International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Enhancing Offline Signature Verification through CNN Model Optimization with PSO Algorithm

Abdoulwase M. Obaid Al-Azzani.[1]*, Abdulbaset M. Qaid Musleh[1]

Submitted: 28/01/2024 **Revised**: 06/03/2024 **Accepted**: 14/03/2024

Abstract: The verification of handwritten signatures is integral to numerous applications such as authentication and document verification. The efficacy of an offline signature verification system relies heavily on the feature extraction stage, because it significantly affects the performance of the system. Both the quality and quantity of extracted features play pivotal roles in enabling the system to distinguish between genuine and forged signatures. In this study, we introduce a novel approach aimed at optimizing the hyperparameters of a Convolutional Neural Network (CNN) model for handwritten signature verification by leveraging a Particle Swarm Optimization (PSO) algorithm. The PSO algorithm, inspired by the flocking behavior of birds, is a population-based optimization method. We delineated a search space encompassing various hyperparameter ranges, including the number of convolutional filters, dense layers, dropout rate, and learning rate. Through iterative updates to the positions and velocities of the particles, the PSO algorithm navigates this search space to identify the optimal set of hyperparameters that maximizes the accuracy of the CNN model. Our approach was evaluated across diverse datasets including BHSig260-Bengali, BHSig260-Hindi, GPDS, and CEDAR, each containing a varied assortment of handwritten signature images. The experimental results demonstrate the effectiveness of our proposed method, achieving a remarkable accuracy of 98.3% on the testing dataset.

Keywords: *Offline Signature Verification, Deep Learning, Particle Swarm Optimization Algorithm and Convolutional Neural Network.*

1. Introduction

Handwritten signatures, as a type of behavioural biometric, serve as crucial authentication mechanisms across various industries, including banking, law enforcement, and the workplace[1]. Consequently, the distinction between genuine and forged signatures has become a focal point in biometric systems research. Signature verification systems fall into two primary categories: offline and online systems. Online systems digitally capture signatures in real time, whereas offline systems rely on previously captured signature images. The feature extraction stage in offline systems is particularly vital and significantly affects the system performance. Convolutional Neural Networks (CNNs), a popular deep-learning technique, have achieved notable success in tasks such as handwritten signature recognition [2][3]. However, achieving optimal performance with CNNs often necessitates meticulous tuning of hyperparameters such as filter size, number of filters, dropout rates, and learning rates. Hyperparameter optimization presents a challenge because of variations in the optimal settings depending on the dataset and task. Automated methods for hyperparameter optimization have attracted increasing interest [4]. The Particle Swarm Optimization (PSO) algorithm, inspired by bird swarming behavior, has emerged as a promising strategy [5]. PSO has been demonstrated to be effective in optimizing neural networks, including CNNs (Zhang et al.2022)[7][8][9][10][11][12]. In this study, we propose a novel approach for optimizing CNN hyperparameters for handwritten signature recognition using the PSO algorithm. Our objective is to automatically determine the optimal hyperparameter set that maximizes CNN model accuracy. We defined a search space comprising hyperparameter ranges, including the number of convolutional filters, dense layer count, dropout rates, and learning rates. The PSO algorithm navigates this space iteratively and gradually converges towards the optimal hyperparameter set. The remainder of this paper is organized as follows. Section 2 provides a concise review of previous studies on handwritten signature recognition and hyperparameter optimization. Section 3 outlines the methodology, including the CNN model architecture, PSO algorithm, and the experimental setup. Section 4 presents the experimental findings and performance evaluation. Finally, Section 5 summarizes the findings of the study and discusses potential future research directions.

[1]*Computer Science Department, Faculty of Computer and Information Technology, Sana'a University, Yemen.*
ORCID ID : 0000-0003-1088-1254
[3] *Computer Science., Sana'a University, Yemen.*
* *Corresponding Author Email: ,* aledresi200@yahoo.com

2. Literature Review

Research on offline signature verification (OSV) has yielded various techniques to address its inherent challenges. In this section, we provide an overview of several OSV methods, along with the utilization of CNN architectures and PSO algorithms in related domains. Traditional approaches to OSV often rely on handcrafted features, such as histograms of oriented gradients (HOG), scale-invariant feature transforms (SIFT), and local binary patterns (LBP), coupled with classifiers, such as support vector machines (SVM) and k-nearest neighbors (KNN) [13]. However, these methods may struggle to accurately capture complex spatial patterns and variations in signature images. Researchers have explored innovative approaches to addressing these issues. For instance, in [14], the authors utilized max pooling within a CNN architecture to extract micro-deformations and effectively distinguish between authentic signatures and skilled forgeries.

Similarly, in [15]. In [16], structural and directional features extracted from signatures were used for classification tasks using LSTM and Bidirectional LSTM (BiLSTM) models. Writer-dependent signature verification schemes have also been explored, in which a CNN pretrained for writer identification was employed for the OSV.

CNN Architectures: CNNs have demonstrated exceptional performance in various image classification tasks. Models such as ResNet, VGGNet, and AlexNet have achieved state-of-the-art results on large-scale image datasets such as ImageNet [17]. The application of CNN architectures in OSV is an area of active research, aiming to leverage their capabilities in learning discriminative features from signature images [18]. One such study was conducted in [19]. Focused on a combination of convolutional neural network (CNN) algorithms, including VGG16 and ResNet50, for image embedding. The researchers incorporated triplet loss optimization to further improve the performance of the models. Additionally, they employed machine learning classifiers such as support vector machines, artificial neural networks, random forests, and XGBoost in their evaluation. The experiments were conducted on two datasets: the ICDAR-2011 dataset, which consists of signatures written in Latin (Dutch alphabet) script, and BHSig260-Bengali dataset. In [6], the authors presented a novel online handwritten signature verification algorithm that leveraged a feature dimensionality reduction method known as CAE-MV. The proposed algorithm compresses and selects features from original signature data to construct a set of signature features. To classify and verify the signature feature set, we employed a depth-wise separable Convolutional Neural Network (DWSCNN) based on depth-wise separable convolution. Unlike traditional CNN architectures, DWSCNN effectively reduces the number of neural network parameters, resulting in a significant reduction in running time and resource usage. Remarkably, despite the reduction in parameters, the classification performance of DWSCNN remains comparable to that of CNN. Consequently, our algorithm offers improved efficiency without compromising the accuracy of the online handwritten signature verification tasks.

PSO Algorithm: The PSO algorithm, inspired by collective behaviors observed in nature, has been widely utilized in optimization problems. In image processing, PSO has been employed for tasks such as image segmentation, feature selection, and image reconstruction [20].

Hybrid PSO Approaches: Several hybrid or adaptive solutions that combine PSO with other techniques have been proposed. For example, ALPSO, introduced in [21], optimizes a radial basis function neural network (RBFNN) by incorporating a linearly decreasing inertia weight. Other approaches combine PSO with slide-mode control [22] or adaptive strategies [23][24]to enhance optimization performance.

In conclusion, while traditional methods have been explored for OSV, there is growing interest in leveraging DL techniques, particularly CNN architectures, to capture discriminative features from signature images. Additionally, the potential of optimization algorithms, such as PSO, to enhance the OSV system performance remains largely untapped. This study aimed to bridge this gap by developing a model that combines CNN architectures with the PSO algorithm for OSV.

3. Methodology

In this study, we introduce a model for offline signature verification (OSV) that integrates Convolutional Neural Network (CNN) architectures with a Particle Swarm Optimization (PSO) algorithm. The framework of our proposed model encompasses several essential steps~~,~~ including dataset preparation, preprocessing, feature extraction using CNN architectures, optimization through PSO, and model training and evaluation. Our approach introduces a novel method for optimizing the hyperparameters of a CNN model specifically tailored for handwritten signature recognition by leveraging the PSO algorithm. Inspired by the collective behavior observed in bird flocks, the PSO algorithm operates as a population-based optimization technique.

1.1 Datasets

The datasets utilized in this study played a pivotal role in shaping the experimental framework and facilitating the evaluation of the proposed methodologies. Table 1 provides an overview of the datasets employed, delineating the key attributes essential for understanding their composition and significance.

Table 1. Dataset used by the proposed system.

Dataset	*Signers*	*Genuine*	*Forged*	*Training*	*Testing*
GPDS-300	300	24	30	10,200	6,000
CEDAR	100	24	24	1,540	1,100
Bengali	100	24	30	3,400	2,000
Hindi	160	24	30	5,440	3,200

Signers: This column denotes the number of unique individuals whose signatures contribute to the respective datasets. For instance, the GPDS-300 dataset comprises signatures from 300 unique signers, whereas CEDAR[25] features signatures from 100 individuals. Genuine: Reflecting on the number of genuine signatures per signer, this field underscores the availability of authentic signature samples for analysis. Each signer in the GPDS-300 [26]dataset contributed 24 genuine signatures. Forged: Enumerating the number of forged signatures generated for each genuine signature, this column highlights the synthetic data augmentation process employed to enrich the dataset. For example, in the Bengali dataset, 30 forged signatures were generated per signature. Training: This section quantifies the total number of signature samples allocated for training the models. It encompasses both genuine and forged signatures and provides insights into the scale and complexity of the dataset. For example, the GPDS-300 dataset offers 10,200 signature samples for training purposes. Testing: This field complements the training dataset by representing the aggregate number of signature samples reserved exclusively for evaluating the model performance. In the CEDAR dataset, 1,100 signature samples were earmarked to test the efficacy of the model. Figures 1,2 and 3 show some of the samples used for the datasets.

The comprehensive depiction of dataset attributes facilitates a nuanced understanding of the experimental setup, ensuring the transparency and reproducibility of the research methodology.

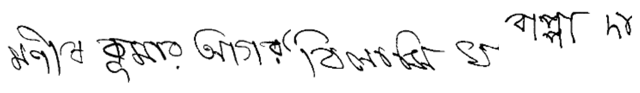


Fig. 1. Sample signatures in the BHSig260-Bengali dataset.

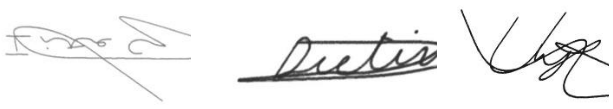


Fig. 2. Sample of GPDS-300 dataset.

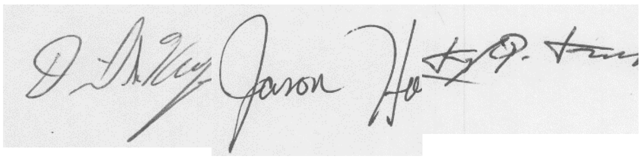


Fig. 3. Sample of the signature CEDAR dataset.

1.2 Preprocessing Techniques

Preprocessing techniques play a crucial role in enhancing the quality and consistency of signature images before feature extraction. These techniques encompass a range of operations aimed at standardizing and refining images for subsequent analysis.

Resizing: A commonly employed preprocessing step involves resizing images to a standard size. This ensured uniformity across the dataset and facilitated the comparison and analysis, such as (100 × 100) pixels.

Grayscale Conversion: Converting images to grayscale is another fundamental pre-processing technique. This simplifies the image representation by removing color information, allowing for more efficient processing and analysis.

By incorporating these preprocessing techniques into the workflow, the quality and consistency of the signature images can be significantly improved, ultimately enhancing the effectiveness of the subsequent feature extraction and analysis processes. Figure 4 illustrates the steps of the proposed model.

1.3 Feature Extraction Using CNN Architectures

CNN model architectures have garnered considerable attention owing to their efficacy in learning discriminative features from image data. In this phase, a pre-trained CNN model such as VGGNet [17] or ResNet [27] serves as the foundation for feature extraction from preprocessed signature images. Typically, a CNN model is modified by discarding the fully connected layers and utilizing the output of the final convolutional layer as the feature representation. These features encapsulate both the global and local patterns inherent in the signatures. The proposed CNN model adopts a sequential approach with incrementally added layers, each designed to contribute to the extraction and representation of the key features. Here, it represents the breakdown of architecture.

(1) **2D Convolutional Layer (Conv2D)**: The model begins with a Conv2D layer that applies A specified number of filters, each 3 × 3, to the input data. Employing a Rectified Linear Unit (ReLU) activation function introduces nonlinearity, allowing the model to effectively discern intricate patterns and features.

(2) **Max Pooling Layers (MaxPooling2D)**: Following each convolutional layer, MaxPooling2D layers are introduced to reduce the spatial dimensions of the feature maps. These layers achieve this reduction by selecting the maximum value within each pooling region, typically of size 2x2.

(3) **Flattening**: The output of the final convolutional layer is flattened using the flattening function, which transforms the 2D feature maps into a 1D vector.

(4) **Dense (Fully Connected) Layers**: Two dense layers were added post-flattening by employing ReLU activation functions. The number of neurons in these layers can be customized according to the requirements of the model.

(5) **Final Dense Layer**: The number of neurons in the final dense layer was aligned with the total number of classes in the dataset. Utilizing the "softmax" activation function transforms the output of the model into a probability distribution across classes. This facilitates multiclass classification by predicting class probabilities for each input signature image. The equations provided in subsequent sections comprehensively outline the CNN model architecture and elucidate the structural components and their respective functionalities within the model.

$$Co\ layer\ \boldsymbol{A}_i = ReLU(\boldsymbol{W}i * \boldsymbol{X}i - 1 + \boldsymbol{b}_i) \quad (1)$$

$$Max - pooling\ layer\ Y_i = MaxPooling(\boldsymbol{A}_i) \quad (2)$$

$$Flatten\ \ F_i = \ Flatten\ (\boldsymbol{Y}_i) \quad (3)$$

$$Dropout\ \ D_i = Dropout(\boldsymbol{F}_i) \quad (4)$$

$$Dense\ H_i = ReLU(\boldsymbol{W}i \cdot \boldsymbol{D}i - 1 + \boldsymbol{b}_i) \quad (5)$$

$$Output\ layer\ O = Softmax(W \cdot \boldsymbol{H}_n + \boldsymbol{b}) \quad (6)$$

where:

$\mathbf{X}i$ represents the input to the layer i.

$\mathbf{A}_i$ where represents the activation of layer i.

Y_i where represents the output of the max-pooling layer.

F_i where denotes the flattened output.

D_i represents the output after dropout is applied.

H_i represents the activation of the dense layer.

O represents the output prediction.

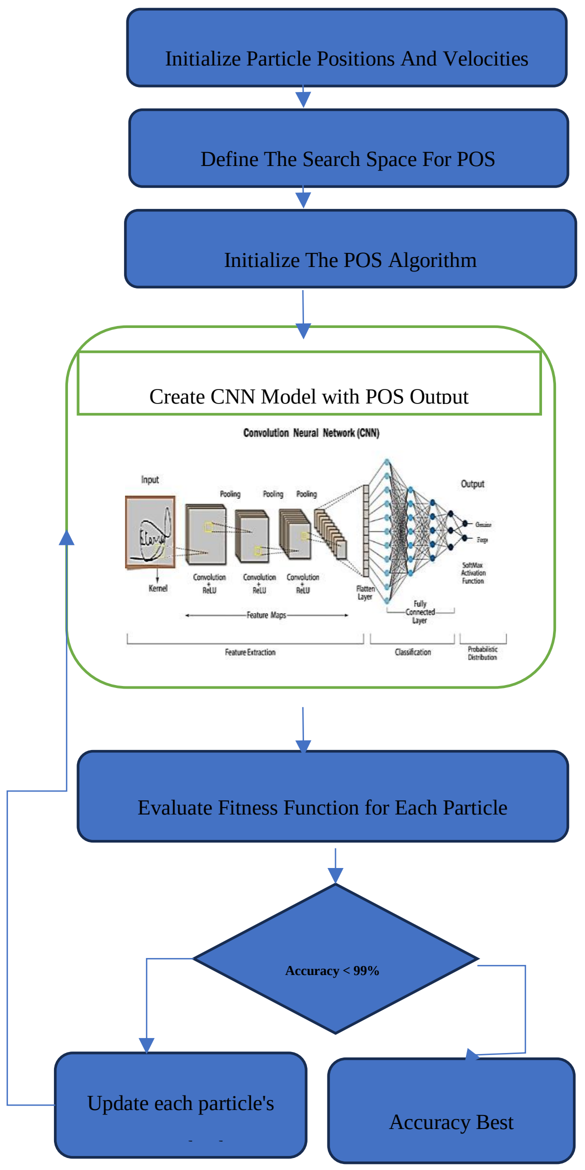

Initialize Particle Positions And Velocities
Define The Search Space For POS
Initialize The POS Algorithm
Create CNN Model with POS Output
Convolution Neural Network (CNN)
Input
Pooling
Pooling
Pooling
Output
Kernel
Convolution + ReLU
Convolution + ReLU
Convolution + ReLU
Flatten Layer
Fully Connected Layer
SoftMax Activation Function
Feature Maps
Feature Extraction
Classification
Probabilistic Distribution
Evaluate Fitness Function for Each Particle
Accuracy < 99%
Update each particle's
Accuracy Best

Fig. 4. Proposed model architectures.

1.4 Optimization Using PSO Algorithm

The PSO algorithm is integrated into the proposed model to enhance its efficiency. Our primary aim was to identify the optimal set of parameters to maximize the accuracy of an offline signature verification (OSV) system. The PSO algorithm facilitates this optimization process by systematically exploring the parameter space. At its core, the PSO algorithm iteratively updates a swarm of particles, with each particle representing a potential solution within the parameter space. These updates are guided by the particle's experience and the best global solution found thus far. By continuously adapting and refining their positions, the particles collectively converged toward the optimal solution.

The fitness function employed in the PSO algorithm is crucial for guiding the optimization process. In our case, the fitness function was directly related to the accuracy of the CNN model in the validation set. This ensures that the optimization process prioritizes solutions that result in higher model accuracy.

The particle positions within the parameter space correspond to the specific values of the various model parameters. These parameters may include the number of filters, learning rate, regularization parameters, and even the number of elements in flattened layers in the CNN architecture. By exploring different combinations of these parameters, the PSO algorithm effectively searches for the configuration that yields the highest accuracy for OSV.

The proposed method implements a Particle Swarm Optimization (PSO) algorithm to optimize the parameters of a Convolutional Neural Network (CNN) model for offline signature verification. The PSO algorithm aims to maximize the accuracy of the CNN model on a validation dataset by exploring a predefined range of hyperparameters

Table 2. Hyperparameters used in the proposed system.

Hyperparameter	***Range***
Filters Number	Choice (16, 128)
Unit	Choice (128, 1024)
Dropout	Choice (0.25, 0.50)
Learning Rate	Choice (0.00001, 0.001)

Table 2 represents the range of values for each parameter the spectrum of options considered during the optimization process. For instance, the "Filter Size" parameter ranges from 16 to 128, indicating that the PSO algorithm will explore values within this range to determine the optimal filter size for the CNN model.

The fitness function utilized in the PSO algorithm incorporated the accuracy of the CNN model in the validation set. It is defined as:

$$Fitness = -Accuracy$$

where represents the classification accuracy of the CNN model on the validation set. The PSO algorithm iteratively updates the positions and velocities of particles according to the following equation: The PSO algorithm ensures that the particle positions remain within the specified search space by enforcing bounds on parameter values. Upon completion of the PSO iterations, the algorithm outputs the best discovered parameters, along with the corresponding accuracy achieved by the CNN model. The updated positions were bounded within the search space. Equation (8,9) show the update method for updating the equation for both the velocity of particle i and location of particle i.

$$L_i(t+1) = \boldsymbol{L}_i(t) + \boldsymbol{V}_i(t+1) \qquad (7)$$

$$\boldsymbol{V_i(t+1) = \omega \cdot V_i(t) + c_1 \cdot R_1 \cdot (P_i(t) - L_i(t)) + c_2 \cdot R_2 \cdot (P_g(t) - L_i(t))} \qquad (8)$$

where:

$\mathbf{L}_i$ where denotes the location of particle i at time t.

V_i where denotes the velocity of particle i at time t.

ω is the inertia weight.

(c_2) and (c_1) are social and cognitive weights, respectively.

$\mathbf{R}_1$ and $\mathbf{R}_2$ are random vectors.

$\mathbf{P}_i$ denotes the best personal location of particle i at time t.

$\mathbf{P}_g$ denotes the global best location among all the particles at time t.

4. Experiments Results

At this stage of our research methodology, the array of parameter outputs generated by the Particle Swarm Optimization (PSO) algorithm is passed onto the Convolutional Neural Network (CNN) model. Each array encapsulates a set of values that represent the crucial parameters essential for configuring the CNN architecture. These parameters included the number of filters for the four layers in the CNN model, size of the kernel for each layer, number of neurons in the fully connected layers, and learning rate. Upon receiving these parameters, the CNN model processed them along with the training image array. Subsequently, the efficiency of the model was evaluated using a separate test dataset to ascertain its accuracy, which served as a metric for quantifying the effectiveness of the specific parameter configuration generated by the PSO algorithm. This accuracy value is then relayed to the PSO algorithm to serve as its fitness value, thereby guiding further iterations. Because the PSO algorithm completes the designated number of generations, the best accuracy is selected, preserving the corresponding parameter weights within the CNN. Through these iterative steps, our proposed model achieved a remarkable accuracy of 98%, indicating its robust performance in offline signature-verification tasks. The CNN model architecture comprises four layers: the input layer, Conv1, Conv2, Conv3, and Conv4. These layers, combined with the optimized parameters, form the backbone of our system for signature image processing and the verification of genuine handwritten signatures. Table 3 presents a comprehensive list of the parameters utilized in our CNN model tailored to the specific requirements of each dataset employed in our study. These parameters serve as critical components for configuring the CNN architecture to achieve optimal performance. To provide a comprehensive analysis, we compare the results obtained using our proposed model with those reported in previous studies. Figures 5, 6, 7, and 8 depict graphical representations of the performance curves, showing the variations in loss, validation loss, validation accuracy, and overall accuracy across different datasets. Furthermore, Tables 4, 5, 6, and 7 present a comparative analysis of the results obtained from our proposed model with those reported in other studies, providing insights into the efficacy and competitiveness of our approach in offline signature verification tasks.

Table 3. Parameter configuration of the proposed system.

Parameter	*GPDS-300*	*CEDAR*	*BHSig260-B*	*BHSig260-H*
Learning	0.00001	0.0001	0.0001	14
Parameters	4,24,56,243	24,24,840	1,60,84,440	12,91,458
Layer 1	Conv2D(85, 3x3)	(16, 3x3)	(16, 3x3)	(64, 3x3)
Layer 2 &MaxPool(2, 2)	Conv2D(85, 3x3)	Conv2D (16,3x3)	Conv2D(16, 3x3)	Conv2D (64,3x3)
Layer 3 &MaxPool(2, 2)	Conv2D(80, 3x3)	Conv2D (16,3x3)	Conv2D(128,3x3)	Conv2D (64,3x3)
Layer 4 &MaxPool(2, 2)	Conv2D(80, 3x3)	Conv2D (16,3x3)	Conv2D(128,3x)	Conv2D (46,3x3)
Dropout	0.5	0.5	0.25	0.5
Flatten	1024	1024	1024	512
Accuracy	93.01	98.3	97.22	98

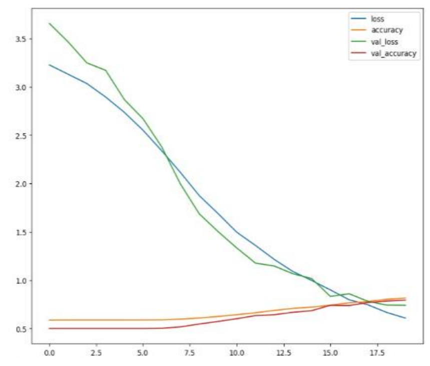
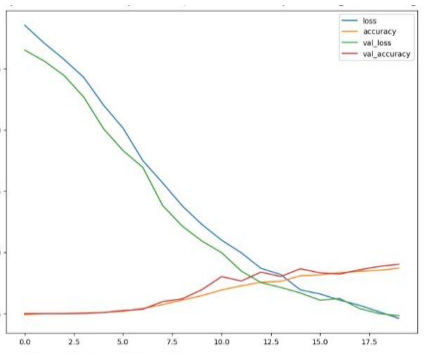


Fig. 5. The CEDAR Dataset.



Z

Fig. 6. The GPDS-300 datasetx

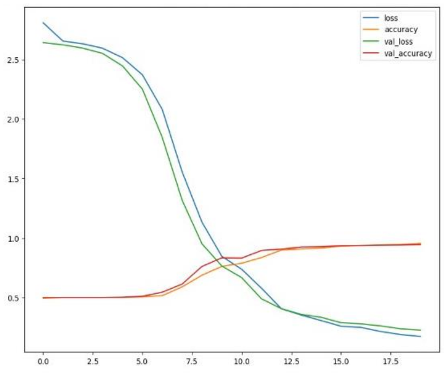


Fig. 7. The BHSig260-H Dataset.

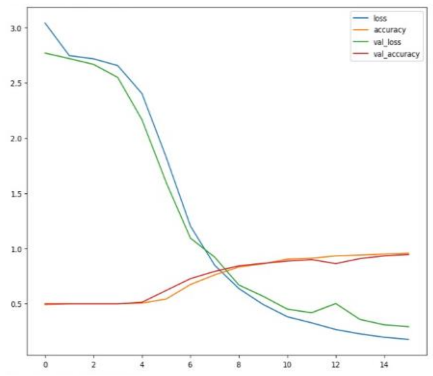


Fig. 8. The BHSig260-B Dataset.

In the table 4 provides a comparison of different methods used on the BHSig260-Hindi datasets, along with their corresponding accuracy, false acceptance rate (FAR), false rejection rate (FRR), and equal error rate (EER).

LBP & ULBP[28]: This method achieved an accuracy of 75.53% with a FAR and FRR of 24.47%. The EER, which represents the point where the FAR and FRR are equal, is also 24.47%.
SigNet[29]: SigNet achieved an accuracy of 84.64% with a FAR and FRR of 15.36%. The EER is also 15.36%.
CNN & BiLSTM [16]: This method achieved a significantly higher accuracy of 97.80%. The FAR is 2.0% and the FRR is 2.44%. The EER is 2.23%.
Proposed model: The proposed model outperformed the other methods with an accuracy of 98.00%. It achieved a FAR and FRR of 1.91% and 1.89%, respectively. The EER is 1.88%.

Table 4. Comparison results for BHSig260-Hindi datasets.

Methods	Accuracy	FAR	FRR	EER
[28]	75.53	24.47	24.47	24.47
[29]	84.64	15.36	15.36	15.36
[16]	97.80	2.0	2.44	2.23
Proposed model	98.00	1.91	1.89	1.88

In the table 5 presents a comparison of different methods applied to the BHSig260-Bengali dataset, along with their respective accuracy, false acceptance rate (FAR), false rejection rate (FRR), and equal error rate (EER).
LBP & ULBP[28]: This method achieved an accuracy of 66.18% with a FAR and FRR of 33.82%. The EER, which represents the point where the FAR and FRR are equal, is also 33.82%.
SigNet[29] achieved an accuracy of 86.11% with a FAR and FRR of 13.89%. The EER is also 13.89%.
CNN & BiLSTM [16] This method achieved an accuracy of 91%. The FAR is 1.36% and the FRR is 2.13%. The EER is 1.76%.
Proposed model: The proposed model outperformed the other methods with an accuracy of 97.22%. It achieved a FAR and FRR of 1.2% and 1.8%, respectively. The EER is 1.75%.
In summary, the proposed model demonstrated the highest accuracy and the lowest FAR, FRR, and EER values among the compared methods, indicating its superior performance on the BHSig260-Bengali dataset.

Table 5. Comparison results for the BHSig260-Bengali dataset.

Methods	*Accuracy*	*FAR*	*FRR*	*EER*
[28]	66.18	33.82	33.82	33.82
[29]	86.11	13.89	13.89	13.89
[16]	91	1.36	2.13	1.76
Proposed model	97.22	1.2	1.8	1.75

Based on the data presented in Table 6. presents a comparison of the performance of different methods with our method on the CEDAR dataset in terms of FAR, FRR, EER, and accuracy. "Our method" outperforms the other methods in terms of FAR, FRR, EER, and accuracy, indicating its superior performance on the CEDAR dataset. LBP and ULBP[28] achieved a False Acceptance Rate (FAR) of 8.33%, False Rejection Rate (FRR) of 8.33%, Equal Error Rate (EER) of 8.33%, and accuracy of 91.67%. On the other hand, the "Multi-Path Siamese (MA-SCN)" method [6] yields an FRR of 18.35%, FAR of 19.21%, EER of 18.92%, and accuracy of 80.75%. Additionally, the "Siamese CNN" method [30] achieves a FAR of 6.78%, FRR of 4.20%, and accuracy of 95.66%. In comparison, our proposed method outperformed these approaches with an FAR of 1.34%, FRR of 1.78%, EER of 1.94%, and accuracy of 98.3%.

Table 6. Comparison results for the CEDAR datasets.

Method	*FAR*	*FRR*	*EER*	*Accuracy*
[31]	8.33	8.33	8.33	91.67
[6]	19.21	18.35	18.92	80.75
[30]	6.78	4.20	–	95.66
Our method	1.34	1.78	1.94	98.3

Table 7 compares the performance of different methods on the GPDS300 datasets. The methods evaluated are Compact correlated features, SigNet[29], CNN & BiLSTM[16], and the Proposed model.
The table provides the accuracy, false acceptance rate (FAR), false rejection rate (FRR), and equal error rate (EER) for each method.
Compact correlated features[32]: This method achieved an accuracy of 88.79%. The FAR, FRR, and EER were all 11.21%.
SigNet[29]: SigNet achieved an accuracy of 76.83%. The FAR, FRR, and EER were all 23.17%.
CNN & BiLSTM[16]: This method achieved an accuracy of 89.88%. The FAR was 9.37%, the FRR was 11.05%, and the EER was 10.16%.
Proposed model: The proposed model achieved the highest accuracy of 93.01%. It had a FAR of 8.14%, an FRR of 9.43%, and an EER of 7.67%.
In summary, the Proposed model demonstrated the highest accuracy and the lowest FAR, FRR, and EER values among the compared methods, indicating its superior performance on the GPDS300 datasets.

Table 7. Comparison results for the GPDS300 datasets.

Methods	*Accuracy*	*FAR*	*FRR*	*EER*
[32]	88.79	11.21	11.21	11.21
[29]	76.83	23.17	23.17	23.17
[16]	89.88	9.37	11.05	10.16
Proposed model	93.01	8.14	9.43	7.67

5. Conclusion

This study introduces a novel method that utilizes the Particle Swarm Optimization (PSO) algorithm to optimize the hyperparameters of Convolutional Neural Network (CNN) models for handwritten signature recognition. The primary aim is to identify the most effective combination of hyperparameters that maximizes the accuracy of the CNN model in recognizing handwritten signatures. Inspired by the social behavior observed in flocking birds, the PSO algorithm was employed to explore a predefined search space comprising hyperparameter ranges. The hyperparameters under consideration included the number of convolutional filters, dense layers, dropout rate, and learning rate. Through iterative updates to the velocities and particle positions, the PSO algorithm identified the optimal hyperparameter combination that yielded the highest accuracy. To evaluate the proposed approach, multiple datasets containing diverse sets of handwritten signature images were utilized, including the BHSig260-Bengali, BHSig260-Hindiin, GPDS, and CEDAR datasets. The experimental results demonstrated the efficacy of the PSO algorithm in optimizing the hyperparameters of the CNN model. Notably, the proposed method achieved an impressive accuracy of 98% on the testing dataset, surpassing the performance of the existing approaches. These findings underscore the potential of leveraging metaheuristic algorithms, such as PSO, to enhance the performance of Deep Learning (DL) models in signature recognition tasks. By optimizing the hyperparameters, the proposed approach can significantly improve the accuracy of handwritten signature recognition systems, thereby offering considerable benefits for various applications including authentication and document verification.

Future research avenues could explore the application of alternative metaheuristic algorithms or combination strategies to further enhance the CNN model performance in signature recognition tasks. In addition, investigating the scalability and generalizability of the proposed approach across larger and more diverse datasets would provide valuable insight into its practical applicability and robustness.

References

[1] K. Jain, K. Nandakumar, and A. Ross, "50 years of biometric research: Accomplishments, challenges, and opportunities," Pattern Recognit Lett, vol. 79, pp. 80–105, Aug. 2016, doi: 10.1016/j.patrec.2015.12.013.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[3] Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks." [Online]. Available: http://code.google.com/p/cuda-convnet/

[4] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential Model-Based Optimization for General Algorithm Configuration," 2011, pp. 507–523. doi: 10.1007/978-3-642-25566-3_40.

[5] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), IEEE, 1998, pp. 69–73. doi: 10.1109/ICEC.1998.699146.

[6] X. Zhang, Z. Wu, L. Xie, Y. Li, F. Li, and J. Zhang, "Multi-Path Siamese Convolution Network for Offline Handwritten Signature Verification," in ACM International Conference Proceeding Series, Association for Computing Machinery, Jan. 2022, pp. 51–58. doi: 10.1145/3512850.3512854.

[7] F. C. Soon, H. Y. Khaw, J. H. Chuah, and J. Kanesan, "Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition," IET Intelligent Transport Systems, vol. 12, no. 8, pp. 939–946, Oct. 2018, doi: 10.1049/iet-its.2018.5127.

[8] Wang, Y. Sun, B. Xue, and M. Zhang, "Evolving Deep Convolutional Neural Networks by Variable-Length Particle Swarm Optimization for Image Classification," in 2018 IEEE Congress on Evolutionary Computation (CEC), IEEE, Jul. 2018, pp. 1–8. doi: 10.1109/CEC.2018.8477735.

[9] S. S. Talathi, "Hyper-parameter optimization of deep convolutional networks for object recognition," in 2015 IEEE International Conference on Image Processing (ICIP), IEEE, Sep. 2015, pp. 3982–3986. doi: 10.1109/ICIP.2015.7351553.

[10] S. Y. S. Leung, Y. Tang, and W. K. Wong, "A hybrid particle swarm optimization and its application in neural networks," Expert Syst Appl, vol. 39, no. 1, pp. 395–405, Jan. 2012, doi: 10.1016/j.eswa.2011.07.028.

[11] E. Camci, D. R. Kripalani, L. Ma, E. Kayacan, and M. A. Khanesar, "An aerial robot for rice farm quality inspection with type-2 fuzzy neural networks tuned by particle swarm optimization-sliding mode control hybrid algorithm," Swarm Evol Comput, vol. 41, pp. 1–8, Aug. 2018, doi: 10.1016/j.swevo.2017.10.003.

[12] J. Raitoharju, S. Kiranyaz, and M. Gabbouj, "Training Radial Basis Function Neural Networks for Classification via Class-Specific Clustering," IEEE Trans Neural Netw Learn Syst, vol. 27, no. 12, pp. 2458–2471, Dec. 2016, doi: 10.1109/TNNLS.2015.2497286.

[13] R. Plamondon and S. N. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," IEEE Trans Pattern Anal Mach Intell, vol. 22, no. 1, pp. 63–84, 2000, doi: 10.1109/34.824821.

[14] Tsourounis, I. Theodorakopoulos, E. N. Zois, and G. Economou, "From text to signatures: Knowledge transfer for efficient deep feature learning in offline signature verification," Expert Syst Appl, vol. 189, p. 116136, Mar. 2022, doi: 10.1016/j.eswa.2021.116136.

[15] J. Zheng et al., "DWSCNN Online Signature Verification Algorithm Based on CAE-MV Feature Dimensionality Reduction," IEEE Access, vol. 12, pp. 22144–22157, 2024, doi: 10.1109/ACCESS.2024.3355449.

[16] T. Longjam, D. R. Kisku, and P. Gupta, "Writer independent handwritten signature verification on multi-scripted signatures using hybrid CNN-BiLSTM: A novel approach," Expert Syst Appl, vol. 214, p. 119111, Mar. 2023, doi: 10.1016/j.eswa.2022.119111.

[17] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Sep. 2014, [Online]. Available: http://arxiv.org/abs/1409.1556

[18] S. Masoudnia, O. Mersa, B. N. Araabi, A.-H. Vahabie, M. Amin Sadeghi, and M. N. Ahmadabadi, "Multi-Representational Learning for Offline Signature Verification using Multi-Loss Snapshot Ensemble of CNNs."

[19] Christianto, J. Colin, and G. P. Kusuma, “International Journal of Computing and Digital Systems Authentic Signature Verification Using Deep Learning Embedding With Triplet Loss Optimization And Machine Learning Classification.” [Online]. Available: http://journals.uob.edu.bh

[20] T. Sinha, A. Haidar, and B. Verma, “Particle Swarm Optimization Based Approach for Finding Optimal Values of Convolutional Neural Network Parameters,” in 2018 IEEE Congress on Evolutionary Computation (CEC), IEEE, Jul. 2018, pp. 1–6. doi: 10.1109/CEC.2018.8477728.

[21] H.-G. Han, W. Lu, Y. Hou, and J.-F. Qiao, “An Adaptive-PSO-Based Self-Organizing RBF Neural Network,” IEEE Trans Neural Netw Learn Syst, vol. 29, no. 1, pp. 104–117, Jan. 2018, doi: 10.1109/TNNLS.2016.2616413.

[22] J.-F. Qiao, C. Lu, and W.-J. Li, “Design of Dynamic Modular Neural Network Based on Adaptive Particle Swarm Optimization Algorithm,” IEEE Access, vol. 6, pp. 10850–10857, 2018, doi: 10.1109/ACCESS.2018.2803084.

[23] M. A. Ferrer, J. F. Vargas, A. Morales, and A. Ordonez, “Robustness of Offline Signature Verification Based on Gray Level Features,” IEEE Transactions on Information Forensics and Security, vol. 7, no. 3, pp. 966–977, Jun. 2012, doi: 10.1109/TIFS.2012.2190281.

[24] Y. Wang, H. Zhang, and G. Zhang, “cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks,” Swarm Evol Comput, vol. 49, pp. 114–123, Sep. 2019, doi: 10.1016/j.swevo.2019.06.002.

[25] S. S. and A. X. MEENAKSHI K. KALERA, “OFFLINE SIGNATURE VERIFICATION AND IDENTIFICATION USING DISTANCE STATISTICS,” Intern J Pattern Recognit Artif Intell, vol. 18, no. 07, p. . 1339-1360, 2004.

[26] M. A. Ferrer, M. Diaz-Cabrera, and A. Morales, “Static Signature Synthesis: A Neuromotor Inspired Approach for Biometrics,” IEEE Trans Pattern Anal Mach Intell, vol. 37, no. 3, pp. 667–680, Mar. 2015, doi: 10.1109/TPAMI.2014.2343981.

[27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition.” [Online]. Available: http://image-net.org/challenges/LSVRC/2015/

[28] S. Pal, A. Alaei, U. Pal, and M. Blumenstein, “Performance of an Off-Line Signature Verification Method Based on Texture Features on a Large Indic-Script Signature Dataset,” in 2016 12th IAPR Workshop on Document Analysis Systems (DAS), IEEE, Apr. 2016, pp. 72–77. doi: 10.1109/DAS.2016.48.

[29] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Llados, and U. Pal, “SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification,” Jul. 2017, [Online]. Available: http://arxiv.org/abs/1707.02131

[30] W. Xiao and Y. Ding, “A Two-Stage Siamese Network Model for Offline Handwritten Signature Verification,” Symmetry (Basel), vol. 14, no. 6, Jun. 2022, doi: 10.3390/sym14061216.

[31] R. Kumar, J. D. Sharma, and B. Chanda, “Writer-independent off-line signature verification using surroundedness feature,” Pattern Recognit Lett, vol. 33, no. 3, pp. 301–308, Feb. 2012, doi: 10.1016/j.patrec.2011.10.009.

[32] Dutta, U. Pal, and J. Llados, “Compact correlated features for writer independent signature verification,” in 2016 23rd International Conference on Pattern Recognition (ICPR), IEEE, Dec. 2016, pp. 3422–3427. doi: 10.1109/ICPR.2016.7900163.