SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V12I7P125

Volume 12 Issue 7, 315-326, July 2025

Original Article

Offline Signature Verification Using Pre-Trained Deep Convolution Neural Network: SqueezeNet

Bhimraj Prasai Chetry[1], Biswajit Kar[2]

[1,2]*Department of Instrumentation Engineering, Central Institute of Technology Kokrajhar, Assam, India.*

[1]*Corresponding Author : bp.chetry@cit.ac.in*

Received: 13 May 2025 Revised: 15 June 2025 Accepted: 16 July 2025 Published: 31 July 2025

***Abstract** - Offline Signature Verification is a very important research area because signatures evolve throughout a person's life and have many applications such as person authentication, verification in financial transactions, institute certifications, legal documentation, etc. It has been socially, legally, and culturally accepted as a behavioural biometric for centuries. So, it is more prone to forgery than any other biometrics. So, in order to counteract forgery and accept genuine signatures, we have proposed an offline signature verification system using a pre-trained deep convolutional neural network called "SqueezeNet v1.0" to enhance the verification accuracy of the system. Here, the use of a pretrained SqueezeNet model is an effective approach, especially when we need a lightweight model that can perform well with fast inference in resource-constrained environments like signature verification. Signature verification is challenging work because of large intra-class diversity and small inter-class distinction while considering forgeries. Despite the progress made with traditional methods, these techniques often face challenges related to feature engineering and performance under noisy conditions, making them less effective compared to modern deep learning-based approaches. With the progress of deep learning, offline signature verification has seen significant improvements, particularly Convolutional Neural Networks (CNNs), which are able to self learn hierarchical feature representations from raw signature images, eliminating the need for manual feature extraction. Here, skilled forgery signatures of each user are used for training and testing purposes to make the system robust and more accurate. Our system is trained and tested on the CEDAR database for all fifty-five users having different types of signature information, yielding average testing accuracy of 98.98% using random forgeries and 98.07% using skilled forgeries. Testing accuracy of random forgeries lies between 93.75%-100% and testing accuracy of skilled forgeries lies between 72.92%-100%.*

***Keywords** - Behavioural biometric, CEDAR database, Convolutional Neural Networks, Offline signature verification, SqueezeNet.*

1. Introduction

Offline signature verification refers to the process of verifying handwritten signatures by using scanned images or digitally represented signatures, without real-time input from the signer. It is important for various sectors, including banking, legal, and governmental systems, all of which require signature-based authentication to verify the authenticity of signed transactions and documents. Offline signature verification is normally based on either structural or statistical properties of signatures. The prior processes involve various classification techniques, e.g., feature extraction, template matching, and machine learning algorithms to identify whether signatures are true or forged. Nonetheless, such methods often have limited robustness concerning the variety of styles and subtle variations of true vs. forged signatures. Traditional offline signature verification methods have relied on handcrafted features, such as geometric characteristics, pixel intensity values, and structural aspects of the signature. Sadly, they often struggle to deal with the large variance of signature styles and the nuances that differentiate genuine signatures from forgeries [1]. Recently, deep learning has dramatically changed the world of offline signature verification by being a more robust solution to more complicated variations in signature dynamics. Deep learning models, including Convolutional Neural Networks (CNNs), have shown themselves to be useful and powerful methods for signature verification, especially given their ability to learn hierarchical representations from image data [2] automatically. Deep learning models are capable of learning fine/grained and complex patterns in signature images without manual feature engineering, allowing for improved accuracy and robustness in similar signature verification tasks [3, 4]. Unlike other machine learning methods, deep learning models do not need extracted features that have been manually engineered and can handle complex variations associated with signature dynamics (i.e. scale, orientation, distortion, etc.). Deep learning in signature verification can be broadly categorized into two main types of approaches: verification-based models (authenticating a query signature against a reference signature) and classification-based models (determining the authenticity of a signature based on learned patterns) [5, 6]. Multiple studies have demonstrated that deep learning approaches generally outperform traditional methods of offline signature verification, lessening the false acceptance and false rejection rates [2, 7]. Use of lightweight, powerful pre-trained SqueezeNet CNN for the first time in an offline signature verification using both skill and random forgery to get very good verification results in a constrained environment, like a signature, is one of the novelties of our work.

2. Literature Review

Classic techniques for offline signature verification typically rely on manually extracted features and statistical methods. Early systems used feature extraction methods, including zoning (i.e., splitting the signature into separate pieces) and geometric features (i.e., length of strokes, curvature). Common methods include:

- Dynamic Time Warping (DTW): DTW is used for signature verification, especially online signature verification, but has also been adapted to offline signatures [8]. DTW works by aligning the two sets in time to compare the similarity of the two sequences.
- Hidden Markov Models (HMM): HMMs have been utilized in signature verification tasks for both offline and online scenarios. HMMs are trained with a sequence of signature features and learn the transition across different parts of the signature [9, 10]. The system measures how likely a subject is to sign the signature when looking at the learned model of a genuine signature.
- Support Vector Machines (SVM): SVM classifiers have been used for offline signature verification [11, 12]. In the study, by extracting features including geometric shapes, curvature, and stroke direction, the SVM classifier checks between genuine and forged signatures.

Although traditional approaches have been effective, these mostly failed from a feature engineering and variability perspective, particularly where features are conflated with noise (e.g. ink density, type of paper), and they are less scalable than approaches based on deep learning. Deep learning has brought significant advances to offline signature verification. Most notably, deep models can learn hierarchical representations of features from the raw signature image rather than rely on manual feature extraction. There are also some deep learning based approaches. Here are some of the major categories of deep learning based techniques:

- Convolutional Neural Networks (CNNs): CNNs are the most common offline signature verification architecture. CNNs have shown the best performance on some benchmark datasets [2]. CNNs can automatically derive spatial features from signature images, including stroke shapes and textures, and classify the signature as genuine or forged. CNNs typically consist of multiple convolutional and pooling layers, and a few fully-connected layers, where the latter performs the ultimate classification. For instance, [6] put together a system for signature verification using a CNN, and it easily surpassed traditional methods in terms of both accuracy and robustness. This model was trained with a large number of offline signature images and appears to generalise very well to unseen datasets.
- Recurrent Neural Networks (RNNs): RNNs and advanced versions of them, such as Long Short-Term Memory (LSTM) networks, are structured to work on sequential data. Therefore, RNNs are appropriate for signature verification tasks, especially when signature images are formed in sequences. RNNs have been previously used for online signature verification tasks. In the paper presented in [13], the authors adapted RNNs for an offline signature verification task, where they modelled the signature as a sequence of extracted features. [13] Introduced RNNs to perform offline signature verification and were able to achieve substantial gains in accuracy by modelling the temporal dependencies between stroke segments in the signature image.
- Generative Adversarial Networks (GANs): GANs are used to augment existing signature datasets by generating synthetic signature samples and can assist in the underlying structure of training offline signature verification systems. It will be mostly used in cases where training data is scarce [14]. GANs comprise two neural networks, the generator and discriminator. When the training procedure begins, the generator will generate synthetic signatures, and the discriminator will differentiate between real and generated signatures. This method has been demonstrated to improve performance by providing verification models with example signature samples from diverse sources [15].
- Transfer Learning: Transfer learning entails taking a model that has undergone training on a vast dataset and employing it by making it specific to signature verification tasks. This can help address the problem of insufficient labelled data, which is one of the problems with signature verification. For example, by using survey Convolutional Neural Networks (CNNs) that are pre-trained on vast image datasets like ImageNet, it is possible to accomplish higher performance with a smaller labelled sample set by transferring the general knowledge base of the model to the new signature verification dataset [2, 16, 17].

Hybrid models combining machine learning and deep learning have been proposed alongside deep learning approaches to boost the accuracy and robustness of offline signature verification. For example, some researchers have combined CNNs with SVMs or HMMs to increase classification accuracy [18].

3. Proposed Model

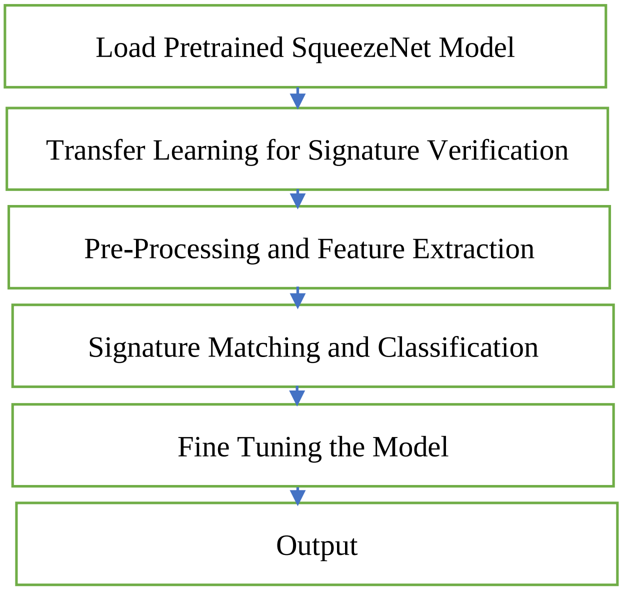


Fig. 1 Block diagram for offline signature verification using deep learning pretrained SqueezeNet network

The block diagram above in Figure 1 represents how a pretrained SqueezeNet model can be adapted for offline signature verification. The process involves multiple steps, from loading the pretrained SqueezeNet Network to final verification. Below is the breakdown of the blocks:

3.1. Block Diagram Description

The following steps outline how a pretrained SqueezeNet model can be adapted for offline signature verification:

Step 1: Pretrained Model (SqueezeNet)

SqueezeNet is a lightweight CNN architecture as shown in Figure 2. It is ideal for applications requiring limited computational resources. The network will achieve almost the same accuracy as other architectures (e.g., ResNet, VGG) for fewer parameters. Many pretrained versions of SqueezeNet are also available in popular deep learning libraries and can be customized for specific tasks, like signature verification [17].

SqueezeNet is a Convolutional Neural Network (CNN) architecture that is lightweight and is designed to strike a balance between model size and accuracy. It can be very helpful in low-computational resources or for instances when you need a smaller model with significantly quicker inference, thus it is an ideal solution for using offline signature verification systems based on a situation that may be constrained in resources.

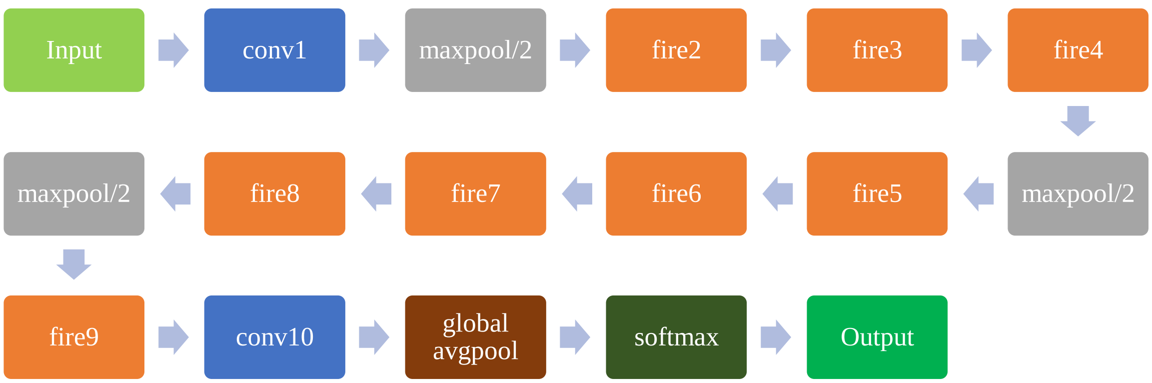


Fig. 2 Basic architecture of pre-trained SqueezeNet

Features of SqueezeNet:

- Compact Structure: When compared to other CNN architectures like VGG and ResNet, SqueezeNet has a small model size that achieves comparable performance with far fewer parameters.
- Fire Modules: SqueezeNet introduced the Fire Module, which is made of two layers: a squeeze layer consisting of 1x1 convolutions and an expand layer consisting of a combination of 1x1 and 3x3 convolutions. This two-layer structure is designed to reduce the number of parameters with some accuracy.
- Pre-trained Weights: As with other CNN architectures, they can fine-tune SqueezeNet with pre-trained weights for datasets like ImageNet. It can train the model specifically for any task, such as offline signature verification.

The use of a pretrained SqueezeNet model for offline signature verification is practical, especially when a lightweight model that can work quickly and efficiently in low-resource situations is valuable or necessary. Utilizing transfer learning and modifying SqueezeNet, a deep Convolutional Neural Network (CNN), you can then utilize it to identify distinguishing features that separate genuine and forged signatures [19].

Step 2: Transfer Learning for Signature Verification

Transfer learning means taking a pretrained model from a vast dataset (e.g. ImageNet) and customizing it on a specialized dataset for a specialized task (e.g. offline signature verification). There are unique patterns to signatures (e.g., strokes, curvature, and speed) that require the models to adapt from general visual patterns to specific handwriting features [20].

Step 3: Pre-processing and Feature Extraction

The first step in using SqueezeNet for offline signature verification is to preprocess the signature image and resize it to meet SqueezeNet's input dimensions. In this case, all the images within the dataset were resized into a fixed size of 227x227 pixels before inputting them to the image input layer. Each of the signature images in the CEDAR database was in grayscale PNG format. Therefore, the images were converted to RGB images to be compliant with the SqueezeNet image input layer. The image is then passed through SqueezeNet to extract the feature maps from the intermediate layers. These features are able to represent important aspects of a signature, including stroke patterns, shapes, and other exclusive signature features [21].

Step 4: Signature Comparison and Classification

After the extraction of the features, the next step is to compare the input signature features extracted with the reference features from genuine signatures. The similarity between these features can be calculated using distance-based similarity metrics such as Euclidean distance or cosine similarity. Following the matching process, a decision-making process (e.g., thresholding or a SoftMax classifier) is used to assign the signature a genuine or forged classification based on the resulting similarity score [22].

Step 5: Fine-tuning the Model

Fine-tuning is critical for appropriately adapting the pretrained model to the signature dataset. Fine-tuning will require freezing the first layer as the model detects simple patterns such as textures and edges, and retraining the later layers with the signature dataset. The model will learn high-level signature-specific features while appropriately adapting the embedded low-level features (e.g. lines and patterns), capturing the biases in the data that typically lead to challenging signature verification tasks [23].

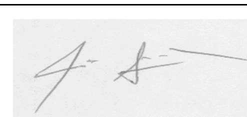
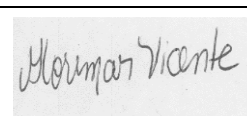
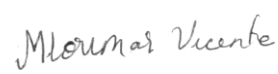
Table 1. First 05 layers of pre-trained network SqueeezeNet as a sample

Layers	Name	Type	Activations
1	data 227*227*3 images with 'Zerocenter' normalization	Image Input	227(S)*227(S)*3(C)*1(B)
2	conv1 64 3*3*3 convolutions with stride [2 2] and Padding [0 0 0 0]	2-D Convolution	113(S)*113(S)*64(C)*1(B)
3	relu_conv1 ReLU	ReLU	113(S)*113(S)*64(C)*1(B)
4	pool1 3*3 max pooling With stride [2 2] and Padding [0 0 0 0]	2-D Max Pooling	56(S)*56(S)*64(C)*1(B)
5	fire2-squeeze 1*1 16 1*1*64 convolutions with stride [2 2] and Padding [0 0 0 0]	2-D Convolution	56(S)*56(S)*16(C)*1(B)

Table 1 above shows the sample of the first 05 layers of the SqueezeNet Pre-Trained network out of a total of 68 layers, having 1.2M total learnables.

3.2. About Database

The CEDAR dataset was selected for testing the system's performance due to its widespread use by researchers and accessibility. The CEDAR dataset comprises 55 writers, each contributing 24 genuine and 24 skilled forged signatures. The dataset includes 1,320 genuine and 1,320 skilled forgeries, all stored in grayscale PNG format [18]. Figure 3 below shows some samples of genuine signatures for some writers from the CEDAR dataset and corresponding skilled forged signatures for those same writers. The CEDAR database is organized into two subfolders: full_forg and full_org. The full_forg folder contains 1,320 forged signatures—24 for each of the 55 writers—while the full_org folder holds 1,320 genuine signatures, also with 24 samples per writer [24].

User	Genuine Signatures	Skill Forgery Signatures
1		
2		

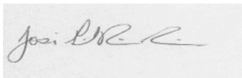
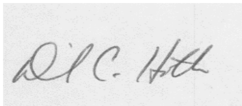
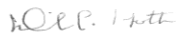
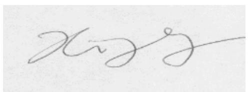
3		
4		
5		

Fig. 3 Some sample signatures of the CEDAR database

3.3. Verification

3.3.1. Training

Here, training is done with randomly selected 60% signatures data for each of the 55 users of the CEDAR database. The Training Confusion Matrix for User1 is shown in Figure 4 below.

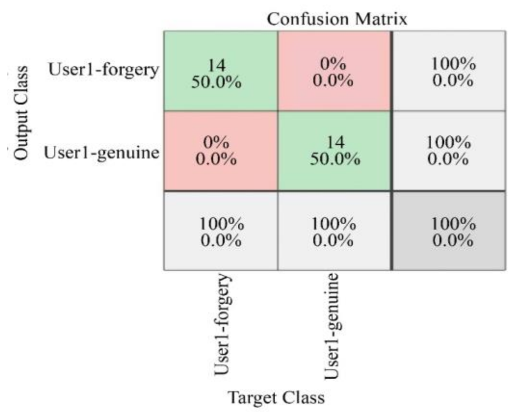


Fig. 4 Training confusion matrix for user1

3.3.2. Testing

Here, testing was done with all the 100% signatures data twice, once with the skill forgery data given in the CEDAR database. And once with the random forgery created using various users' data mixed with some skill forgery data of the same particular user.

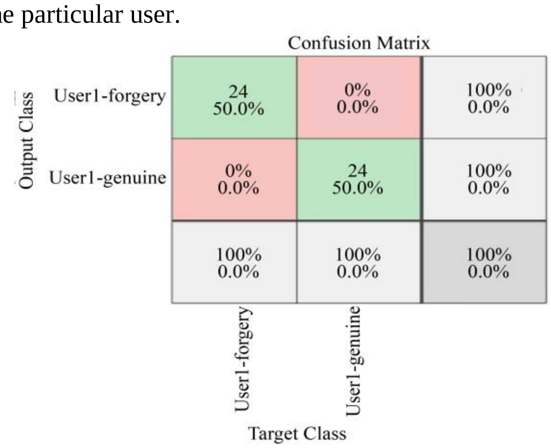


Fig. 5 Testing confusion matrix for user1 using random forgeries

Finally, accuracy is calculated for each case. Testing the Confusion Matrix for User1, using Random Forgeries and Skilled Forgeries, is shown in Figures 5 and 6, respectively.

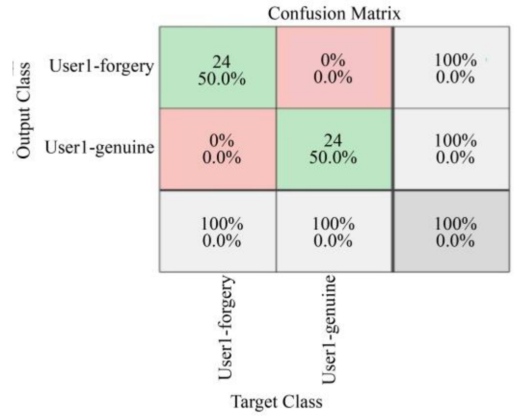


Fig. 6 Testing confusion matrix for user1 using skilled forgeries

4. Verification Results and Performance Evaluation

Accuracy is a basic metric to evaluate how well an offline signature verification system correctly classifies genuine and forged signatures. In addition, accuracy is commonly used in pattern recognition and classification tasks and is defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (1)$$

TP (True Positive): Genuine signatures correctly classified as genuine.

TN (True Negative): Forged signatures correctly classified as forged.

FP (False Positive): Forged signatures incorrectly classified as genuine.

FN (False Negative): Genuine signatures incorrectly classified as forged.

In offline signature verification, accuracy is a commonly used criterion to measure the effectiveness of a system. The above formula in Equation (1) was employed to evaluate the accuracy of an offline signature verification system by the authors, highlighting the importance of accuracy with complementary metrics such as False Acceptance Rate (FAR), False Rejection Rate (FRR), etc, in evaluating the performance of the system [22, 25]. Likewise, the same formula was used in their research to study offline signature verification, measuring the performance of offline signature verification employing discrete wavelet transforms and other machine learning methodologies by [21]. Here, training, testing accuracy and training time for all 55 users are recorded individually in Table 2 below for analysis.

Table 2. Training and verification results (training 60% randomly and testing with all 100%)

User	Training Accuracy(%)	Testing Accuracy (%)		Training Elapsed Time
		Random Forgeries	Skilled Forgeries	
User1	100	100.00	100.00	1 min 17 sec
User2	100	100.00	100.00	1 min 21 sec
User3	100	97.92	100.00	1 min 23 sec
User4	100	97.92	91.67	1 min 18 sec
User5	100	95.83	97.92	1 min 21 sec
User6	100	97.92	72.92	1 min 19 sec
User7	100	97.92	100.00	1 min 23 sec
User8	100	97.92	97.92	1 min 22 sec
User9	100	97.92	97.92	1 min 17 sec
User10	100	100.00	100.00	1 min 18 sec
User11	100	97.92	100.00	1 min 20 sec
User12	100	100.00	100.00	1 min 12 sec
User13	100	100.00	100.00	1 min 19 sec
User14	100	100.00	93.75	1 min 15 sec
User15	100	100.00	100.00	1 min 4 sec
User16	100	100.00	89.58	42 sec
User17	100	100.00	100.00	44 sec
User18	100	100.00	100.00	43 sec
User19	100	100.00	100.00	41 sec
User20	100	100.00	97.92	1 min 2 sec
User21	100	100.00	100.00	47 sec
User22	100	100.00	100.00	56 sec
User23	100	93.75	95.83	49 sec
User24	100	97.92	97.92	51 sec
User25	100	100.00	100.00	53 sec
User26	100	100.00	100.00	53 sec
User27	100	100.00	95.83	1 min 33 sec
User28	100	100.00	100.00	1 min 21 sec
User29	100	100.00	100.00	1 min 21 sec
User30	100	100.00	100.00	1 min 22 sec
User31	100	97.92	97.92	1min 27 sec
User32	100	100.00	100.00	1 min 20 sec
User33	100	97.92	89.58	1 min 25 sec
User34	100	100.00	100.00	1 min 17 sec
User35	100	100.00	100.00	1 min 18 sec
User36	100	100.00	100.00	1 min 16 sec
User37	100	100.00	100.00	1 min 18 sec
User38	100	100.00	93.75	46 sec
User39	100	97.92	97.92	1 min 24 sec
User40	100	100.00	100.00	1 min 22 sec
User41	100	95.83	100.00	1 min 32 sec
User42	100	100.00	100.00	1 min 19 sec
User43	100	100.00	100.00	1 min 23 sec
User44	100	93.75	100.00	46 sec
User45	100	97.92	100.00	1 min
User46	100	100.00	100.00	45 sec
User47	100	97.92	97.92	50 sec
User48	100	95.83	87.50	51 sec
User49	100	100.00	100.00	52 sec

User50	100	100.00	100.00	51 sec
User51	100	100.00	100.00	1 min 31 sec
User52	100	100.00	100.00	51 sec
User53	100	95.83	100.00	1 min 21 sec
User54	100	100.00	100.00	1 min 22 sec
User55	100	100.00	100.00	1 min 24 sec
Average	100	98.98	98.07	1 min 9 sec
Highest	100	100.00	100.00	1 min 33 sec
Lowest	100	93.75	72.92	41 sec

Apart from the accuracy above, Precision, Recall and F1-Score metrics were used to evaluate the proposed model's performance for testing with both random and skilled forgeries. A breakdown of each metric with formulas contextualized for signature verification is given below [26]:

Accuracy: It measures the proportion of total correct predictions (both positive and negative) out of all predictions. The formula for it is given above in Equation (1).

Precision: Precision measures the proportion of true positive predictions out of all predicted positives. The formula for it is given below in Equation (2)

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall (Sensitivity or True Positive Rate): Recall measures the proportion of true positives out of all actual positive cases. The formula for it is given below in Equation (3)

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

F1-Score: F1 Score is the harmonic mean of Precision and Recall. It balances the two when you want to consider both false positives and false negatives. The formula for it is given below in Equation (4)

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4}$$

True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) have been recorded in Tables 3 and 4 below from the Confusion Matrix of each individual User while testing once with Random Forgery and subsequently with Skilled Forgeries. Further, Precision, Recall, and F1 Scores were calculated for all 55 users while testing using random and skilled forgeries, and these were presented in Tables 3 and 4, respectively.

Table 3. Precision, recall and F1-score while testing using random forgeries for all 55 users

User	Random Forgeries						
	TP	TN	FP	FN	Precision %	Recall %	F1-Score %
User1	24	24	0	0	100.00	100.00	100.00
User2	24	24	0	0	100.00	100.00	100.00
User3	23	24	0	1	100.00	95.83	97.87
User4	23	24	0	1	100.00	95.83	97.87
User5	22	24	0	2	100.00	91.67	95.65
User6	23	24	0	1	100.00	95.83	97.87
User7	23	24	0	1	100.00	95.83	97.87
User8	23	24	0	1	100.00	95.83	97.87
User9	23	24	0	1	100.00	95.83	97.87
User10	24	24	0	0	100.00	100.00	100.00
User11	23	24	0	1	100.00	95.83	97.87
User12	24	24	0	0	100.00	100.00	100.00
User13	24	24	0	0	100.00	100.00	100.00
User14	24	24	0	0	100.00	100.00	100.00
User15	24	24	0	0	100.00	100.00	100.00
User16	24	24	0	0	100.00	100.00	100.00
User17	24	24	0	0	100.00	100.00	100.00
User18	24	24	0	0	100.00	100.00	100.00
User19	24	24	0	0	100.00	100.00	100.00

User20	24	24	0	0	100.00	100.00	100.00
User21	24	24	0	0	100.00	100.00	100.00
User22	24	24	0	0	100.00	100.00	100.00
User23	21	24	0	3	100.00	87.50	93.33
User24	23	24	0	1	100.00	95.83	97.87
User25	24	24	0	0	100.00	100.00	100.00
User26	24	24	0	0	100.00	100.00	100.00
User27	24	24	0	0	100.00	100.00	100.00
User28	24	24	0	0	100.00	100.00	100.00
User29	24	24	0	0	100.00	100.00	100.00
User30	24	24	0	0	100.00	100.00	100.00
User31	24	23	1	0	96.00	100.00	97.96
User32	24	24	0	0	100.00	100.00	100.00
User33	23	24	0	1	100.00	95.83	97.87
User34	24	24	0	0	100.00	100.00	100.00
User35	24	24	0	0	100.00	100.00	100.00
User36	24	24	0	0	100.00	100.00	100.00
User37	24	24	0	0	100.00	100.00	100.00
User38	24	24	0	0	100.00	100.00	100.00
User39	24	23	1	0	96.00	100.00	97.96
User40	24	24	0	0	100.00	100.00	100.00
User41	22	24	0	2	100.00	91.67	95.65
User42	24	24	0	0	100.00	100.00	100.00
User43	24	24	0	0	100.00	100.00	100.00
User44	21	24	0	3	100.00	87.50	93.33
User45	23	24	0	1	100.00	95.83	97.87
User46	24	24	0	0	100.00	100.00	100.00
User47	24	23	1	0	96.00	100.00	97.96
User48	22	24	0	2	100.00	91.67	95.65
User49	24	24	0	0	100.00	100.00	100.00
User50	24	24	0	0	100.00	100.00	100.00
User51	24	24	0	0	100.00	100.00	100.00
User52	24	24	0	0	100.00	100.00	100.00
User53	22	24	0	2	100.00	91.67	95.65
User54	24	24	0	0	100.00	100.00	100.00
User55	24	24	0	0	100.00	100.00	100.00
Average					**99.78**	**98.18**	**98.94**
Highest					**100.00**	**100.00**	**100.00**
Lowest					**96.00**	**87.50**	**93.33**

Table 4. Precision, recall and F1-score while testing using skilled forgeries for all 55 users

User	**Skilled Forgeries**						
	TP	**TN**	**FP**	**FN**	**Precision %**	**Recall %**	**F1-Score %**
User1	24	24	0	0	100.00	100.00	100.00
User2	24	24	0	0	100.00	100.00	100.00
User3	24	24	0	0	100.00	100.00	100.00
User4	20	24	0	4	100.00	83.33	90.91
User5	23	24	0	1	100.00	95.83	97.87
User6	11	24	0	13	100.00	45.83	62.86
User7	24	24	0	0	100.00	100.00	100.00
User8	23	24	0	1	100.00	95.83	97.87

User9	23	24	0	1	100.00	95.83	97.87
User10	24	24	0	0	100.00	100.00	100.00
User11	24	24	0	0	100.00	100.00	100.00
User12	24	24	0	0	100.00	100.00	100.00
User13	24	24	0	0	100.00	100.00	100.00
User14	21	24	0	3	100.00	87.50	93.33
User15	24	24	0	0	100.00	100.00	100.00
User16	19	24	0	5	100.00	79.17	88.37
User17	24	24	0	0	100.00	100.00	100.00
User18	24	24	0	0	100.00	100.00	100.00
User19	24	24	0	0	100.00	100.00	100.00
User20	23	24	0	1	100.00	95.83	97.87
User21	24	24	0	0	100.00	100.00	100.00
User22	24	24	0	0	100.00	100.00	100.00
User23	22	24	0	2	100.00	91.67	95.65
User24	23	24	0	1	100.00	95.83	97.87
User25	24	24	0	0	100.00	100.00	100.00
User26	24	24	0	0	100.00	100.00	100.00
User27	22	24	0	2	100.00	91.67	95.65
User28	24	24	0	0	100.00	100.00	100.00
User29	24	24	0	0	100.00	100.00	100.00
User30	24	24	0	0	100.00	100.00	100.00
User31	24	23	1	0	96.00	100.00	97.96
User32	24	24	0	0	100.00	100.00	100.00
User33	19	24	0	5	100.00	79.17	88.37
User34	24	24	0	0	100.00	100.00	100.00
User35	24	24	0	0	100.00	100.00	100.00
User36	24	24	0	0	100.00	100.00	100.00
User37	24	24	0	0	100.00	100.00	100.00
User38	21	24	0	3	100.00	87.50	93.33
User39	24	23	1	0	96.00	100.00	97.96
User40	24	24	0	0	100.00	100.00	100.00
User41	24	24	0	0	100.00	100.00	100.00
User42	24	24	0	0	100.00	100.00	100.00
User43	24	24	0	0	100.00	100.00	100.00
User44	24	24	0	0	100.00	100.00	100.00
User45	24	24	0	0	100.00	100.00	100.00
User46	24	24	0	0	100.00	100.00	100.00
User47	24	23	1	0	96.00	100.00	97.96
User48	18	24	0	6	100.00	75.00	85.71
User49	24	24	0	0	100.00	100.00	100.00
User50	24	24	0	0	100.00	100.00	100.00
User51	24	24	0	0	100.00	100.00	100.00
User52	24	24	0	0	100.00	100.00	100.00
User53	24	24	0	0	100.00	100.00	100.00
User54	24	24	0	0	100.00	100.00	100.00
User55	24	24	0	0	100.00	100.00	100.00
Average					**99.78**	**96.36**	**97.77**
Highest					**100.00**	**100.00**	**100.00**
Lowest					**96.00**	**45.83**	**62.86**

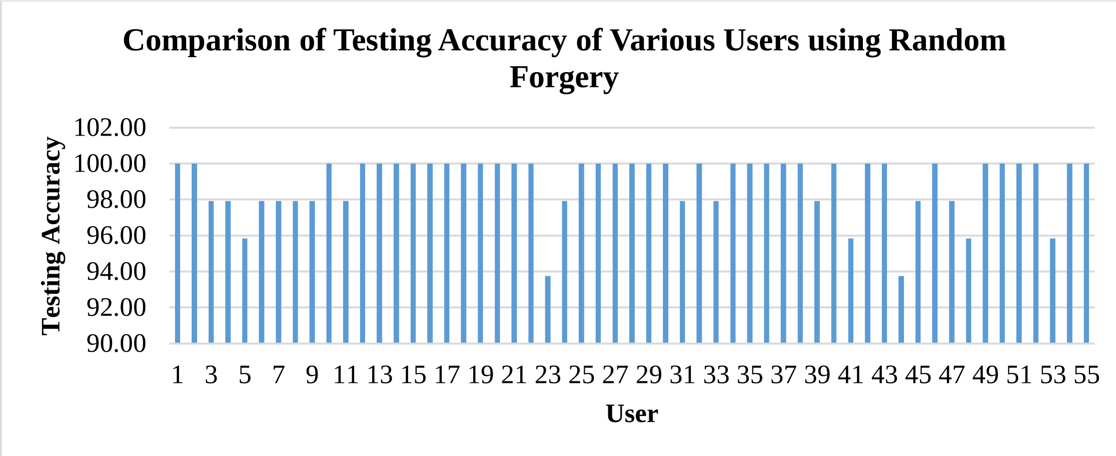


Fig. 7 Comparison of testing accuracy of various users using random forgery

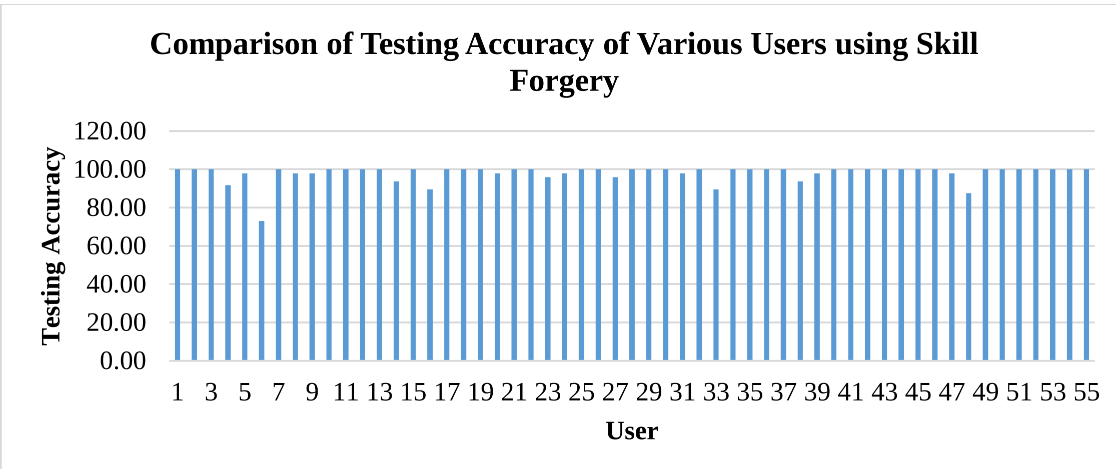


Fig. 8 Comparison of testing accuracy of various users using skill forgery

Table 5. Verification Report of Proposed Method

Performance Parameters	Results Obtained (%)	
	Random Forgery	Skilled Forgery
Average Testing Accuracy	98.98	98.07
Average Precision	99.78	99.78
Average Recall	98.18	96.36
Average F1-Score	98.94	97.77

The performance analysis report of our proposed offline signature verification system in terms of Testing Accuracy, Precision, Recall and F1-Score, which performs remarkably well, is shown in Table 5.

5. Results and Conclusion

Our Pretrained SqueezeNet is trained and tested on the CEDAR database for all fifty-five users, yielding average testing accuracy of 98.98% using Random Forgeries and 98.07% using Skilled Forgeries, as shown in Table 2 above. For random forgeries, testing accuracy ranges from (93.75%-100%) for all fifty-five users; similarly, for skilled forgeries, testing accuracy ranges from (72.92%-100%). The average training time elapsed is observed to be 1 min 09 sec, with a highest of 1 min 33 sec and a lowest of 41 sec. The highest and lowest training accuracy is 100%. The highest testing accuracy for random as well as skill forgery is 100%. The lowest testing accuracy is 93.75% and 72.92% respectively, for random forgery and skill forgery. All the above parameters are depicted in Table 2 above. It is clear that the model's performance with average testing accuracy of 98.98% (Using Random Forgery) and 98.07% (Using Skill Forgery) indicates great overall efficiency. The average precision of 99.78% (Using Random Forgery) and 99.78% (Using Skill Forgery) shows that the model effectively reduces false positives. With an average recall of 98.18% (Using Random Forgery) and 96.36% (Using Skill Forgery), the model minimizes false negatives. Moreover, a well-balanced tradeoff between recall and precision, known as F1-Score, comes out to be 98.94% (Using Random Forgery) and 97.77% (Using Skill Forgery). All the above findings suggest the model's dependability, high accuracy, and careful handling of false positives and negatives. Hence, the values of Average Testing Accuracy, Average Precision, Average Recall and Average F1-Score from Tables 2, 3, 4 and 5 indicate that our proposed system performs remarkably well in line with the state-of-the-art results presented to date.

Using a pretrained SqueezeNet deep learning model for offline signature verification is an effective approach, especially when a lightweight model is needed that can perform well in resource-constrained environments with limited signature data. Comparison of the testing accuracy of various users using Random Forgery and Skill Forgery is shown in Figures 7 and 8 as a bar diagram, respectively.

6. Conclusion

Deep learning has brought notable changes in offline signature verification. In terms of accuracy and resilience, models like CNNs, RNNs, and GANs have surpassed conventional methods. However, issues like real-time processing, complex forgeries, and small datasets must be resolved. Hybrid models that combine deep learning and conventional techniques and developments in data augmentation and model optimization are expected to propel further advancements in signature verification systems as the field develops. By altering a pretrained SqueezeNet model for the purpose of offline signature verification, we have taken advantage of deep learning's power while keeping a lightweight and efficient model appropriate for implementation on devices with constrained computational resources. SqueezeNet is a lightweight Convolutional Neural Network (CNN) architecture designed to achieve a reasonable balance between accuracy and model size. It is especially helpful for applications with limited computational resources or a smaller model with rapid inference, making it a good choice for offline signature verification tasks in resource-constrained environments. In real-world scenarios, signature verification can be significantly challenged by various distortions such as stamps, overlapping text, smudges, or background noise, which are out of the scope of this research. These interferences can hinder accurate verification and authentication. Therefore, the development of robust techniques for the effective removal or mitigation of such distortions presents a valuable direction for future research. Advancing in this area could greatly enhance the reliability and accuracy of automatic offline signature verification systems.

References

[1] Luiz G. Hafemann, Robert Sabourin, and Luiz S. Oliveira, "Offline Handwritten Signature Verification — Literature Review," *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, Montreal, QC, Canada, pp. 1-8, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[2] Jahandad et al., "Offline Signature Verification Using Deep Learning Convolutional Neural Network (CNN) Architectures GoogLeNet Inception-v1 and Inception-v3," *Procedia Computer Science*, vol. 161, pp. 475-483, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[3] Ansu Liz Thomas, and J.E. Judith, "A Comprehensive Analysis of Feature Extraction Techniques for Human Activity Recognition Using Deep Learning," *2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT)*, Kollam, India, pp. 1876-1882, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[4] S.D. Bhavani, and R.K. Bharathi, "A Multi-Dimensional Review on Handwritten Signature Verification: Strengths and Gaps," *Multimedia Tools and Applications*, vol. 83, pp. 2853-2894, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[5] Neha Sharma, Sheifali Gupta, and Puneet Mehta, "A Comprehensive Study on Offline Signature Verification," *Journal of Physics: Conference Series*, vol. 1969, pp. 1-17, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[6] G. Abinesh, V. Kavitha, and J.V. Prajith, "Signature Verification Using Deep Learning and CNN," *International Journal of Innovative Science and Research Technology*, vol. 10, no. 3, pp. 374-381, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[7] Aman Singla, and Ajay Mittal, "Exploring Offline Signature Verification Techniques: A Survey Based On Methods and Future Directions," *Multimedia Tools and Applications*, vol. 84, pp. 2835-2875, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[8] A. Piyush Shanker, and A.N. Rajagopalan, "Off-Line Signature Verification Using DTW," *Pattern Recognition Letters*, vol. 28, no. 12, pp. 1407-1414, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[9] R. Kashi et al., "A Hidden Markov Model Approach to Online Handwritten Signature Verification," *International Journal on Document Analysis and Recognition*, vol. 1, pp. 102-109, 1998. [CrossRef] [Google Scholar] [Publisher Link]

[10] S. Adebayo Daramola, and T. Samuel Ibiyemi, "Offline Signature Recognition Using Hidden Markov Model (HMM)," *International Journal of Computer Applications*, vol. 10, no. 2, pp. 17-22, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[11] Faiza Eba Batool et al., "Offline Signature Verification System: A Novel Technique of Fusion of GLCM and Geometric Features Using SVM," *Multimedia Tools and Applications*, vol. 83, pp. 14959-14978, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[12] D.S. Sunil Kumar, "Offline Signature Verification Based on Ensemble of Features Using Support Vector Machine," *International Journal of Computer Applications*, vol. 184, no. 45, pp. 24-29, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] Rajib Ghosh, "A Recurrent Neural Network Based Deep Learning Model for Offline Signature Verification and Recognition System," *Expert Systems with Applications*, vol. 168, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[14] Xiaoguang Jiang, "Offline Handwritten Signature Recognition Based on Generative Adversarial Networks," *International Journal of Biometrics*, vol. 16, no. 3/4, pp. 236-255, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[15] M. Muzaffar Hameed et al., "OffSig-SinGAN: A Deep Learning-Based Image Augmentation Model for Offline Signature Verification," *Computers, Materials & Continua*, vol. 76, no. 1, pp. 1267-1289, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[16] Shyang-Jye Chang, and Tai-Rong Wu, "Development of a Signature Verification Model Based on a Small Number of Samples," *Signal Image and Video Processing*, vol. 18, pp. 285-294, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[17] Saeeda Naz, Kiran Bibi, and Riaz Ahmad, "DeepSignature: Fine-Tuned Transfer Learning Based Signature Verification System," *Multimedia Tools and Applications*, vol. 81, pp. 38113-38122, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[18] Fadi Mohammad Alsuhimat, and Fatma Susilawati Mohamad, "A Hybrid Method of Feature Extraction for Signatures Verification Using CNN and HOG A Multi-Classification Approach," *IEEE Access*, vol. 11, pp. 21873-21882, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] Min Hao et al., "SqueezeNet: An Improved Lightweight Neural Network for Sheep Facial Recognition," *Applied Sciences*, vol. 14, no. 4, pp. 1-13, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[20] Yash Gupta et al., "Handwritten Signature Verification Using Transfer Learning and Data Augmentation," *Proceedings of International Conference on Intelligent Cyber-Physical Systems*, pp. 233-245, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[21] Suvarna Joshi, and Abhay Kumar, "Feature Extraction Using DWT with Application to Offline Signature Identification," *Proceedings of the Fourth International Conference on Signal and Image Processing 2012*, pp. 285-294, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[22] Shih-Yin Ooi, Andrew Beng-Jin Teoh, and Thian-Songa Ong, "Offline Signature Verification through Biometric Strengthening," *2007 IEEE Workshop on Automatic Identification Advanced Technologies*, Alghero, Italy, pp. 226-231, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[23] Deepa AB, and Varghese Paul, "Brain Tumor Classification with Selective Fine Tuning Using Transfer Learning," *Science & Technology Asia*, vol. 30, no. 2, pp. 71-83, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[24] Md Ajij et al., "Off-Line Signature Verification Using Elementary Combinations of Directional Codes from Boundary Pixels," *Neural Computing & Applications*, vol. 35, pp. 4939-4956, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[25] Oona Rainio, Jarmo Teuho, and Riku Klén, "Evaluation Metrics and Statistical Tests for Machine Learning," *Scientific Reports*, vol. 14, pp. 1-14, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[26] Muhammad Azi Saputra, and Ida Nurhaida, "Signature Originality Verification Using A Deep Learning Approach," *Electronic Journal of Education Social Economics and Technology*, vol. 5, no. 1, pp. 19-29, 2024. [CrossRef] [Google Scholar] [Publisher Link]