International Journal of Theoretical and Applied Research (IJTAR)
ISSN: 2812-5878

Homepage: https://ijtar.journals.ekb.eg

Original article

Offline signature verification using deep learning method

Nehal Hamdy Al-banhawy [a], Heba Mohsen [b], Neveen I. Ghali [c], Ayman Khedr [d] *

[a] *Mathematics and Computer Science Department, Faculty of Science, Al-Azhar University, Cairo, Egypt.*
[b] *Lecturer, Computer Science Department, Faculty of Computers and Information Technology, Future University in Egypt, New Cairo, Egypt.*
[c] *Head of Digital Media Technology Department, Faculty of Computers and Information Technology, Future University in Egypt, New Cairo, Egypt.*
[d] *Professor of Information Systems, Faculty of Computers and Artificial Intelligence, Helwan University, Cairo, Egypt*.

ARTICLE INFO

Received 01/06/2023
Revised 12/11/2023
Accepted 14/11/2023

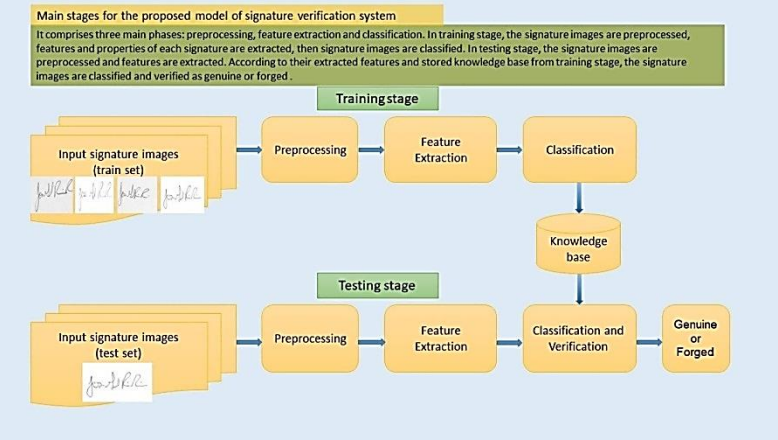
Keywords

Signature analysis
Signature verification
Deep learning
Convolutional Neural Network

ABSTRACT

One of the most challenging biometric authentication problems in recent years that we experience in daily life is signature verification. Signature verification systems are classified into two main approaches: offline systems and online systems. The offline signature verification systems are more difficult than the online systems since online systems have further information, such as the velocity of writing, motion style, and pen pressure, which allow for extracting more features. This paper presents a deep learning method based on using the convolutional neural network (CNN) model for solving the offline signature verification problem to prevent the process of faking signatures that thieves practice. The CNN model was applied for extracting features and classifying whether the signature is genuine or forged. Our proposed method succeeded in achieving an accuracy of 94.73% on the CEDAR dataset by using two types of signatures: genuine signatures and skilled forged signatures to test the performance of the system, which indicates that the method was effective and it can be supported by more feature extractors to get better results.

Graphical abstract



* *Corresponding author*
E-mail address: nehalhamdy2017@gmail.com

DOI: **10.21608/IJTAR.2023.205346.1051**

Special issue "selected papers from the International Annual Conference on Basic and Applied Science (IACBAS-2023)"

1. Introduction

One of the most commonly used techniques to determine the identity of individuals that we experience in our daily lives is handwritten signature analysis, particularly in banking systems, financial applications, administrative applications, and security. Individuals can be authenticated using two main types of characteristics: physiological and behavioral. The physiological characteristics include face, iris, fingerprint, and DNA. While behavioral characteristics include handwritten signatures and gait, a signature is still the most reliable way for authentication in many financial transactions because its acquisition process is simple and it requires less effort. However, this study is considered one of the most difficult challenges because the signatures of the same person may be affected by the person's situation at different times and are not necessarily similar [1, 2].

A handwritten signature verification system is used to verify the authenticity of individuals automatically by their signatures. Therefore, the performance of this system is measured by its ability to classify the genuine signatures and the forged signatures correctly. Forgery is categorized according to its similarity to the genuine signature into three main types: random forgery, simple forgery (unskilled), and skilled forgery. In the case of random forgery, the forger does not have any information about the signatory's name or the shape of his signature and he may use his or her own signature. In such a case, the forger's signature shape is entirely different from the original signature. In the case of simple forgery, the forger knows only the signatory's name with no information about the shape of his signature. In such a case, the forger's signature shape might be similar to the signatory's original signature. In a skilled forgery case, the forger knows both the signatory's name and signature shape with some practice to make a good faking of the signatory's signature. It is difficult to verify this kind of forgery because of its high similarity to the signatory's signature [3, 4]. Figure 1 illustrates some samples of genuine signatures for some writers and how the skilled forged signatures for them are similar to the genuine signatures.

The systems of signature verification are divided into two main approaches: offline/static systems and online /dynamic systems. As for offline systems, a signature is scanned or captured from a document such as a passport or a bank check as an image, and then the system reads that image of the signature and extracts features from it. On the other hand, the online systems use devices such as smartphones, tablets, and electronic pads to capture more information such as velocity of writing, motion style, and pen pressure, while the user is entering his signature, which enables the system to extract more features [3, 5].

The signature verification system has two basic scenarios when dealing with users: writer (user) independent (WI) and writer (user) dependent (WD). In writer-independent, one model is trained for all writers and extracts a similarity/dissimilarity vector then uses it for comparing the questioned signature with reference signatures. Most researchers prefer this method because it does not require retraining the system when a new writer is added, but it needs large enough data to perform well. In writer-dependent, there is one specific model for each writer that is trained on his signatures and is responsible for verifying them. When adding a new writer to the system, the system needs to be trained on it [6].

Every handwritten signature verification system includes three primary stages, preprocessing, feature extraction, and classification. Signature features represent properties that can be detected from the entire line path to describe every signature as a vector of values, which allows to discriminate between the signatures of different writers [5]. Feature extractors can be classified as handcrafted feature extractors or learning feature representations from row data. Handcrafted feature extractors are widely used for signature verification [2, 7, 8], but it need human experts, extensive domain understanding, and more time when dealing with huge datasets. With the breakthrough in artificial intelligence and the appearance of deep learning techniques, a revolution has occurred in the performance of automatic feature extraction systems because of their ability to learn more meaningful features from data by themselves effectively and handle a large and complex dataset in less time [9]. Signature verification systems that are based on deep learning methods have achieved great advancement in recent years, especially the convolutional neural network (CNN) [10]. CNNs are currently supposed to be the most extensively used deep learning architectures for learning features from images because of their various successes in different fields, including pattern recognition and image classification [11].

The proposed models of CNN differ according to the number of convolutional layers, filters, kernel size, max-pooling layers, and fully connected layers. Some researchers used simple models of CNN to solve the signature verification problem, but either they did not achieve enough success as in [12] or they achieved successful performance by using a large number of samples for training (more than 750 samples for each signer) as in [13]. Some researchers resorted to using common CNN models with a large number of layers, as in [14] that used two deep models of CNN (one with 22 layers and the other model with 42 layers) and obtained noticeable performance but required more computations and more time. Some researchers used pre-trained models of CNN instead of building a new deep CNN model to make it capable of extracting a high feature representation of the images in addition to learning the significant features from the signature images in the new dataset, as in [10] and [15].

For the classification stage, classifiers like Euclidean distance [16], Neural Network (NN) [2, 7, 12, 13 and 17], Support Vector Machine (SVM) [7, 8,

11, 18-21] and K- nearest neighbor (K-NN) [11 and 19] are widely used in signature verification problem, but NN and SVM are widely used as a result of their superb performance compared to the rest.

Since the use of signature verification systems is essential in many organizations, such as banks, it requires speed as well as accuracy and secrecy. Although deep learning proved its success in signature verification problem solving, deeper models need essential computational resources like a powerful GPU and large memory that may be highly expensive and time-consuming. Therefore, the objective of our paper is to present a simple and effective offline signature verification system based on using important preprocessing steps for signature images and a simple CNN model for feature extraction and classification that can verify genuine and forged signatures automatically in less time, even though the forgery is highly skilled.

The rest of the paper is ordered as follows: related works, the used dataset, the preprocessing steps of it, and the proposed models of CNN are presented in Section 2. The results and discussion are illustrated in Section 3. Finally, the conclusion is explained in Section 4.

2. Materials and Methods

The offline handwritten signature verification system includes three primary stages: preprocessing, feature extraction, and classification. The researchers proposed different techniques to perform feature extraction and classification stages; we will summarize some of them in the following subsection.

2.1 Related works

Over the last years, many researchers have proposed several systems to solve the signature verification problem due to its importance and difficulty. Some of the handcrafted feature extraction methods that are used in signature verification systems will be outlined first, and then learning features models will be presented.

Kumar et al. [7] proposed a WI offline signature verification schema by extracting a set of features that is based on using the surroundedness property of a signature, which represents the shape and texture attributes of the signature, by using two classifiers: support vector machine and multilayer perceptron, on two datasets: GPDS300 and CEDAR datasets. The best result was achieved by using a multilayer perceptron, with 86.24% accuracy for the GPDS300 dataset and 91.67% accuracy for the CEDAR dataset.

Bhunia et al. [8] proposed a WD signature verification method that used discrete wavelet transform (DWT) and local quantized patterns (LQP) for extracting texture features from signature images. They used one-class support vector machines (SVMs) for classification by considering only genuine signatures for training. Four offline datasets were used to test the system: GPDS-300, MCYT, BHSig-260, and CEDAR datasets that achieved Average Error Rate (AER) results as follows: 4.18, 6.10, 10.91, and 1.64, respectively.

Jain et al. [2] proposed a signature verification methodology based on extracting geometrical features like center, connected components, and isolated points from each signature image. An artificial neural network (ANN) was used for classification and was tested on MCYT-100, MCYT-75, GPDS-4000, BHSig260, and CVBLSig datasets. This model was trained on genuine signatures only and all datasets were tested on random forgeries with accuracy of 97.36%, 97.33%, 92.32%, 97.79%, 95.29%, 97.55%, and 83.38% for MCYT-100, MCYT-75, GPDS300, BHSig Bengali, BHSig Hindi, CVBLSig-V1 and CVBLSig-V2, respectively. Some datasets were tested on skilled forgeries too which achieved an accuracy of about 79.32%, 83.2%, 76.03%, and 83.5 for MCYT-100, GPDS300, BHSig Bengali, and Hindi, respectively.

With the improvement in artificial intelligence and the appearance of deep learning techniques, the performance of automatic feature extraction systems has also improved. Signature verification systems that use deep learning have achieved great advancement recently [10]. Yapıcı et al. [12] proposed an offline system using a convolutional neural network (CNN). The CNN was used to perform feature extraction and classification separately for the WD scenario and WI scenario. The obtained results displayed that the WI scenario implemented 62.5% accuracy and the WD scenario implemented 75% accuracy when using the GPDS synthetic signature dataset that contains signatures of 4000 different writers.

Gideon et al. [13] used CNN to extract features and classification for images of the dataset, which consists of 6000 signatures for 3 users, each of whom has 1000 genuine and 1000 forged signatures. They separated the forged signatures from the genuine signatures for every user and considered each of them as a separate user resulting a dataset with 6 users. By using an 80-20 data split ratio, they achieved a training accuracy of 98.11% and a validation accuracy of 98.23%.

Jahandad et al. [14] used two different models of deep convolutional neural network architectures called Inception-v1 (deep network with 22 layers) and Inception-v3 (with 42 layers deep) for extracting features and classifying genuine and skilled forged signatures. Signatures of 1000 users of the common GPDS dataset were used to evaluate the system. The best results were obtained by the Inception-v1 model when using only 20 users from the dataset. The results are 83% validation accuracy and 17% Equal Error Rate (EER) for Inception-v1, 75% validation accuracy and 24% EER.

Vohra et al. [18] discussed two models for solving the signature verification problem. The first model extracted the following features: histogram of gradient, aspect ratio, shape, contour area, bounding area, and

convex hull area, then used SVM as a classifier. The second model used CNN for feature extraction and classification. ICDAR Dutch dataset was used here, which has 320 signatures taken from 4 individuals plus 80 signature images taken from four individuals. The SVM model obtained 86.39% accuracy and the CNN model obtained 83.78%.

Hung et al. [15] presented an offline signature system that is based on using Siamese Triplet CNN for extracting features and a fully connected neural network for binary classification to verify the genuine and forged signatures automatically. The used model of CNN was the Xception model (with 71 layers), which is a pre-trained model on the ImageNet dataset, and then it was trained on both genuine and forged signatures of the Bengali dataset from BHSig260. The obtained results are 27.97 False Rejection Rate (FRR), 13.66 False Acceptance Rate (FAR), and 14.18 EER with the use of random forgeries.

Some researchers used a hybrid between CNN and another feature extractor for feature extraction. Alsuhimat et al. [11] presented a hybrid method that used CNN and Histogram of Oriented Gradients (HOG) for feature extraction in signature verification. The system merged the significant features from the HOG method and the CNN method together. Then it tested the extracted features using three classifiers: long short-term memory (LSTM), SVM, and K-Nearest Neighbor (KNN) on two datasets; CEDAR and UTSig. After evaluating the system, it obtained 93.7% accuracy by using LSTM, 94.1% by using SVM, 91.3% by using KNN with CEDAR dataset, 95.4% by using LSTM, 95.2% by using SVM, and 92.7% by using KNN with UTSig dataset.

It is obvious that there is a vast interest in using convolutional neural networks for solving signature verification problem, whether it is in feature extraction only or in feature extraction and classification.

2.2 Methodology

2.2.1 Dataset

The CEDAR dataset was used for evaluating the performance of the system because of its common use by researchers and ease of access.

The CEDAR dataset contains 55 writers; each of them has 24 genuine signatures and 24 skilled forged signatures. Therefore, the dataset has entirely 1320 genuine signatures and 1320 skilled forged signatures in a grayscale PNG format [11]. Figure 1 illustrates some samples of genuine signatures for some writers from the CEDAR dataset and their opposite skilled forged signatures for those writers.

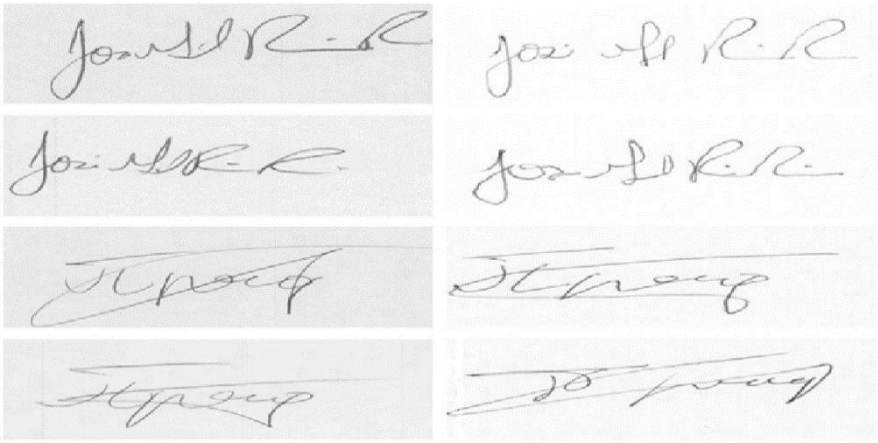


Fig. 1. Some samples from the CEDAR dataset. On the right, the genuine signatures of two writers (two signatures for each one) and on the left, the skilled forged signatures of that writers.

2.2.2 Preprocessing of Dataset

Data preprocessing is a very important phase in a handwritten signature verification system because it improves the signature image, removes unimportant influences, and reduces the computational time therefore it affects the accuracy. The preprocessing steps are ordered as follows:

Binarization and background elimination

The Otsu thresholding method is used here to convert the grayscale images to binary. Besides, the background is removed by converting it to black (has zero pixel value) and the signature to white (has 255 pixel value).

Noise removal

The median filter is used to remove the noise from the images for enhancing them, whereas for every pixel in the input image, it produces an output pixel that contains the median value of 3 x 3 neighborhoods around it.

Thinning

A binary image was converted to thin by applying a specific morphological operation to it.

Cropping

The bounding box of the signature was detected and the other black pixels were cut out of the images, leaving only the binarized signature.

Resizing

All images in the dataset were entered into the model with a fixed size of 128 x 128.

The result of each step in preprocessing is illustrated in Figure 2.

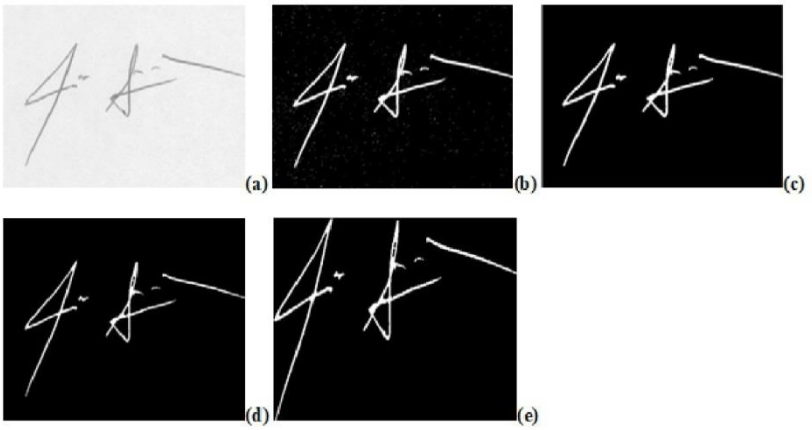


Fig. 2. Results of preprocessing. (a) Original image in grayscale, (b) after binarization and background elimination, (c) after using median filter, (d) after thinning, and (e) after cropping.

2.2.3 Feature Extraction and Classification

Our models are based on writer-dependent (WD) scenario, so there is a classifier for verifying every writer in the system. The use of CNN for extracting features from the training dataset is focused on here. A CNN model consists of three main layers: convolutional layers, pooling layers, and fully connected layers.

The convolutional layer operates the convolution filters on the input images to produce the feature maps. These feature maps that are created by the convolution filters are processed through the activation function before yielding the output. When going deeper into convolutional layers to the last one, we find that the produced feature maps describe parts of shapes and objects in the image. The pooling layer decreases the dimension of the image where it merges neighboring pixels of a specific area in the image into a single representative value [22]. The fully connected layer comes after several convolutional and pooling layers; it may be one or more that aim to perform high-level reasoning. It takes all the neurons in the previous layer and connects them to all the neurons in the current layer to collect all information. The feature extraction operation is done by convolutional and pooling layers, while the classification operation is done by fully connected layers with a softmax activation function for multiclass classification or a sigmoid activation function for binary classification [23].

In this paper, three different models that have CNN as a feature extractor are applied, and the results of them are compared. The first model is the same model as in ref [12], but it is applied here on another dataset (CEDAR dataset). The other two models are proposed here and based on the first model, with some differences. The description of each of the three models is illustrated as follows:

1- CNN model [12]

The first model that is used here is built on the model in [12], but it is applied to another dataset. The CNN technique is used for feature extraction and verification because of its wide success in solving several image classification problems. The CNN model in [12] has five convolutional layers, two max-pooling layers, three fully connected layers, and two dropout layers. Every convolutional layer is supported by a zero-padding layer. Each convolutional layer and fully connected layer (except the last fully connected layer) are attached by the Rectified Linear Unit (ReLU) activation function. The first and second fully connected layers are followed by two dropout layers with the 0.5 parameter. The last fully connected layer is supported by a softmax activation function with two outputs. The input layer, with a size of 128 x 128, is connected to the first convolutional layer, which has 32 filters; thus it generates 32 feature maps. The feature maps are fed to the Rectified Linear Unit (ReLU) activation function before yielding the output, then the output of the activation function is fed to the next layer, and so on. All convolutional layers have a kernel size of 3 x 3. Only the second and fifth convolutional layers are followed by max-pooling layers that have a size of 3 x 3 with a stride size of 2 x 2. Table 1 illustrates the details of the layers in this model of CNN. Because we have

two outputs, the categorical cross-entropy loss function is used to evaluate how good the model is; a lower loss means better performance. The training of the model was performed using the Adam optimizer with a 10^{-4} learning rate and a number of epochs = 60.

Table 1. The architecture of CNN of the first model [12].

Layer	Size	Other Parameters
Input	128 × 128 × 3	
Convolution(C1)+ ReLU	32 × 3 × 3	Stride = 1 , padding = 1
Convolution(C2)+ ReLU	64 × 3 × 3	Stride = 1 , padding = 1
MaxPooling	64 × 3 × 3	Stride = 2
Convolution(C3)+ ReLU	128 × 3 × 3	Stride = 1 , padding = 1
Convolution(C4)+ ReLU	64 × 3 × 3	Stride = 1 , padding = 1
Convolution(C5)+ ReLU	128 × 3 × 3	Stride = 1 , padding = 1
MaxPooling	128 × 3 × 3	Stride = 2
Fully Connected(F1)+ ReLU+ Dropout	256	
Fully Connected(F2)+ ReLU+ Dropout	256	
Fully Connected(F3) + Softmax	2	

2- Proposed Model 1 (CNN & SVM)

The first proposed model is built on using the same CNN as before (without the last fully connected layer) for feature extraction and SVM for classification because of its effective performance in classification problems, as reported in [8, 11, 18-21]. The feature vector that is produced from CNN is fed to the SVM classifier to verify the genuine and forged signatures. The SVM is a significant and simple classifier. Support Vector Machine has two types: linear SVM and non-linear SVM. Linear SVM finds the best hyperplane that separates classes linearly. Non-linear SVM uses kernels to transform the nonlinear data into a higher dimensional space that can be separated linearly. This model applied a two-class classification of SVM (with a linear kernel) where there are two outputs (genuine, forged) then a squared hinge loss function was used here. The Adam optimizer with a 10^{-4} learning rate is used for training the model with a number of epochs = 60.

3- Proposed Model 2 (CNN with Sigmoid)

The second proposed model is built on using CNN for feature extraction and verification. This CNN model has five convolutional layers, three max-pooling layers, three fully connected layers and two dropout layers. It differs from the first CNN model by the following:

1. Increasing the kernel size of the first convolutional layer to 11 x 11.
2. Adding one more max-pooling layer after the fourth convolutional layer.
3. The last fully connected layer is supported by a sigmoid activation function with one output; therefore, a binary cross-entropy loss function was used.

Each convolutional layer and fully connected layer, except the last fully connected layer, are attached by the ReLU activation function. The first and second fully connected layers are followed by two dropout layers with the 0.5 parameter. The training of the model was performed using the Adam optimizer with a 10^{-4} learning rate and 5-fold cross-validation with a number of epochs = 40 for each fold. Table 2 displays the details of layers in this model of CNN.

Table 2. The architecture of improved CNN of the third model.

Layer	Size	Other Parameters
Input	128 × 128 × 3	
Convolution(C1)+ ReLU	**32 × 11 × 11**	**Stride = 1 , padding = 1**
Convolution(C2)+ ReLU	64 × 3 × 3	Stride = 1 , padding = 1
MaxPooling	64 × 3 × 3	Stride = 2
Convolution(C3)+ ReLU	128 × 3 × 3	Stride = 1 , padding = 1
Convolution(C4)+ ReLU	64 × 3 × 3	Stride = 1 , padding = 1
MaxPooling	**64 × 3 × 3**	**Stride = 2**
Convolution(C5)+ ReLU	128 × 3 × 3	Stride = 1 , padding = 1
MaxPooling	128 × 3 × 3	Stride = 2
Fully Connected(F1)+ ReLU+ Dropout	256	
Fully Connected(F2)+ ReLU+ Dropout	256	
Fully Connected(F3) + sigmoid	**1**	

3. Results and Discussion

For measuring the performance of the system, the accuracy is calculated. It is defined as the ratio of the number of correctly verified samples to the total number of test samples, as in Equation 1. In general, higher accuracy indicates better performance.

$$verificatin\ accuracy = \frac{number\ of\ correctly\ verified\ samples}{total\ number\ of\ test\ samples} \times 100 \quad (1)$$

The first model was trained on 22 samples for each writer by taking 20 samples for testing, distributed between two classes (10 genuine, 10 forged), before training and the rest of 28 samples were used for training and validation with 4:1 ratio. The number of epochs was set to 60. The Adam optimizer with a 10^{-4} learning rate was used for training after many attempts to find the appropriate learning rate. This model achieved 93.18% accuracy, which is considered an impressive result despite the simplicity of the model in addition to the difficulty of identifying skillful forgery.

The second model (proposed model 1) gave a worse performance than the first model by using the same ratio for training and testing as before, which may refer to the fact that the CNN works better with a large number of samples for training, that is, 11 genuine and 11 forged signatures for each writer.

The third model (proposed model 2) achieved better performance by increasing the kernel size of the first convolutional layer to 11 x 11 to recognize more information from the input image and adding one more max-pooling layer after the fourth convolutional layer. The number of samples for testing was reduced to 8 distributed between two classes (genuine, forged) before training, and the rest of 40 samples were used for training and validation using 5-fold cross-validation. The number of epochs was set to 40 for each fold and the training was completed after 200 epochs. As we are dealing with two classes here (genuine and forged), a sigmoid activation function was used in the last fully connected layer to make a binary classification with one output. The sigmoid function gives an output value between 0 and 1, which means that the signature is genuine if the output is bigger than 0.5, and it is forged if the output is less than 0.5. The obtained results for the three models on the CEDAR dataset are given in Table 3.

After training and testing the model, it is clear that the preprocessing steps are very important for enhancing the performance of the system. Therefore, if the researcher in reference [12] had applied the important steps of preprocessing on the GPDS synthetic signature dataset before training the model, he would have achieved more than 75% success.

Increasing the number of samples for training to 32 did not improve the system much because the number of samples was large in the beginning. The system did not need a large number of epochs because the training fit early, as illustrated in Figure 3. Therefore, the number of epochs was reduced to 40 for each fold.

Table 3. The obtained results on the CEDAR dataset for the three models.

System	Feature extraction	classifier	Number of samples per writer	Accuracy
CNN model [12]	CNN	CNN	11 genuine, 11 forged	93.18%
Proposed Model 1 (CNN & SVM)	CNN	SVM	11 genuine, 11 forged	92.77%
Proposed Model 2 (CNN with Sigmoid)	CNN	CNN	16 genuine, 16 forged	94.73 %

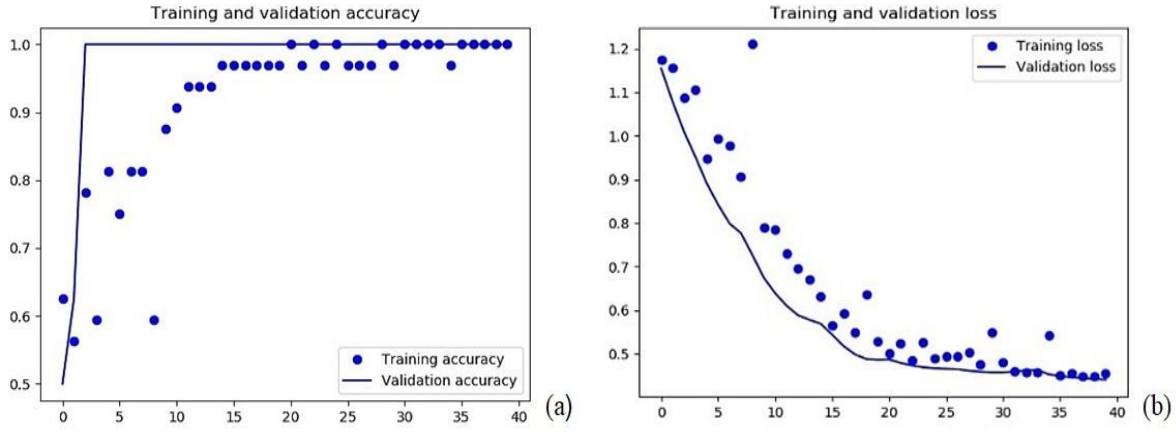


Fig. 3. (a) Graph for training and validation accuracy, (b) Graph for training and validation loss.

In CNN, not only the number of convolutional layers and the number of filters in each of them are important, but also the kernel size in each convolutional layer and the number of max-pooling layers. Increasing the kernel size of the first convolutional layer to 11 x 11, enhances the performance in less number of epochs, but it takes more time for each epochs, so we added one more max-pooling layer after the fourth convolutional layer. The CNN achieved comparable performance in feature extraction, but it needs more samples for training. The number of samples is very substantial and highly influential, where the higher number of samples gives better performance of the system, but it has to be large enough to show its effectiveness to enhance the performance of the system.

It is obvious that we can get good performance from a simple model of CNN supported by enhanced images that have been preprocessed by the main steps of preprocessing. The deep CNN models, as in references [14] and [15], are not essential for effective performance. Jahandad et al. [14] achieved better accuracy (83%) on the GPDS dataset by using the model with 22 layers and not the model with 42 layers. Hung et al. [15] obtained a good performance on the Bengali dataset by using a model of 71 layers, but both [14] and [15] need more computations, whereas more layers mean more complexity and more computational time. We achieved 94.73% accuracy on the CEDAR dataset by using a model composed of 13 layers. Gideon et al. [13] acquired an impressive performance (98.23% validation accuracy) on their own dataset by using a simple CNN model that contains 8 layers, but they had a large number of samples of each writer (1000 genuine and 1000 forged) and used 0.8 of them for training.

It is noted that using both genuine and forged signatures for training improves the performance of the system. Jain et al. [2] trained their system on genuine signatures only and achieved perfect performance when testing the system on random forgeries, but it was not good enough when it was tested on skilled forgeries (79.32%, 83.2%, 76.03%, and 83.5 for MCYT-100, GPDS300, BHSig Bengali and Hindi, respectively) by using an ANN classifier.

When comparing our method with other methods that used the same dataset, our method obtained better performance than Kumar et al. [7], which achieved 91.67% accuracy by using a multilayer perceptron classifier, and Alsuhimat et al. [11], which performed 94.1% accuracy by using an additional feature extractor with CNN for feature extraction and SVM for classification. However, Alsuhimat et al. [11] performed better when matching with our second model that used SVM for classification, because they used an additional feature extractor with CNN. Bhunia et al. [8] attained an outstanding performance (1.64 AER), but they used another metric for measuring the performance of the system that does not allow us to compare with them.

4. Conclusion

The offline handwritten signature verification is one of the hardest challenges in recent years. The deep learning method based on CNN architecture has been achieved wide success in the image classification field, so it was applied here to solve signature verification problem. The CNN model was trained as writer-dependent for extracting features and classifying the queried signatures as genuine or forged. The obtained results indicated that CNN succeeded in extracting features from signatures and gave better performance than SVM in classification. The preprocessing step is very important and enhances the performance of the system because it removes noise and external effects that are not related to the signature itself. The success of signature verification will increase if the CNN is combined with supplementary feature extractors.

References

1. Hafemann LG, Sabourin R, Oliveira LS. Offline handwritten signature verification—literature review. In2017 seventh international conference on image processing theory, tools and applications (IPTA). IEEE. (2017) 1-8. https://doi.org/10.1109/IPTA.2017.8310112.
2. Jain A, Singh SK, Singh KP. Signature verification using geometrical features and artificial neural network classifier. Neural Computing and Applications. 33(2021) 6999-7010. https://doi.org/10.1007/s00521-020-05473-7.
3. Al-Omari YM, Abdullah SN, Omar K. State-of-the-art in offline signature verification system. In2011 International Conference on Pattern Analysis and Intelligence Robotics. IEEE. 1(2011) 59-64. https://doi.org/10.1109/ICPAIR.2011.5976912.
4. Mosaher QS, Hasan M. Offline handwritten signature recognition using deep convolution neural network. European Journal of Engineering and Technology Research. 7(4)(2022) 44-47. https://doi.org/10.24018/ejeng.2022.7.4.2851.
5. Al-banhawy NH, Mohsen H, Ghali NI. Signature identification and verification systems: a comparative study on the online and offline techniques. Future Computing and Informatics Journal. 5(1)(2020) 3. http://doi.org/10.54623/fue.fcij.5.1.3.
6. Serdouk Y, Nemmour H, Chibani Y. Artificial Immune Recognition System for Offline Handwritten Signature Verification. In: Bhattacharyya S, Mukherjee A, Pan I, Dutta P, Bhaumik AK, (Eds). Hybrid Intelligent Techniques for Pattern Analysis and Understanding. CRC Press. New York, 2017, pp.

49-68. http://dx.doi.org/10.1201/9781315154152-3.
7. Kumar R, Sharma JD, Chanda B. Writer-independent off-line signature verification using surroundedness feature. Pattern recognition letters. 33(3)(2012) 301-308. https://doi.org/10.1016/j.patrec.2011.10.009.
8. Bhunia AK, Alaei A, Roy PP. Signature verification approach using fusion of hybrid texture features. Neural Computing and Applications. 31(2019) 8737-8748. https://doi.org/10.1007/s00521-019-04220-x.
9. Ahmed SF, Alam MS, Hassan M, et al. Deep learning modelling techniques: current progress, applications, advantages, and challenges. Artificial Intelligence Review. 56(2023) 13521–13617. https://doi.org/10.1007/s10462-023-10466-8.
10. Foroozandeh A, Hemmat AA, Rabbani H. Offline handwritten signature verification and recognition based on deep transfer learning. In 2020 International Conference on Machine Vision and Image Processing (MVIP). IEEE. (2020) 1-7. https://doi.org/10.1109/MVIP49855.2020.9187481.
11. Alsuhimat FM, Mohamad FS. A Hybrid Method of Feature Extraction for Signatures Verification Using CNN and HOG a Multi-Classification Approach. IEEE Access. 11(2023) 21873-21882. https://doi.org/10.1109/ACCESS.2023.3252022.
12. Yapici MM, Tekerek A, Topaloglu N. Convolutional neural network based offline signature verification application. In 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT). IEEE. (2018) 30-34. https://doi.org/10.1109/IBIGDELFT.2018.8625290.
13. Gideon SJ, Kandulna A, Kujur AA, Diana A, Raimond K. Handwritten signature forgery detection using convolutional neural networks. Procedia computer science. 143(2018) 978-987. https://doi.org/10.1016/j.procs.2018.10.336.
14. Jahandad, Sam SM, Kamardin K, Sjarif NN, Mohamed N. Offline Signature Verification using Deep Learning Convolutional Neural Network (CNN) Architectures GoogLeNet Inception-v1 and Inception-v3. Procedia Computer Science. 161(2019) 475-483. https://doi.org/10.1016/j.procs.2019.11.147.
15. Hung PD, Bach PS, Vinh BT, Tien NH, Diep VT. Offline handwritten signature forgery verification using deep learning methods. In: Zhang YD, Senjyu T, So-In C, Joshi A. (Eds). Smart Trends in Computing and Communications. Springer Nature Singapore. Singapore. 2022 pp. 75-84. https://doi.org/10.1007/978-981-16-9967-2_8.
16. Dey S, Dutta A, Toledo JI, Ghosh SK, Lladós J, Pal U. Signet: Convolutional siamese network for writer independent offline signature verification. arXiv preprint arXiv:1707.02131. (2017). https://doi.org/10.48550/arXiv.1707.02131.
17. Lopes JA, Baptista B, Lavado N, Mendes M. Offline handwritten signature verification using deep neural networks. Energies. 15(20)(2022) 7611. https://doi.org/10.3390/en15207611.
18. Vohra K. Signature verification using support vector machine and convolution neural network. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 12(1S)(2021) 80-89. https://doi.org/10.17762/turcomat.v12i1S.1564.
19. Foroozandeh A, Hemmat AA, Rabbani H. Offline handwritten signature verification based on circlet transform and statistical features. In 2020 International Conference on Machine Vision and Image Processing (MVIP). IEEE. (2020) 1-5. https://doi.org/10.1109/MVIP49855.2020.9116909.
20. Batool FE, Attique M, Sharif M, et al. Offline signature verification system: a novel technique of fusion of GLCM and geometric features using SVM. Multimed Tools Appl. (2020) 1-20. https://doi.org/10.1007/s11042-020-08851-4.
21. Ajij M, Pratihar S, Nayak SR, Hanne T, Roy DS. Off-line signature verification using elementary combinations of directional codes from boundary pixels. Neural Computing and Applications. (2021) 1-18. https://doi.org/10.1007/s00521-021-05854-6.
22. [22] Kim P. Convolutional Neural Network. In: MATLAB Deep Learning. Apress, Berkeley, CA. 2017. pp 121–147. https://doi.org/10.1007/978-1-4842-2845-6_6.
23. Gu J, Wang Z, Kuen J, et al. Recent advances in convolutional neural networks. Pattern recognition. 77(2018) 354-377. https://doi.org/10.1016/j.patcog.2017.10.013