Information Sciences and Technological Innovations

www.isti.reapress.com

Inf. Sci. Technol. Innov. Vol. 3, No. 1 (2026) 331–345.

Paper Type: Original Article

Fingerprint Presentation Attack Detection: A Comparative Evaluation of Support Vector Machine and K-Nearest Neighbor Classification

Ilobekemen Perpetual Oladoja[1,*], Ugwu Chukwuemeka Christian[2], Ifeoluwa Daniel Oseni[1], Oladoja Teniola Abdulhafiz[3]

[1] Department of Computer Science, School of Computing, Federal University of Technology, Akure; ipoladoja@futa.edu.ng; oseniid@futa.edu.ng.

[2] Department of Data Science, School of Computing, Federal University of Technology, Akure; ugwucc@futa.edu.ng.

[3] Department of Mathematics and Computer Science, Elizade University, Ilara-Mokin, Ondo State, Nigeria; tenioladoja05@gmail.com.

Citation:

Received: 13 August 2025 *Revised: 07 November 2025* *Accepted: 22 January 2026*	Oladoja, I. P., Ugwu, C. C., Oseni, I. D., & Oladoja, T. A. (2026). Fingerprint presentation attack detection: A comparative evaluation of support vector machine and K-nearest neighbor classification. *Information Sciences and Technological Innovations*, *3*(1), 331-345.

Abstract

The human hand consists of the wrist, palm, and fingers. It is used to feel, hold, and move objects. Each finger has a unique ridge pattern that develops before birth and remains unchanged throughout a person's lifetime which is a major reason why it is often used for identification purposes. Fingerprint recognition is used to verify an identity by comparing a person's fingerprint with a previously registered fingerprint. Despite its widespread use, fingerprint recognition remains vulnerable to presentation attacks. A presentation attack occurs when a fake biometric sample is presented to a sensor to impersonate a genuine user. This study compares Support Vector Machine (SVM) and K-Nearest Neighbor (KNN) for detecting fake fingerprints using Local Binary Pattern (LBP) features from the FVC2000 and SOCOfing datasets. On the combined SOCOFing dataset, SVM achieved an accuracy of 86.8%, a precision of 71.4%, a recall of 87.1%, and an F1-score of 75.6%. Both models showed improved performance from the Easy to the Hard tier, while performance on the combined tier was lower. SVM had a peak accuracy of 97.6% and ROC-AUC of 99.7% on the most heavily altered (Hard) subset alone. This suggests that, counterintuitively, heavier digital alteration may be easier to detect than subtle alteration. On the FVC2000 benchmark, both classifiers exceeded 99.5% accuracy, although this result may reflect the relative ease of separating genuine sensor captures from synthetically generated samples. These results indicate that classical Machine Learning (ML) models remain a viable and competitive option for fingerprint presentation attack detection, with SVM outperforming KNN, while detection performance is strongly dependent on alteration or attack severity rather than being uniform across difficulty levels.

Keywords: Fingerprint presentation attack detection, Biometric system, Spoofing attack, Support vector machine, Local binary pattern, K-nearest neighbor.

1 | Introduction

A biometric system is an information system that uses the biological features of a human being for identity authentication or authorization. Common biological features include, but are not limited to, the iris, fingerprint, facial structure, and voice of an individual. These features are sufficiently unique to each person

Corresponding Author: ipoladoja@futa.edu.ng

https://doi.org/10.48314/isti.vi.59

Licensee System Analytics. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0).

that even biological identical twins do not share identical patterns despite their close similarity. Fingerprint recognition is one of the most commonly used biometric systems globally.

Biometric systems are among the most secure information systems in existence [1]. Despite the uniqueness of each human feature pattern, bad actors attempt to trick these systems into granting unauthorized access using presentation attacks.

A presentation attack is a type of sensor-based spoofing attack in which an attacker presents a fake biometric feature to a system's sensor (such as a camera) with the sole purpose of falsely claiming an identity and bypassing the biometric system [2].

One of the most common presentation attack instruments is the fake fingerprint [3]. Presentation attacks can be categorized as either static or dynamic; static presentation attacks are less computationally intensive than dynamic ones [4].

A fingerprint scanner is a convenient, reliable, and accurate example of a biometric system [5]. Attackers circumvent fingerprint-based biometric systems by fabricating artificial fingerprint replicas from materials such as gelatin, paraffin wax, wood glue, or modelling compounds, which are then used to impersonate authorized users. Such a replica can be produced with or without the cooperation of the individual being cloned [6], and can be reproduced through printing, casting, or fabrication of a prosthetic finger that simulates a genuine biometric trait.

Artificial intelligence can likewise be used by attackers to generate artificial fingerprints or copies capable of deceiving a biometric system. A robust Fingerprint Presentation Attack Detection (FPAD) method is therefore needed to enhance the security and integrity of fingerprint recognition systems [7].

Several works have examined whether a fingerprint sample presented to a sensor is real or fake, most commonly through liveness detection, which uses physiological properties to distinguish a real fingerprint from a fake one. Liveness detection methods are generally expected to be non-invasive, user-friendly, and fast.

This research develops, analyzes, and compares two supervised Machine Learning (ML) algorithms, which are Support Vector Machine (SVM) and K-Nearest Neighbor (KNN), for detecting fingerprint presentation attacks. SVM finds an optimal boundary that helps separate fake fingerprints from genuine ones. KNN is a non-parametric algorithm that classifies a new sample based on the labels of its nearest neighbors, without assuming any particular data distribution. The specific objectives of this work are to

I. Extract discriminative texture features from fingerprint images using Local Binary Patterns (LBPs).
II. Train and tune SVM and KNN classifiers on these features.
III. Evaluate and compare both classifiers on standard performance metrics to determine which is better suited for this task.

The motivation for this work is to create a lightweight, classical machine-learning alternative to the deep-learning-based methods that dominate the recent presentation-attack-detection literature, as classical models typically require less computation and training data.

2|Literature Review

In this section, we review related work on FPAD. The existing approaches can be divided into four families: Deep transfer learning, ensemble learning, frequency-domain analysis, and reference quality metric-based approaches. These families are reported to suffer from poor generalization to unseen attack materials, low accuracy for lightweight models, and high computational complexity for deep architectures.

In many applications, fingerprint biometric authentication is used to verify identity. It is unique as a security trait, but still vulnerable to presentation attacks using spoofs, which are clones of an original biometric feature

used to impersonate a security measure. FPAD refers to methods and techniques that detect and prevent such fraudulent attacks by distinguishing a real fingerprint from a fake one.

Webster et al. [8] introduced psychophysically informed training for FPAD. The goal was to improve generalization to unseen attacks by weighting the training loss based on human-annotated saliency maps and annotation time. These measures reflect the perceptual relevance of different fingerprint regions. The LivDet 2021 dataset was used to evaluate the proposed method, and the results showed strong benchmark performance with a relatively small amount of training data.

Purnapatra et al. [9] introduced a large presentation-attack dataset of contactless single-fingertip images and synthetically generated deepfakes, and used DenseNet-121 and NasNetMobile CNNs for live-vs-spoof binary classification; DenseNet-121 achieved a 0.14% attack classification error rate.

Adam and Sathesh [10] applied Ridgelet transform to extract features from fingerprint images and used an ensemble of SVM classifiers to distinguish genuine fingerprints from spoof samples. Their approach achieved an accuracy of 90.34% and performed better than the single-classifier methods considered by the study. The findings suggest that fingerprint liveness detection can be improved when transform-based feature extraction is combined with ensemble classification.

Five commercial fingerprint scanners were evaluated by Kauba et al. [6] using spoof fingerprints that are made of silicone, latex, wax, and other materials. Most of the scanners were found to be vulnerable to presentation attacks. The Integrated Biometrics Columbo was the main exception, as its active capacitive sensing technology prevented successful spoofing. The findings show that several commercial fingerprint systems can still be vulnerable to inexpensive spoofing materials.

A dynamic presentation attack detection method that uses sequential thermal and optical image frames was proposed by Husseis et al. [11]. First-order statistical features such as mean and entropy were computed from the image sequences and subsequently classified using SVM. The approach achieved Bona Fide Presentation Classification Error Rates (BPCERs) of 18.1% for thermal images and 19.5% for optical images. These results demonstrate the potential of temporal information in distinguishing genuine fingerprints from presentation attacks.

M. Jomaa et al. [12] proposed an end-to-end Deep Learning (DL) framework for multimodal biometrics via the fusion of fingerprint and Electrocardiogram (ECG) signals. EfficientNet models process the fingerprint images, while one-dimensional and two-dimensional convolutional neural networks process the ECG signals. On a customized multimodal dataset, the fusion of both modalities achieved higher classification accuracy than fingerprint-only models.

An adversarial-learning-based method that can detect unseen presentation attacks by jointly training a feature extractor and classifier under adversarial constraints to reduce the influence of spoof material differences was proposed by Pereira et al. [13]. This method achieved better detection of previously unseen attacks than the baseline methods.

In 2018, Marasco et al. [14] studied non-cooperative attacks using spoof replicas created from latent fingerprints left on surfaces. The study evaluated these spoofs with software-based liveness detectors using LBP and Binary Statistical Image Features. The results showed that latent spoofs could bypass existing systems at low false-rejection rates. The study also found that spoof quality did not always predict attack success.

Gomez-Barrero et al. [7] proposed a multi-modal presentation attack detection method using several imaging techniques processed through parallel ad hoc classifiers, combining individual outputs with SVM classifiers and a weighted score-fusion strategy, achieving a 2.7% detection error rate across known and unknown attack materials.

Nikam and Agarwal [15] proposed a texture-based liveness detection method using a single fingerprint image, extracting LBP histograms and wavelet-energy features classified with a hybrid classifier, and achieved up to

96.71% accuracy, demonstrating strong effectiveness without additional hardware and highlighting the value of texture-based features for spoof detection.

Table 1 summarizes these studies. Taken together, the reviewed literature shows that deep-learning-based methods (e.g., [9], [13]) achieve very low error rates but at higher computational and data cost, while classical machine-learning approaches using handcrafted texture features (e.g., [10], [11], [15]) remain competitive and considerably cheaper to train and deploy. However, few of these classical ML studies directly benchmark more than one lightweight classifier side by side on the same feature representation and the same combination of datasets. This study addresses that gap by extracting a common LBP feature representation from a combined FVC2000/SOCOfing dataset and directly comparing SVM and KNN under identical preprocessing and evaluation conditions, in order to identify which classical classifier is better suited to this feature space.

Table 1. Summary of literature review.

S/N	Author	Method	Key Result	Reported Limitation
1	Oduah et al. [5]	IR sensor + ESP32-CAM contactless scanning	97.51% correlation at 3.5 cm	3D-to-2D reconstruction errors from skin deformation
2	Singh et al. [16]	Survey of traditional vs. anomaly-based PAD	Anomaly detection flags unknown attacks	Hard to define "normal" profiles
3	Kauba et al. [6]	Cooperative molds (silicone, latex, wood glue) on COTS scanners	Most commercial scanners vulnerable	Limited to specific commercial hardware
4	Pereira et al. [13]	Adversarial learning with CNN/MLP for PAI-invariance	APCER reduced to 0.81%	Most literature is binary, single-attack-type only
5	Purnapatra et al. [9]	DenseNet-121 / NasNetMobile on noncontact deepfake dataset	0.14% APCER, 0.18% BPCER	Few public noncontact spoof datasets
	Husseis et al. [11]	First-order thermal/Optical video statistics + SVM	18.1% BPCER (thermal), 19.5% (optical)	Small dataset (11 subjects)
6	Adam and Sathesh [10]	Ridgelet transform features + SVM ensemble	90.34% accuracy	Inaccurate on unknown spoof materials
7	Marasco et al. [14]	Pairwise correlation of match/Quality scores on latent spoofs	Latent spoofs bypass systems at low FRR	Tied to specific commercial matchers/materials
8	M. Jomaa et al. [12]	EfficientNet + 1D/2D CNN fusion of fingerprint and ECG	Fusion outperforms single-modality	Lack of public concurrent multimodal datasets
9	Nikam and Agarwal [15]	LBP + wavelet energy, hybrid classifier	Up to 96.71% accuracy	Texture-only; hardware-free but sensor-specific
10	Webster et al. [8]	Human saliency maps + CYBORG loss	Top benchmark on LivDet-2021	Constrained by scarce high-quality attack samples
11	Gomez-Barrero et al. [7]	Multi-imaging + parallel classifiers with SVM fusion	2.7% detection error rate	Requires multiple imaging modalities
12	Oduah et al. [5]	IR sensor + ESP32-CAM contactless scanning	97.51% correlation at 3.5 cm	3D-to-2D reconstruction errors from skin deformation

3 | Methodology

This study follows a five-stage pipeline:

I. Fingerprint image data collection.

II. Preprocessing.

III. LBP feature extraction.

IV. Model training for SVM and KNN.

V. Evaluation against standard classification metrics.

3.1|Data Collection

Two datasets were used and evaluated independently in this study. FVC2000 (Springer supplementary archive 74034_3_En_4_MOESM1_ESM) contributed 3,520 grayscale .tif images across four sub-databases, each comprising 110 fingers * 8 impressions. Of these, 2,640 genuine images from three sensor-based sub-databases - DB1 (optical), DB2 (capacitive), and DB3 (optical) while the remaining 880 images, which were synthetically generated using SFinGe (DB4), formed the non-genuine class. Following the standard FVC benchmark protocol, fingers 101-110 (320 images) formed the training set and fingers 1-100 (3200 images) formed the test set. The SOCOFing dataset was obtained from the Kaggle repository ruizgata/socofing and comprises 55,270 .bmp images, of which 6,000 are genuine (600 subjects, 10 fingers each, both hands) and 49,270 altered images generated using the STRANGE toolbox through the operations obliteration, central rotation, z-cut and applied at three increasing severities to give 17,931 images for Easy, 17,067 for Medium, and 14,272 images for Hard.

3.2|Data Processing

The following preprocessing steps were applied to the fingerprint images:

I. Image resizing: All fingerprint images were resized to a constant 300 × 300 pixels using bicubic interpolation to ensure consistent input dimensionality across both datasets and all sensor sources.

II. Contrast enhancement: Contrast Limited Adaptive Histogram Equalization (CLAHE); clip limit = 2.0, tile grid size = 8 * 8) was applied to each resized image to enhance ridge-valley contrast before feature extraction, as illustrated in *Fig. 1*.

III. Normalization: pixel intensities were min-max normalized to the standard [0. 255] range following CLAHE, yielding a consistently scaled 8-bit image for downstream feature extraction.



Fig. 1. Sample image of both noisy and denoised fingerprints.

3.3|Feature Extraction

Ridge-pattern and texture information were extracted from the preprocessed fingerprint images to differentiate genuine from fake samples. LBP, a texture-based feature extraction technique widely used in FPAD, was adopted for this purpose. LBP features are computed by comparing each pixel with its circular

neighborhood and assigning a binary value based on the intensity difference relative to the centre pixel; the resulting binary values are concatenated into an LBP code per pixel, and a histogram of all LBP codes across the image, computed block-wise and concatenated, forms the final feature vector. Scikit-image was used to convert the raw fingerprint pixels into a 2D array of texture codes, and NumPy was used to convert this output into normalized per-block histograms.

The LBP descriptor was configured with a circular neighborhood of radius R = 1 and P = 8 sampling points, using the rotation-invariant uniform pattern variant, which maps each neighborhood to one of P + 2 = 10 distinct codes. Each 300 * 300 preprocessed image was partitioned into a non-overlapping 8 * 8 grid (64 spatial blocks) prior to histogramming; a 10-bin histogram was computed per block and L1-normalized independently, and the 64 block histograms were concatenated into a single feature vector of dimension 8 * 8 * 10 = 640. Figure 2 shows a representative processed fingerprint image alongside its corresponding LBP code map.



a.



b.

Fig. 2. LBP transformation of a representative fingerprint sample; a. preprocessed fingerprint image, and b. corresponding LBP code map.

3.4 | Model Training

The feature-engineered dataset was split into training and test sets in an 80:20 ratio, a split ratio commonly used in ML classification tasks [10]. The partitioning procedure was a single hold-out split implemented as a

GroupShuffleSplit (scikit-learn, random_state = 42) grouped by subject identified and extracted from each filename, ensuring no subject's fingerprints appeared in both partitions. The split was therefore subject-disjoint rather than class-stratified. Class balance in the training set was enforced by randomly undersampling the majority (altered) class to match the genuine count, while the test set retained its natural, imbalanced class distribution to reflect real-world deployment conditions.

SVM is a supervised ML model used here to distinguish live fingerprints from fake presentations. The SVM decision function assigns each fingerprint sample to a class based on its position relative to an optimal separating hyperplane, as given in *Eq. (1)*:

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b\right), \quad \textbf{(1)}$$

n where f(x) is the decision function, α_i are the Lagrange multipliers, y_i is the class label (+1 for live, −1 for fake), $K(x_i, x)$ is the kernel function measuring similarity between data points x_i and x, and b is the bias term. The decision boundary is defined by f(x) = 0, i.e., the separating hyperplane between the two classes.

This research used a Radial Basis Function. The regularization parameter C and kernel parameter were tuned via GridSearchCV with 5-fold stratified cross-validation on the training set (scoring: ROC-AUC), searching C ε {0.1,1,10,100} and γ ε {'scale', 'auto',0.001,0.01,0.1}, jointly with class_weight ε {None, 'balanced' }. The selected optimal configuration was consistent across all four experimental conditions in one respect: γ = 0.001 and class_weight = None were optimal throughout, while C varied by alteration severity (C=100 for Easy and Combined; C = 10 for Medium and Hard).

KNN is a non-parametric supervised learning algorithm that stores training samples without an explicit training phase. It classifies a new sample by computing the distance between that sample and all training samples, identifying the K nearest neighbors, and assigning the majority class label among them. The Euclidean distance metric, defined in *Eq. (2)*, was used for distance computation:

$$d(x_i, x) = \sqrt{\sum_{j=1}^{m} (x_{ij} - x_j)^2}, \quad \textbf{(2)}$$

where x_{ij} is the j-th feature of the i-th training instance, and x_j is the j-th feature of the new instance x. The predicted class is determined by majority vote among the k nearest neighbors, as given in *Eq. (3)*:

$$Y = \text{argmax}_k \sum_{i=1}^{k} I(y_i = c), \quad \textbf{(3)}$$

where Y is the predicted class, I(·) is the indicator function, y_i is the class label of the i-th nearest neighbor, and c is a candidate class label.

K and the distance-weighting scheme were likewise tuned via GridSearchCV with 5-fold stratified cross-validation, searching odd values $K \varepsilon \{3,5,7,9,11,13,15\}$ (avoiding tied votes) with weights ε {'uniform', 'distance'} using a fixed Euclidean metric. K=15 with uniform weighting performed best across all four conditions. This finding suggests that a larger neighbourhood was more effective in the 640-dimensional feature space.

3.5 | Evaluation Metrics

Both models were evaluated using accuracy, precision, recall, F1-score, and ROC-AUC, supplemented by Matthews Correlation Coefficient (MCC) and balanced accuracy to account for the class imbalance present in the test set (genuine samples are consistently outnumbered by altered samples at test time). Accuracy measures the proportion of correctly classified samples; precision measures the proportion of correct positive

predictions among all positive predictions; recall measures the proportion of correctly identified positive samples; and F1-score provides a balanced measure of precision and recall. Because accuracy alone is a poor indicator of performance under class imbalance, precision, recall, and F1-score were computed per class (genuine vs altered) and macro-averaged (unweighted mean across classes) rather than as a single value. Following the International Standard ISO/IEC 30107-3 for biometric presentation attack detection testing and reporting, the Attack Presentation Classification Error Rate (APCER), which expresses the proportion of attack presentations that were incorrectly accepted as bona fide, and the BPCER, which expresses the proportion of bona fide presentations that were incorrectly rejected as attacks, were computed. *Eqs. (4)* and (*5)* respectively show how to calculate their values, where FP, TN, FN, and TP represent False Positives, True Negatives, False Negatives, and True Positives, respectively.

$$\text{APCER} = \frac{\text{FP} * 100}{\text{FP} + \text{TN}}, \tag{4}$$

$$\text{BPCER} = \frac{\text{FN} * 100}{\text{FN} + \text{TP}}. \tag{5}$$

3.6 | Experimental Setup

All experiments were implemented in Python 3.13.7, using scikit-learn 1.7.1, scikit-image 0.26.0, NumPy 2.2.6, OpenCV (opencv-python) 5.0.0, pandas 2.3.1, and joblib 1.6.0. Experiments were run on a Dell Latitude 7410 with an Intel Core i7-10610U CPU (4 physical / 8 logical cores, 1.80GHz) with 16 GB of RAM installed on a Windows 11 Pro (64-bit) computer. No GPU was used or required for this research.

Hyperparameters for both classifiers were selected via grid search combined with 5-fold stratified cross-validation on the training data, with the best-performing configuration determined by ROC-AUC score. For SVM, the search covered four values of the regularization parameter C (0.1, 1, 10, and 100), five values of the kernel parameter gamma ('scale', 'auto', 0.001, 0.01, and 0.1), a fixed RBF kernel, and two class-weighting options (none or balanced), yielding 40 candidate configurations. For KNN, the search covered seven values of K (3,5,7,9,11,13, and 15), two weighting schemes (uniform or distance-based), and the Euclidean distance metric, giving 14 candidate configurations.

4 | Implementation and Results

4.1 | Model Performance

Table 2 shows the performance of both classifiers on the held-out test set for each dataset. For SOCOFing, results are reported for the Easy, Medium, Hard, and Combined conditions, while FVC2000 follows the standard Set A/B protocol.

Table 2. Model performance on the held-out test sets.

Dataset	Model	Alteration Tier	Accuracy	Precision	Recall	F1-Score
FVC2000	SVM	-	99.6%	99.6%[a]	99.6%[a]	99.6%[a]
FVC2000	KNN	-	99.9%	99.9%[a]	99.9%[a]	99.9%[a]
SOCOFing	SVM	Easy	80.2%	75.2% [b]	81.0% [b]	76.7% [b]
SOCOFing	SVM	Medium	94.4%	91.6% [b]	94.7% [b]	93.0% [b]
SOCOFing	SVM	Hard	97.6%	96.6% [b]	97.6% [b]	97.1% [b]
SOCOFing	SVM	Combined	86.8%	71.4% [b]	87.1% [b]	75.6% [b]
SOCOFing	KNN	Easy	52.1%	61.1% [b]	63.3% [b]	51.7% [b]
SOCOFing	KNN	Medium	77.4%	75.2% [b]	82.4% [b]	75.3% [b]
SOCOFing	KNN	Hard	88.7%	85.9% [b]	89.9% [b]	87.3% [b]
SOCOFing	KNN	Combined	62.0%	59.9%[b]	75.3% [b]	53.9% [b]

Note: [a] Weighted average, [b] Macro average (unweighted measure across genuine and altered classes), used for SOCOFing to account for test-set class imbalance.

Fig. 3 presents a comparative view of both classifiers across the four standard metrics on the SOCOFing Combined tier, which uses the full altered fingerprint dataset and serves as the primary basis for the conclusions of this study. On FVC2000, both classifiers achieved near-perfect performance across all evaluation metrics. KNN recorded slightly higher accuracy, precision, recall, and F1-score than SVM, while both classifiers achieved a ROC-AUC of 1.00. ON SOCOFing, the ranking reverses: SVM outperforms KNN on every metric and every alteration tier, with the gap widening as class separability narrows.

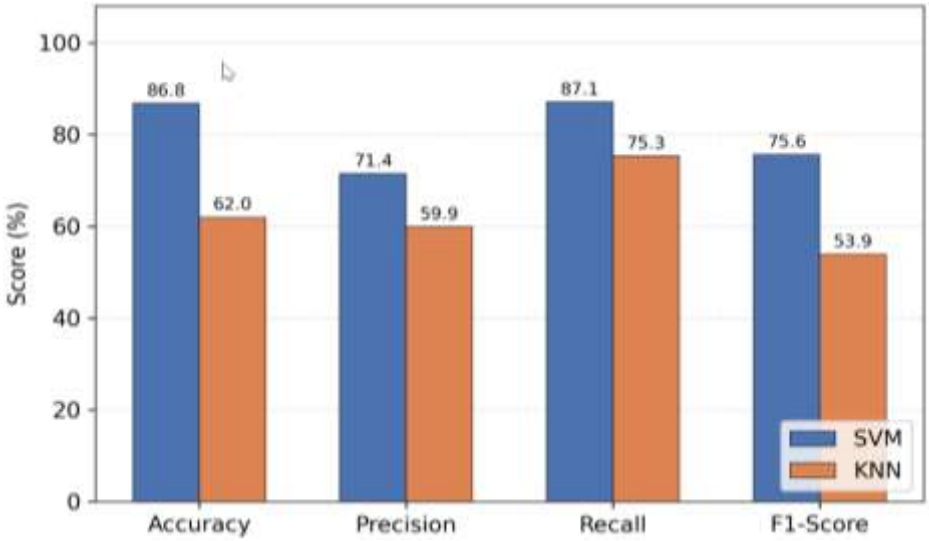

Fig. 3. Comparative analysis of SVM and KNN Across four standard evaluation metrics (SOCOFing, combined tier).

Evidence from *Table 2* and *Fig. 3* shows SVM outperforms KNN on every metric and every alteration tier on the SOCOFing dataset.

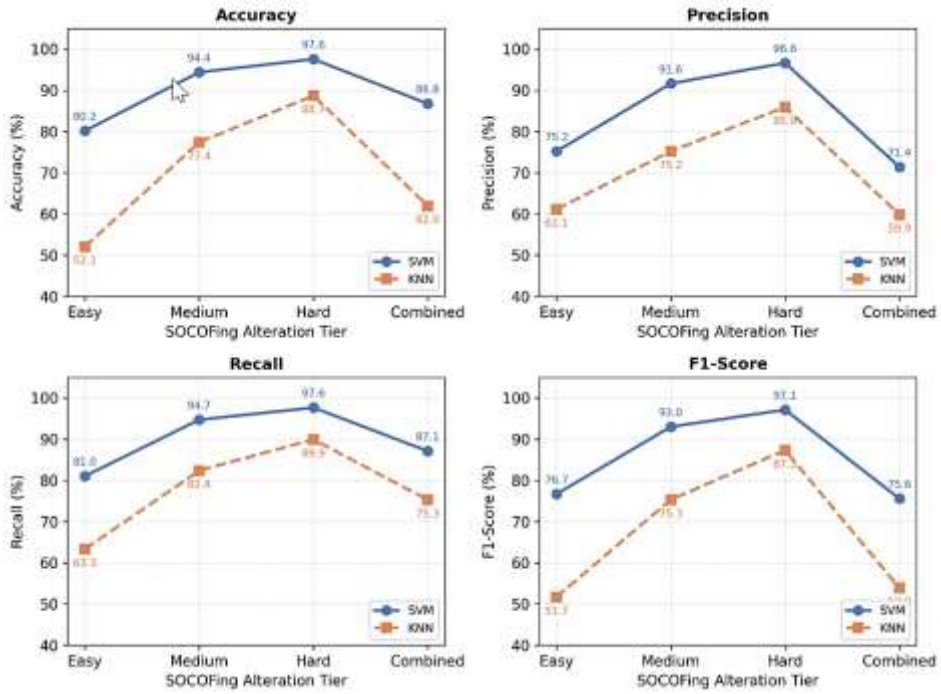

Fig. 4. Comparative analysis of SVM and KNN across four standard evaluation metrics.

4.2 | Confusion Matrix and Error Analysis

Figs. 5 and *6* present the confusion matrices across SOCOFing alteration tiers for SVM and KNN, respectively. *Table 3* reports the corresponding APCER and BPCER for the SOCOFing conditions, following the reporting convention adopted by Husseis et al. [11]and Pereira et al. [13] to facilitate comparison with that prior work.

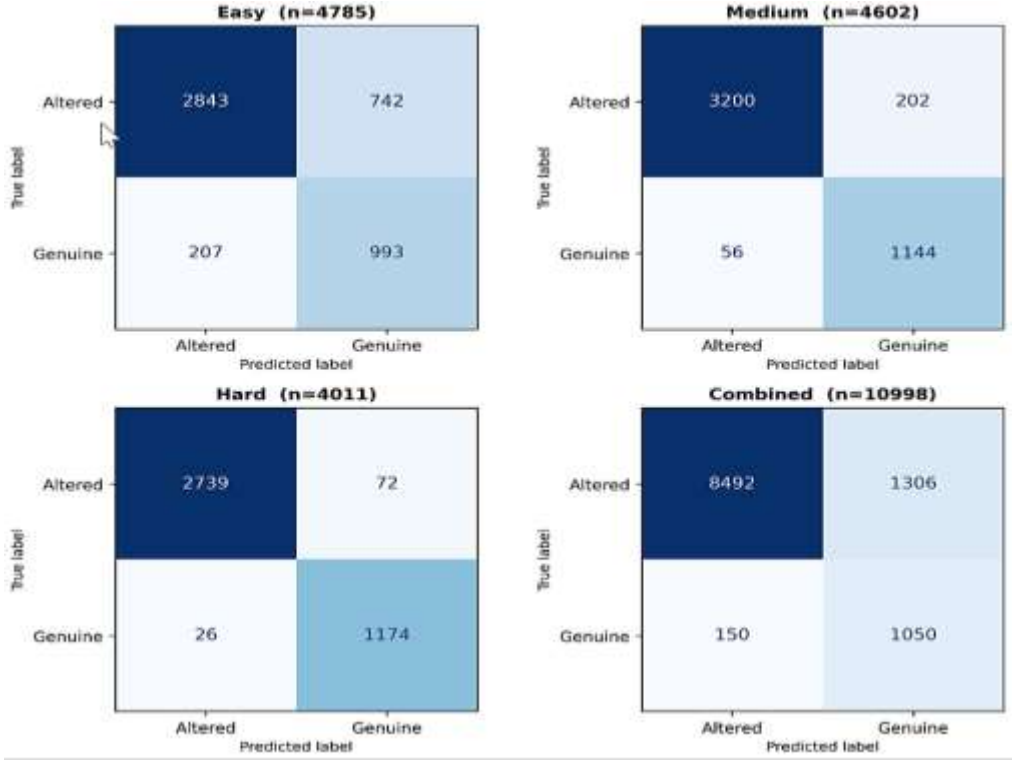

Fig. 5. Confusion matrices for SVM across SOCOFing alteration tiers.

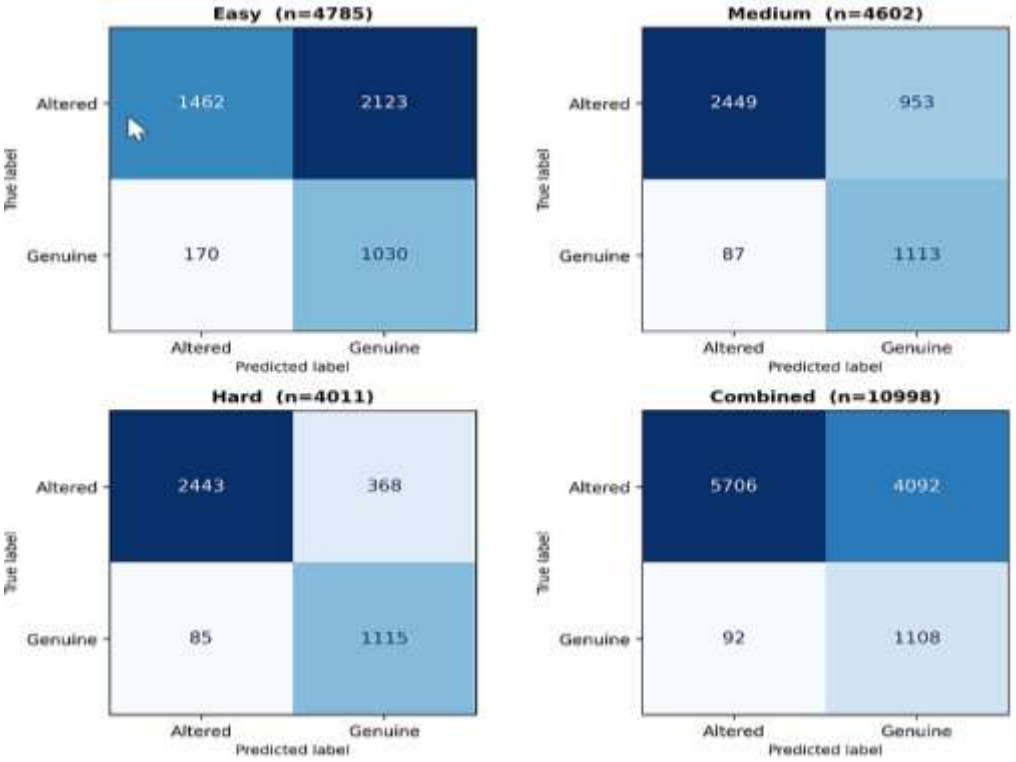

Fig. 6. Confusion matrices for KNN across SOCOFing alteration tiers.

Table 3. PAD error rates (%) for SVM and KNN across SOCOFing alteration tiers.

Model	Easy APCER	Easy BPCER	Medium APCER	Medium BPCER	Hard APCER	Hard BPCER	Combined APCER	Combined BPCER
SVM	20.70	17.25	5.94	4.67	2.56	2.17	13.33	12.50
KNN	59.22	14.17	28.01	7.25	13.09	7.08	41.76	7.67

SVM produced a substantially lower and more balanced error profile than KNN across every alteration tier. On the Combined condition, SVM's APCER (13.33%) is of comparable order of magnitude to the 18.1% and 19.5% BPCER reported by Husseis et al. [11] for an SVM classifier evaluated on dynamic-capture presentation attack data, while KNN's APCER (41.76%) considerably exceeds the error rates reported in either comparator study. Error rates in this study were computed at each classifier's default decision threshold, whereas [11], [13] reported BPCER at a fixed operating point calibrated to APCER=5%; the comparison above is therefore indicative of relative order of magnitude rather than a threshold-matched benchmark, and a threshold-matched benchmark, and a threshold-calibrated re-evaluation is identified as a direction for future work.

KNN's error profile is characterised by a pronounced asymmetry between APCER and BPCER across all four tiers. KNN's APCER exceeds its BPCER by a wide margin, most severely on Combined (41.76% vs 7.67%), indicating that KNN's dominant failure mode is accepting altered fingerprints as genuine rather than rejecting genuine fingerprints as altered. SVM shows a comparatively balanced APCER/BPCER profile at every tier.

For reference, the FVC2000 confusion matrices yield APCER=0.00% and BPCER=0.54% (SVM) and APCER = 0.00% and BPCER = 0.08% (KNN). Both classifiers admitted zero synthetic images as genuine; the only errors were a small number of genuine images rejected as synthetic.

4.3 | Computational Cost

Feature extraction (LBP feature extraction across the full image pool), model training, and per-sample inference time were measured for both classifiers on the hardware specified earlier, which is an Intel Core i7-10610U, 4 cores/8 threads, 1.80GHz, 15.7 GB RAM, no GPU. All three quantities were recorded within the same experimental run for each condition, ensuring that the reported values were obtained under consistent execution settings rather than from independently timed benchmark runs. Training time refers to the full GridSearchCV hyperparameter optimization procedure, which consists of 40 candidate configurations for the SVM evaluated using 5-fold cross-validation, together with the one-time refitting step required to obtain calibrated probability estimates. For KNN, training time similarly includes the complete search over 14 candidate configurations under the same 5-fold cross-validation protocol. *Table 4* presents the resulting computational costs across all five experimental conditions.

Table 4. Computational cost by experimental condition.

Condition	Model	Feature Extraction in Minutes	Training Time Minutes	Total Pipeline Minutes	Inference (ms/Sample)	Test Set Size
FVC2000	SVM	2.9	0.10	3.0	0.339	3,200
FVC2000	KNN	2.9	0.07	3.0	0.072	3,200
SOCOFing easy	SVM	24.2	77.7	102.4	6.265	4,785
SOCOFing easy	KNN	17.9	1.4	19.4	1.727	4,785
SOCOFing Medium	SVM	9.4	52.3	62.0	3.573	4,602
SOCOFing Medium	KNN	16.2	1.1	17.4	1.316	4,602

Table 4. Continued.

Condition	Model	Feature Extraction in Minutes	Training Time Minutes	Total Pipeline Minutes	Inference (ms/Sample)	Test Set Size
SOCOFing hard	SVM	8.0	46.7	54.9	2.249	4,011
SOCOFing hard	KNN	18.0	2.1	20.2	2.342	4,011
SOCOFing Combined	SVM	21.1	55.5	77.4	4.515	10,998
SOCOFing Combined	KNN	76.9	1.8	79.0	1.855	10,998

The computational cost varied considerably between the two classifiers. On SOCOFing, KNN training took 1.1 to 2.1 minutes, while SVM training took 46.7 to 77.7 minutes. The higher SVM cost resulted from the evaluation of 40 RBF-kernel configurations with 5-fold cross-validation. Feature extraction also showed variation across the SOCOFing condition. The largest difference occurred for the Combined condition, where extraction took 76.9 minutes for KNN and 21.1 minutes for SVM.

Inference was much faster than feature extraction and training; KNN required 1.316 to 2.342 milliseconds per sample on SOCOFing, compared with 2.249 to 6.265 milliseconds for SVM. On FVC2000, inference took 0.072 milliseconds for KNN and 0.339 milliseconds for SVM. Overall, feature extraction was the main cost for KNN, while SVM required more time for training. In contrast, inference was fast for both classifiers, taking only a few milliseconds per sample.

4.4 | Comparison with Prior Work

This study achieved a peak SVM accuracy of 97.6% on the Hard-alteration tier of SOCOFing, which is the most heavily distorted fingerprints in the dataset, with an aggregated accuracy of 86.8% across all severities combined. *Table 5* situates both figures against prior FPAD literature. Direct comparison is limited by differences in dataset, attack type, and evaluation protocol across different studies.

Table 5. Computational cost by experimental condition.

Study	Approach	Reported Accuracy / Error Rate
This study – SVM (SOCOFing Hard, best case)	LBP + SVM	97.6% accuracy
This study -SVM (SOCOFing Combined, aggregate)	LBP + SVM	86.8% accuracy
This study – KNN (SOCOFing Combined, aggregate)	LBP + KNN	62.0% accuracy
Nikam and Agarwal [15]	LBP + wavelet energy, hybrid classifier	96.7% accuracy
Adam and Sathesh [10]	Ridgelet transform + SVM ensemble	90.3% accuracy
Gomez-Barrero et al. [7]	Multi-imaging + SVM score fusion	2.7% detection error rate
Husseis et al. [11]	Thermal/Optical video statistics + SVM	18.1% / 19.5% BPCER
Purnapatra et al. [9]	DenseNet-121 (CNN)	0.14% APCER

4.5 | Discussion

The performance of the two classifiers varied for the two datasets. KNN achieved 99.9% accuracy, and SVM achieved 99.6% on the FVC2000 dataset. Both classifiers recorded a ROC-AUC of 1.00. The difference in accuracy was small. FVC2000 dataset contained 320 training images and 3,200 testing images. For the SOCOFing dataset, SVM achieved higher accuracy than KNN under all four conditions. On the combined SOCOFing dataset, SVM achieved an accuracy of 86.8%, compared with 62.0% for KNN. The error rates

also showed a clear difference between the classifiers. KNN recorded an APCER of 41.8%. This result implies that 41.8% of altered fingerprints were classified as genuine. Its BPCER was 7.7%. SVM recorded an APCER of 13.3% and a BPCER of 12.5%. Thus, SVM produced fewer false acceptances of altered fingerprints, while KNN produced fewer false rejections of genuine fingerprints.

The better performance of SVM on SOCOFing agrees with earlier studies that used SVM for FPAD, which include [7], [10], [11]. Both classifiers achieved very high accuracy, with KNN recording 99.9% and SVM 99.6%. The use of LBP features is also consistent with the [15] study.

This study has some limitations. The final results were obtained from a single train/test split rather than repeated cross-validation, so the reported metrics may vary slightly with a different split. However, the subject-disjoint splitting strategy was used throughout the experiments to prevent fingerprints from the same subject from appearing in both the training and testing sets. Neither classifier was tested on alteration types that were absent from the training data. Therefore, generalization to completely unseen attacks remains unverified. The evaluation of the four SOCOFing alteration severities provides information on how classifier performance changes across different attack conditions.

Overall, SVM achieved better results than KNN across the SOCOFing conditions and had an accuracy of 99.6% on the FVC2000 dataset. The two classifiers also recorded low inference times. Future studies should benchmark these classical ML models against DL architectures and extend the SVM kernel search to include more kernel types and configurations. Further studies should also test generalization under a consistent threat model. For example, a model can be trained on one SOCOFing alteration severity and tested on an unseen severity or an alteration method outside the STRANGE toolbox. Comparisons between different attack types, such as synthetic image generation and digital alteration of a genuine fingerprint, should not be used alone to make generalization claims because they represent different attack settings. Repeated subject-disjoint splits can also be used to provide a more reliable evaluation of the models.

5 | Conclusion

In conclusion, this study evaluated SVM and KNN classifiers using LBP features for FPAD on both FVC2000 and SOCOFing datasets. KNN and SVM achieved accuracies of 99.9% and 99.6%, respectively, on the FVC2000 dataset, while on SOCOFing, SVM achieved higher accuracy under all four conditions, with an accuracy of 86.8% on the combined set, compared with 62.0% for KNN. Both classifiers also had low inference times. These results indicate that classifier performance varies with the dataset and attack conditions. Future work should evaluate generalization to unseen attack types and alteration severities and benchmark these classical ML models against deep-learning architectures under matched conditions.

Acknowledgments

Acknowledgements enable you to thank all those who have helped in carrying out the research. Careful thought needs to be given concerning those whose help should be acknowledged and in what order. The general advice is to express your appreciation in a concise manner and to avoid strong emotive language.

Author Contributaion

Conceptualization, I. O.; methodology, I. O.; data curation, I. O.; software, I. O; formal analysis, I. O.; investigation, I. O.; resources, I. O.; validation, I. O.; visualization, I. O.; writing - original draft, I. O.; writing - review and editing, I. O. All authors have read and agreed to the published version of the manuscript.

Funding

This study received no funding.

Data Availability

Implementation data can be found at https://github.com/ifeoseni/Fingerprint-Presentation-Attack-Detection. The dataset used is available in the GitHub repository. The direct source link to the FVC2000 dataset is https://static-content.springer.com/esm/chp%3A10.1007%2F978-3-030-83624-5_4/MediaObjects/74034_3_En_4_MOESM1_ESM.zip, while that of SOCOFing is https://www.kaggle.com/datasets/ruizgara/socofing.

Conflicts of Interest

The authors declare no conflict of interest.

Consent for Publication

The author confirms consent for the publication of this work.

Ethics Approval and Consent to Participate

This article does not contain any studies with human participants performed by the author.

References

[1] Szymkowski, M., & Saeed, K. (2025). Trimodal machine learning based biometrics system. *Scientific Reports*, *15*(1), 20949. https://doi.org/10.1038/s41598-025-06288-z

[2] Pooshideh, M., Beheshti, A., Qi, Y., Farhood, H., Simpson, M., Gatland, N., & Soltany, M. (2024). Presentation attack detection: A systematic literature review. *ACM Computing Surveys*, *57*(1), 1–32. https://doi.org/10.1145/3687264

[3] Plaid. (2024). *Presentation attack: What it is and how to detect it*. https://plaid.com/resources/fraud/presentation-attack/

[4] Thapa, N., Chaudhari, M., & Roy, K. (2023). Presentation attack detection: An analysis of spoofing in the wild (SiW) dataset using deep learning models. *Discover Artificial Intelligence*, *3*(1), 30. https://doi.org/10.1007/s44163-023-00077-1

[5] Oduah, U. I., Kevin, I. F., Oluwole, D. O., & Izunobi, J. U. (2021). Towards a high-precision contactless fingerprint scanner for biometric authentication. *Array*, *11*, 100083. https://doi.org/10.1016/j.array.2021.100083

[6] Kauba, C., Debiasi, L., & Uhl, A. (2020). *Enabling fingerprint presentation attacks: Fake fingerprint fabrication techniques and recognition performance*. https://doi.org/10.48550/arXiv.2012.00606

[7] Gomez-Barrero, M., Kolberg, J., & Busch, C. (2018). Towards fingerprint presentation attack detection based on short wave infrared imaging and spectral signatures. *Proceedings of the Norwegian information security conference (NISK 2018)*. https://www.marta-gomez-barrero.com/publication/gomez-barrero-fp-pad-swir-svm-nisk-2018/

[8] Webster, S., Scheirer, W., & Czajka, A. (2026). Psychophysically-guided training for fingerprint presentation attack detection. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, *8*(5), 707–717. https://doi.org/10.1109/TBIOM.2026.3664679

[9] Purnapatra, S., Miller-Lynch, C., Miner, S., Liu, Y., Bahmani, K., Dey, S., & Schuckers, S. (2023). Presentation attack detection with advanced cnn models for noncontact-based fingerprint systems. *2023 11th international workshop on biometrics and forensics (IWBF)* (pp. 1-6). IEEE. https://doi.org/10.1109/IWBF57495.2023.10157605

[10] Adam, E. E. B., & Sathesh, A. (2021). Evaluation of fingerprint liveness detection by machine learning approach-a systematic view. *Journal of Ismac*, *3*(01), 16–30. https://doi.org/10.36548/jismac.2021.1.002

[11] Husseis, A., Liu-Jimenez, J., Goicoechea-Telleria, I., & Sanchez-Reillo, R. (2020). Dynamic fingerprint statistics: Application in presentation attack detection. *IEEE Access*, *8*, 95594–95604. https://doi.org/10.1109/ACCESS.2020.2995829

[12] M. Jomaa, R., Mathkour, H., Bazi, Y., & Islam, M. S. (2020). End-to-end deep learning fusion of fingerprint and electrocardiogram signals for presentation attack detection. *Sensors*, *20*(7), 2085. https://doi.org/10.3390/s20072085

[13] Pereira, J. A., Sequeira, A. F., Pernes, D., & Cardoso, J. S. (2020). A robust fingerprint presentation attack detection method against unseen attacks through adversarial learning. *2020 international conference of the biometrics special interest group (biosig)* (pp. 1–5). IEEE. https://ieeexplore.ieee.org/abstract/document/9210972

[14] Marasco, E., Cando, S., Tang, L., Ghiani, L., & Marcialis, G. L. (2018). A look at non-cooperative presentation attacks in fingerprint systems. *2018 eighth international conference on image processing theory, tools and applications (IPTA)* (pp. 1-6). IEEE. https://doi.org/10.1109/IPTA.2018.8608133

[15] Nikam, S. B., & Agarwal, S. (2008). Local binary pattern and wavelet-based spoof fingerprint detection. *International Journal of Biometrics*, *1*(2), 141–159. https://doi.org/10.1504/IJBM.2008.020141

[16] Singh, J. M., Madhun, A., Li, G., & Ramachandra, R. (2020). A survey on unknown presentation attack detection for fingerprint. *International conference on intelligent technologies and applications* (pp. 189-202). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-71711-7_16