing Conference, IACC 2009, March, 1173–1178.

[48] Ren, J.-X., Xiong, Y.-J., Zhan, H., & Huang, B. (2023). 2C2S: A two-channel and two-stream transformer based framework for offline signature verification. Engineering Applications of Artificial Intelligence, 118, 105639.

[49] Rivard, D., Granger, E., & Sabourin, R. (2013). Multi-feature extraction and selection in writer-independent off-line signature verification. International Journal on Document Analysis and Recognition, 16(1), 83–103. https://doi.org/10.1007/s10032-011-0180-6

[50] Ruiz-Del-Solar, J., Devia, C., Loncomilla, P., & Concha, F. (2008). Offline signature verification using local interest points and descriptors. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5197 LNCS, 22–29. https://doi.org/10.1007/978-3-540-85920-8_3

[51] Serdouk, Y., Nemmour, H., & Chibani, Y. (2016). New off-line Handwritten Signature Verification method based on Artificial Immune Recognition System. Expert Systems with Applications, 51, 186–194. https://doi.org/10.1016/j.eswa.2016.01.001

[52] Serdouk, Y., Nemmour, H., & Chibani, Y. (2017). Handwritten signature verification using the quad-tree histogram of templates and a support vector-based artificial immune classification. Image and Vision Computing, 66, 26–35.

[53] Shah, A. S., Khan, M. N. A., Subhan, F., Fayaz, M., & Shah, A. (2016). An offline signature verification technique using pixels intensity levels. International Journal of Signal Processing, Image Processing and Pattern Recognition, 9(8), 205–222.
[54] Sharif, M., Khan, M. A., Faisal, M., Yasmin, M., & Fernandes, S. L. (2020). A framework for offline signature verification system: Best features selection approach. Pattern Recognition Letters, 139(February), 50–59. https://doi.org/10.1016/j.patrec.2018.01.021
[55] Singh, S., & Kaur, A. (2014). Off-Line signature verification using sub uniform local binary patterns and support vector machine. International Conference on Chemical Engineering & Advanced Computational Technologies (ICCEACT'2014), Pretoria, South Africa.
[56] Vargas, J. F., Ferrer, M. A., Travieso, C. M., & Alonso, J. B. (2011). Off-line signature verification based on grey level information using texture features. Pattern Recognition, 44(2), 375–385. https://doi.org/10.1016/j.patcog.2010.07.028
[57] Xing, Z.-J., Yin, F., Wu, Y.-C., & Liu, C.-L. (2018). Offline signature verification using convolution Siamese network. Proc.SPIE, 10615, 106151I.
[58] Yapıcı, M. M., Tekerek, A., & Topaloğlu, N. (2021). Deep learning-based data augmentation method and signature verification system for offline handwritten signature. Pattern Analysis and Applications, 24(1), 165–179. https://doi.org/10.1007/s10044-020-00912-6
[59] YIlmaz, M. B., & Yanikoğlu, B. (2016). Score level fusion of classifiers in off-line signature verification. Information Fusion, 32, 109–119.
[60] Yilmaz, M. B., Yanikoglu, B., Tirkaz, C., & Kholmatov, A. (2011). Offline signature verification using classifier combination of HOG and LBP features. 2011 International Joint Conference on Biometrics, IJCB 2011. https://doi.org/10.1109/IJCB.2011.6117473
[61] Zhang, Z., Liu, X., & Cui, Y. (2016). Multi-phase offline signature verification system using deep convolutional generative adversarial networks. 2016 9th International Symposium on Computational Intelligence and Design (ISCID), 2, 103–107.