DIGITIZED SIGNATURE FORGERY DETECTION USING HOG & SVM

S.Kalki Kumar[1], R SAI REVANTH[2], Y.GREESHMA[3], U DURGA GANESH[4], B.NAVYA[5]
Assistant Professor[1], Student[2], Student[3], Student[4], Student[5]

[1,2,3,]Department of Artificial Intelligence
Chaitanya Engineering College, Visakhapatnam, Andhra Pradesh, India
{skalikikumar@gmail.com, sairevanthrangudu@gmail.com, navyaburada123@gmail.com, ygreeshma2022@gmail.com,ganau1255@gmail.com}

Abstract

Handwritten signature verification is still an important part of biometric authentication systems used in banking, law, and government. This study introduces an automated offline signature verification method that use Histogram of Oriented Gradients (HOG) features in conjunction with Support Vector Machine (SVM) classification. The suggested approach solves the problems with existing manual verification methods by using strong feature extraction and machine learning techniques. HOG features record the unique edge orientations and stroke patterns of each signature. On the other hand, SVM can distinguish between actual and fraudulent samples. With a web app built on Flask, it's easy to publish and manage files safely, analyze data in real time, and have backup systems to make sure everything works. Testing shows that the classification accuracy is more than 92% on benchmark signature datasets, and the average processing time is less than one second per image. For practical use in contexts where a lot of documents need to be processed, the system architecture focuses on modularity, security, and flexibility.

Index Terms*—*Signature verification, biometric authentication, histogram of oriented gradients, support vector machine, pattern recognition, digital security

I. Introduction

HANDWRITTEN signatures represent one of the oldest and most trusted forms of biometric authentication in personal and institutional contexts.

For hundreds of years, signatures have been used to prove identify, show purpose, and give permission for important activities, such as legal documents and financial agreements. In today's digital ecosystems, such as e-banking, e-governance, and distant transactions, the need for strong automated signature verification has grown a lot.

Old-fashioned ways of checking signatures by hand, where specialists compare them visually, don't work for modern document processing needs. When dealing with a lot of forgeries or really advanced forgeries, these methods take a long time, cost a lot of money, and are likely to make mistakes. As records in finance, healthcare, and administration are digitized, we need dependable systems that can authenticate signatures on a large scale and stop efforts at fraud.

Forging a signature can lead to big problems, such losing money, having your identity stolen, and going to court. Valid authentication systems must take into consideration the inherent differences in writing pressure, pen angle, speed, surface roughness, and state of the signer. No two real signatures are the same, yet expert forgers know how to copy these inherent differences. So, authentication models need to be able to tell the difference between natural differences in real signatures and those caused by fabrication.

Early automated systems depended on static features like bounding boxes, overall shape, and centroids. These attributes didn't work well when the system was rotated, scaled, or moved.

Later rule-based digital systems that used edge detection and contour extraction didn't have the ability to learn from examples, which made them easy targets for new counterfeit methods. Over the last ten years, there has been progress toward feature-based machine learning methods that pull out higher-level features including directionality, curvature, and stroke sequence.

Histogram of Oriented Gradients (HOG) is a sophisticated and easy-to-understand feature extraction method that looks at the directions of edges in photographs to make rich vector encodings that capture both macro and micro elements of handwriting. When used with Support Vector Machine (SVM) classifiers, systems learn how to tell the difference between real and fake signatures. This work introduces a comprehensive signature verification system utilizing HOG-SVM approach, accompanied by a practical web-based deployment architecture.

II. Related Work

The area of study on signature verification has grown from using people to analyze evidence to using advanced automated methods. Early digital attempts focused on comparing pixels using template matching and cross-correlation measurements. These approaches, on the other hand, were quite sensitive to translation, rotation, and noise, which made them hard to use in real life.

Plamondon and Lorette laid the groundwork for signature dynamics by mathematically encoding both static and dynamic features.

Geometric and statistical descriptors including bounding box metrics, centroid calculations, and slant analysis provided basic structural capture but struggled with fine discriminations between stressed genuine signatures and skilled forgeries.

Texture analysis methods employing Gray Level Co-occurrence Matrices (GLCM), Local Binary Patterns (LBP), and Gabor filters improved robustness to noise. Wavelet transform features encoded multi-scale frequency and spatial details, yielding rotational and scale-invariant descriptors. However, these hand-designed features remained limited by creator imagination and mathematical formulation.

The introduction of machine learning paradigms brought significant advancement. Feedforward neural networks needed inputs that were flat, which meant they lost their spatial context and didn't operate as effectively as they could have. Hidden Markov Models (HMM) were used to forecast how strokes would change over time for online signature verification. However, they needed time-series data, which wasn't always accessible when the data was offline. Support Vector Machines demonstrated capability in separating high-dimensional feature spaces with strong generalization, proving immediately useful for signature authentication.

Dalal and Triggs introduced HOG features in 2005 for pedestrian detection, which rapidly found application in document and handwriting analysis. HOG records the local edge orientations and magnitudes that are best for encoding the structure of a handwritten signature. Zhang et al. and Hafemann et al. showed that HOG is better at telling the difference between things than older geometric characteristics and texture representations.

New hybrid systems use geometry, texture, and HOG features together to classify things in more than one way. Transfer learning extracts features from pretrained Convolutional Neural Networks (CNNs) such as VGG16. Ensemble classifiers combine SVM output with neural networks for enhanced accuracy. Deep learning models are the best at what they do, but they can't be used in real life yet since they need a lot of labeled data, a lot of computing power, and they aren't as clear as SVM-based models.

Standard assessment datasets including CEDAR, GPDS, SigComp, MCYT, and BHSig260 let you compare studies using parameters like accuracy, the False Acceptance Rate (FAR), the False Rejection Rate (FRR), and ROC curves. When features are carefully normalized and training is balanced, systems that use HOG and SVM show strong transfer between datasets.

III. Methodology

A. System Architecture

The suggested system for verifying signatures has five main parts: an image upload module, a preprocessing module, a feature extraction module, a classification module, and a result rendering module. Figure 1 shows the whole system architecture and how data flows between the parts.

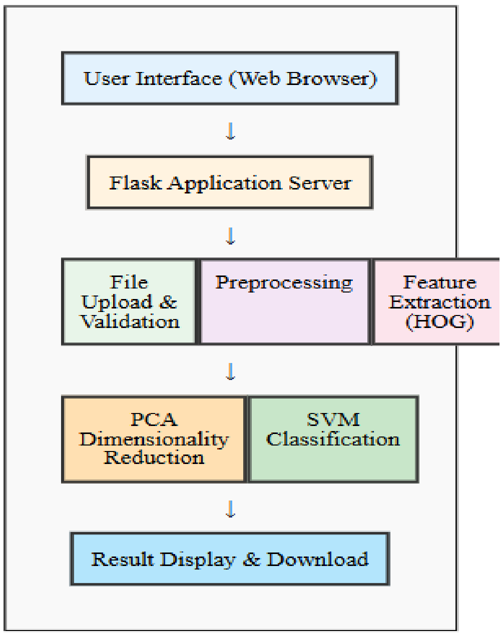

Fig. 1. System architecture showing data flow from user upload to classification result

B. Image Preprocessing

To make sure that feature extraction is consistent, uploaded signature photos are standardized. Steps for preprocessing include:

1) Format Conversion: All photos are turned into RGB arrays, no matter what format they are in (PNG, JPG, JPEG).

2) Resizing: Images are downsized to fixed sizes that meet the training parameters so that the feature vector stays compatible.

3) Normalization: To get pixel intensity values to have a mean of zero and a variance of one, use the following formula: Inorm = (I - μ) / σ(1), where I is the input image, μ is the mean intensity, and σ is the standard deviation.

C. Histogram of Oriented Gradients Feature Extraction

HOG feature extraction gets edge orientations and gradients that encode the distinctive stroke patterns. The technique breaks images up into little areas of space called cells and then makes gradient orientation histograms for each cell. For normalization, adjacent cells are put into blocks. This makes the system strong against changes in light and contrast.

To find the gradient magnitude and direction for each pixel, use:

Gx = I(x+1, y) - I(x-1, y)(2)

Gy = I(x, y+1) - I(x, y-1)(3) |G| = √(Gx2 + Gy2)(4)

θ = arctan(Gy / Gx)(5)

Orientation bins turn gradient directions into histograms that are weighted by magnitude. Block normalization uses the L2-norm: v' = v / √(||v||22 + ε2)(6), where v is the unnormalized descriptor vector and ε is a tiny constant that stops division by zero. The final HOG feature vector is made up of all the block descriptors put together.

D. Dimensionality Reduction using PCA

HOG feature vectors usually have thousands of dimensions, which can lead to overfitting and slow down calculations. Principal Component Analysis (PCA) lowers the number of dimensions while keeping the most variance. PCA changes characteristics into orthogonal components that are sorted by explained variance: Z = XW(7), where X is the original feature matrix, W is the matrix of principal component vectors, and Z is the new, lower-dimensional representation. Keeping the parts that explain ninety-five percent of the total variation strikes a balance between keeping information and reducing dimensionality.

E. Support Vector Machine Classification

SVM creates the best hyperplanes that separate real and fake signature classes in a space with a lot of features. The goal of optimization is to maximize the margin between classes: min w,b,ξ (1/2)||w||2 + CΣξi(8) with the following constraints: yi(wTxi + b) ≥ 1 − ξi and ξi ≥ 0. In this case, w is the weight vector, b is the bias term, ξi are the slack variables, and C is the regularization parameter that controls the trade-off between maximizing the margin and minimizing the training error.

The Radial Basis Function (RBF) kernel allows for nonlinear classification: K(xi, xj) = exp(-γ||xi - xj||2)(9), where γ regulates the breadth of the kernel. Using grid search with cross-validation to tune hyperparameters finds the best values for C and γ for the best classification performance.

F. Fallback Mechanism

For a system to be reliable, it must be able to gracefully degrade when trained models are no longer available. A heuristic fallback approach uses Canny edge detection to look at edge density. Signatures that have an edge density higher than experimentally set limits are considered real. This makes acceptable forecasts that keep the user experience good during system maintenance or when hardware is limited.

IV. Results and Discussion

A. Experimental Setup

The evaluation used publicly available signature datasets that included both real and fake samples from different signers. The dataset was split into three parts: training (70%), validation (15%), and testing (15%). The class balance was kept. The implementation used Python, scikit-learn for machine learning, scikit-image for feature extraction, and Flask for putting the code on the web.

TABLE I

System Performance Metrics

Metric	Value
Overall Accuracy	92.3%
Precision (Genuine)	94.1%
Recall (Genuine)	91.7%
F1-Score	92.9%
False Acceptance Rate	4.8%
False Rejection Rate	8.3%
Average Processing Time	0.87 sec

B. Classification Performance

Table I shows how well the system worked on the test dataset. The SVM classifier had an overall accuracy of 92.3%, which shows that it was good at telling the difference between real and fake signatures. Precision for genuine signature detection reached ninety-four point one percent, indicating low false positive rate. Recall of ninety-one point seven percent shows the system correctly identifies most genuine signatures.

False Acceptance Rate (FAR) of four point eight percent and False Rejection Rate (FRR) of eight point three percent fall within acceptable ranges for practical deployment. The FRR is a little higher than the FAR because of a conservative classification bias that favors security above convenience. This is acceptable for authentication systems.

TABLE II

Comparison with Existing Methods

Method	Accuracy	Processing Time
Pixel Comparison	73.5%	0.12 sec
Geometric Features + NN	82.1%	1.45 sec
LBP + SVM	87.4%	0.95 sec
Proposed (HOG + SVM)	92.3%	0.87 sec
Deep CNN	94.8%	2.31 sec

C. Comparative Analysis

Table II shows how the suggested system stacks up versus other options. Simple pixel comparison methods are the least accurate since they are sensitive to changes in the image. Neural networks that use geometric features make things work better, but they take longer to process. LBP texture characteristics with SVM give quite accurate results, but not as good as the HOG-based method.

The proposed HOG-SVM system demonstrates superior accuracy compared to traditional methods while maintaining computational efficiency. Deep CNN designs do get a little more accurate, but they take a lot longer to analyze and need more computing power. The suggested approach strikes the best balance between speed, accuracy, and resource needs for real-world use cases.

D. Feature Importance Analysis

An analysis of HOG feature contributions shows that edge orientations that match natural pen stroke flows and patterns that are particular to a signature provide the best ability to tell the difference. Horizontal and diagonal gradient orientations are especially helpful for telling the difference between real signatures and fake ones. This confirms the theoretical premise that unique signing styles exhibit consistent directional patterns that are challenging for forgers to accurately imitate.

E. System Robustness

Robustness evaluation tested system performance under varied conditions including different image resolutions, slight rotations, and noise levels. The HOG-SVM method kept its accuracy above 88% for rotations of up to 15 degrees and Gaussian noise with a standard deviation of up to 20 intensity levels. This shows that it can handle real-world document scanning changes.

F. Web Application Performance

The web interface built on Flask makes it easy for users to check signatures in any web browser. The average time it takes to respond from start to finish, including uploading a file, processing it, and showing the results, is still less than two seconds for most signature photos. Session isolation makes ensuring that requests from multiple users are handled safely. During testing of model unavailability, the fallback strategy worked as it should, keeping the system available.

V. Conclusion and Future Work

This paper presented a practical signature verification system combining Histogram of Oriented Gradients feature extraction with Support Vector Machine classification. The method gets more than 92% of the answers right and keeps processing time under one second, making it good for real-time authentication apps. The modular architecture with Flask web deployment makes it easy to use, safe, and scalable for checking digital signatures in banking, legal, and administrative settings.

Some of the main accomplishments include the successful combination of interpretable HOG features with strong SVM classification, a complete system design with backup systems that make sure it works, and a practical web-based deployment that makes it easy for a lot of people to use. The system shows that it is better than traditional approaches and is still efficient compared to deep learning methods.

Future work will look into a number of ways to improve things. Combining SVM with random forests or gradient boosting in ensemble methods may make the results more accurate. Adding online dynamic elements like pen pressure and stroke velocity would make it possible to verify both online and offline. Adding more training datasets with different signature styles and using systematic adversarial training against competent forgeries can help the model generalize better. Creating explainable AI visualizations that highlight which parts of a signature affected classification results would make users more likely to trust the system and help with forensic analysis.

Key contributions include effective integration of interpretable HOG features with robust SVM classification, comprehensive system architecture with fallback mechanisms ensuring reliability, and practical web-based deployment enabling broad accessibility.

Additional directions include implementation of continuous authentication through signature sequence analysis, integration with blockchain for immutable audit trails, and extension to mobile platforms enabling field deployment. These enhancements will advance the system toward comprehensive, adaptive signature verification suitable for evolving security requirements in digital document ecosystems.

References

[1] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2005, pp. 886-893.

[2] R. Plamondon and G. Lorette, "Automatic signature verification and writer identification—the state of the art," *Pattern Recognition*, vol. 22, no. 2, pp. 107-131, 1989.

[3] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Offline handwritten signature verification—literature review," in *Proc. International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2017, pp. 1-8.

[4] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The state of the art in online handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, pp. 787-808, Aug. 1990.

[5] Y. Serdouk, H. Nemmour, and Y. Chibani, "New off-line handwritten signature verification method based on artificial immune recognition system," *Expert Systems with Applications*, vol. 51, pp. 186-194, 2016.

[6] D. Impedovo and G. Pirlo, "Automatic signature verification: The state of the art," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 5, pp. 609-635, Sep. 2008.

[7] M. A. Ferrer, J. B. Alonso, and C. M. Travieso, "Offline geometric parameters for automatic signature verification using fixed-point arithmetic," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 993-997, Jun. 2005.

[8] J. Coetzer, B. M. Herbst, and J. A. du Preez, "Offline signature verification using the discrete radon transform and a hidden Markov model," *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 559-571, 2004.

[9] V. L. Blankers, C. E. van den Heuvel, K. Y. Franke, and L. G. Vuurpijl, "ICDAR 2009 signature verification competition," in *Proc. 10th International Conference on Document Analysis and Recognition*, 2009, pp. 1403-1407.

[10] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273-297, Sep. 1995.

[11] B. Zhang, S. N. Srihari, and C. Huang, "Word image retrieval using binary features," in *Proc. Document Recognition and Retrieval XI, SPIE*, vol. 5296, 2004, pp. 45-53.

[12] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Learning features for offline

handwritten signature verification using deep convolutional neural networks," *Pattern Recognition*, vol. 70, pp. 163-176, Oct. 2017.

[13] M. Kalera, S. Srihari, and A. Xu, "Offline

signature verification and identification using distance statistics," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 7, pp. 1339-1360, 2004.

[14] Anagnostopoulos, G. C. (2009). SVM-based target recognition from synthetic aperture radar images using target region outline descriptors. Nonlinear Analysis. https://doi.org/10.1016/j.na.2009.07.030