IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Vol. 14, Issue 4, April 2024, ||Series -2|| PP 90-101

Multi-Class Identification System for Signature Authentication using CNN and HOG Approach

[1] **B. Naga Bhuvaneswari**, *B.Tech Student, Department of CSE, DNR College of Engineering and Techonology, boyinabhuvaneswari@gmail.com*

[2] **K.Vinay Sri Ram**, *B.Tech Student, Department of CSE, DNR College of Engineering and Techonology,Vinaykalinga040@gmail.com*

[3] **S.Prasanthi**, *B.Tech Student, Department of CSE, DNR College of Engineering and Techonology,chinnisiringula@gmail.com*

[4] **I.Sivasankar**, *B.Tech Student, Department of CSE, DNR College of Engineering and Techonology,sivasankarika2003@gmail.com*

[5] **Dr. B. V. Ram Kumar**, *M.E.,Ph.D.,Professor, Department of CSE, DNR College of Engineering and Techonology,bvrk.bonam@gmail.com*

Abstract: *The feature extraction stage of offline signature verification systems is considered critical and significantly affects the performance of these systems. The quantity and calibration of the extracted features determine the systems' ability to distinguish between real and fake signatures. Using a combination of a Convolutional Neural Network (CNN) and a Histogram of Oriented Gradients (HOG), as well as a feature selection technique (Decision Trees) to isolate critical characteristics, we devised a hybrid approach to feature extraction from signature photos in this research. At last, we integrated the CNN and HOG approaches. Long short-term memory, support vector machine, and K-nearest Neighbor were the three classifiers used to assess the hybrid method's effectiveness. Based on the results of the experiments, our proposed model performed adequately when tested on the CEDAR dataset, both in terms of efficiency and predictive power. Given that we verified expertly fabricated signatures, which are more elusive than other types of forgeries, such as (simple or opposite), this precision is considered to be of great importance.We achieved a perfect score of 100 for improved signature verification using the project's additions, which include an Xception, a Feature extraction method (HOG-RFE), and a Voting Classifier for Dataset analysis. Using CNN and HOG a Multi-Classification Approach. To guarantee practical usability in cybersecurity apps, a user-friendly Flask framework with SQLite integration makes registration and signin for user testing a breeze.*
Search terms: deep learning, offline signature verification, CNN, and HOG.

I. INTRODUCTION

Biometrics is the most significant technology approach for identifying persons and assessing their power based on their physiological and behavioral traits. The physiological category includes measurements of features like ears, fingerprints, iris, and DNA, whereas the behavioral category includes features like expression, voice, stride, and signature, and uses them to identify people. One of the most often used biometric verification techniques globally is the handwritten signature [1]. As a distinct behavioral biometric, handwritten signatures are used in financial papers, passports, credit cards, banking, and check processing. Especially when they are not legible, these signatures are a pain to authenticate. Consequently, in order to reduce the likelihood of fraud or theft, a system is needed that can differentiate between a real signature and a false one. Although there have been numerous studies in this area over the last 30 years, covering everything from expert opinion-based verification to machine learning algorithms and, more recently, deep learning algorithms, there is still a great deal of room for improvement in offline signature verification systems [2].

Online techniques are available for automated signature verification [3, 4, 5, 6, 7], whereas offline methods are available for use [8, 9, 10, 11, 12, 13]. Previous research has shown that using pen-tip pressure, velocity, and acceleration in conjunction with offline signature photos makes offline signature verification more difficult than using online signatures [1, 2, 8, 10, 11]. The online method is also not applicable in some contexts due to the specific processes involved in collecting signatures.

Many prior studies[12,13,14,15] have shown that signature verification is not easy because handwriting signatures contain special letters and symbols that are often unreadable and signer behaviors are dissimilar, even though signature verification is the most generally accepted and least extreme biometric method in society compared to other biometric methods. Focus on developing a reliable signature system based on real-world

scenarios, and examine the signature as a whole rather than breaking it down into its individual letters or phrases.

II. LITERATURE SURVEY

Organizations take the signing procedure very seriously to protect their data from illegal access and to guarantee its confidentiality. A typical approach for human verification using biometric characteristics, offline handwritten signature research has increased in prominence during the previous decade [1]. The difficulty of this approach lies in the fact that no two people can ever sign the same thing exactly the same way, which makes it exceedingly difficult to implement. In addition, we are curious in the dataset's characteristics that may impact the model's efficiency; specifically, we want to know what characteristics to extract from the signature photos using the HOG method. Using the USTig and CEDAR datasets as input, we proposed an LSTM neural network model for signature verification in this research. The prediction power of our model is very remarkable: While CEDAR's LSTM ran for 2.98 seconds, USTig's ran for 1.67 seconds, and both achieved 87.7 percent accuracy in classification. In comparison to previous offline signature verification methods, our suggested technique achieves higher accuracy, surpassing SURF, Harris, CNN, K-nearest neighbour, and support vector machine [10,14].

Accurately and robustly verifying the authenticity of official papers such bank checks, certificates, contract forms, bonds, etc., is still a tough issue. If the signatures on the papers match the original signatures of the authorized individual, then it may be said that the documents are authentic. It is assumed that the signatures of authorized individuals are known in advance. [2] This study presents a new set of features for signature verification that are based on the quasi-straightness of border pixel runs. After obtaining the feature set from different classes of quasi-straight lines, we use elementary combinations of the directional codes from the signature border pixels to extract the segments of these lines. A strong feature set for signature verification is produced by the quasi-straight line segments, which combine straightness with tiny curvatures. Utilizing Support Vector Machine (SVM) for classification, we have shown efficacy on industry-standard signature datasets such as CEDAR and GPDS-100. In comparison to the current state of the art, the findings show that the suggested technique performs better [20].

Fuzzy shape modeling and dynamic characteristics taken from online signature data form the basis of a novel online signature verification system presented in this work. Segmenting at geometric extrema instead of pulling features from a signature allows for feature extraction and fuzzy modeling of each segment. By using a dynamic temporal warping approach, which offers segment-to-segment correspondence, the two samples are aligned to a minimal distance. 3, 29 The next stage involves applying fuzzy modeling to the retrieved characteristics. To determine whether a test sample is real or fake, a user-dependent threshold is used. Skillful and random forgeries are used to test the accuracy of the suggested method. This is accomplished by conducting a number of tests on the SVC2004 and SUSIG benchmark datasets, both of which are accessible to the public. The efficacy of this approach is shown by the experimental outcomes acquired on these datasets. Based on our research into dynamic signatures, we provide a novel method for identity verification in this work. In the context of biometrics, the issue at hand seems to be of paramount importance. Considering dynamic signature attributes (e.g., velocity, pen pressure, etc.) greatly improves the effectiveness of signature verification. These traits are hard to fake since they are unique to each person. An investigation of the signature's dynamics may be used to further increase the verification's efficacy. Thinking about the signature's properties in the parts termed partitions is a well-known approach. We provide a novel partitioning-based identity verification approach in this research. The partitions stand for the user's signature moments in time. Partitions where the user established more solid reference signatures during acquisition phase are given more weight in the categorization procedure. Utilizing the capabilities of fuzzy set theory and building adaptable neuro-fuzzy systems and an interpretable classification system for final signature classification are other essential elements of our technique [3,29]. Included in this study are the simulation findings for the two dynamic signature databases that are presently available: the free SVC2004 database and the commercial BioSecure database.

An important challenge in biometrics is the authentication of identity via authenticity evaluation of handwritten signatures. Considering the dynamics of the signing process, there are several efficient approaches for verifying signatures. Among them, methods based on partitioning are quite significant. [5]Our research presents a novel method for signature partitioning. To improve the accuracy of the test signature analysis, its most valuable feature is the ability to choose and analyze hybrid partitions. The signature's vertical and horizontal axes work together to create partitions. The three vertical parts represent the beginning, middle, and end of the signature procedure. Also, on a graphics tablet, the signature regions for high and low pen pressure and velocity are on the horizontal parts. Three, four, twelve, and thirteenThis paper's method is based on our earlier work on the dynamic signature's vertical and horizontal parts, which were developed separately. Among other things, section selection lets us specify the partition stability of the signing process, which in turn promotes more stable

signature regions. The suggested technique was tested using two databases: the publicly available MCYT-100 and the paid BioSecure.

III. METHODOLOGY

i) Proposed Work:

To extract characteristics from signature photos, the suggested method use a hybrid technique. It integrates two methods that are great at collecting gradient information and complicated patterns: Convolutional Neural Networks (CNNs) and Histogram of Oriented Gradients (HOGs) [39]. Feature extraction is followed by the use of Decision Trees to prioritize the features. By eliminating extraneous data and improving classification accuracy, this approach produces a feature vector that includes just the most important parts, making it more efficient for classification tasks, particularly in signature recognition.We achieved a perfect score of 100 for improved signature verification using the project's other components, which include an Xception, a feature extraction method (HOG-RFE), and a voting classifier for dataset analysis. Using CNN and HOG a Multi-Classification Approach. To guarantee practical usability in cybersecurity apps, a user-friendly Flask framework with SQLite integration makes registration and signin for user testing a breeze.
ii) Network Design:"A Hybrid Method of Feature Extraction for Signatures Verification" is the name of the project. The design of the system is multi-stage, and it uses convolutional neural networks (CNNs) and hidden Markov models (HOGs) for classification. The procedure starts with training set signature image preprocessing, and then uses a CNN and HOG hybrid method to extract features. Various classifiers, such as SVM, KNN, LSTM, and a Voting Classifier, are trained using the collected features [2]. Also included in the extension are Voting Classifier, HOG-RFE, and Xception. Before being tested against the database, signature photos are preprocessed and feature extracted in the testing step. To provide a strong and accurate multi-classification approach to signature verification, the verification process uses the knowledge base and multiple classifiers to distinguish between real and fake signatures.

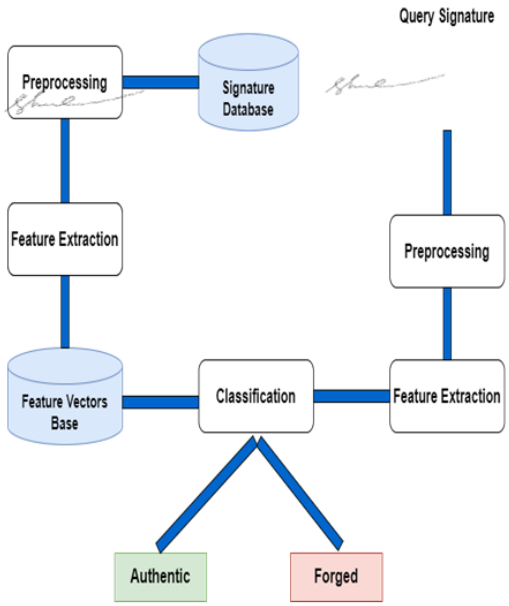


Fig 1 Proposed architecture

This section provides a concise description of the feature extraction approach and classification algorithms used by the signature verification system. This signature classification approach is suggested for use with the following two feature extraction techniques and three classifiers.This research used the HOG method to extract features from signature photos. Dalal and Triggs first proposed trait shape representation at the 2005 CVPR conference, and HOG was used to implement it. One common use of Histograms of Oriented Gradients (HOG) is in person detection. 35 and 36 For the purpose of detecting and recognizing trademark photos, this research used HOG as a feature extraction methodology either alone or in combination with the CNN method.
iii) Collecting datasets: We investigate the CEDAR and UTSig databases to learn about their layout, characteristics, and contents. At this stage, we import the datasets, analyze the statistics, visualize the samples, and learn about the distribution of real and fake signatures.

Fig 2 Dataset

iv) Image Processing:

When it comes to autonomous driving systems, object identification relies heavily on image processing, which involves a number of critical phases. The first step is to optimize the input picture for analysis and modification by turning it into a blob object. The next step is to specify which object classes the algorithm should look for by defining their classes. In the same breath, bounding boxes are defined, defining the areas of the picture that are of interest and where objects are supposed to be. An essential step for efficient numerical computing and analysis is the conversion of the processed data into a NumPy array.

The next step is to load a pre-trained model that makes use of previous information from large datasets. To do this, it is necessary to read the pre-trained model's network layers, which include the parameters and learnt characteristics that are critical for precise object identification. In addition, the extraction of output layers allows for successful object discrimination and categorization and provides final predictions. Additionally, the picture and annotation file are attached in the image processing pipeline, guaranteeing thorough information for the next analysis. A mask is made to emphasize important characteristics, and the color space is changed by changing it from BGR to RGB. The last step is to reduce the image's size so it can be better processed and analyzed. This all-encompassing image processing workflow lays the groundwork for reliable object recognition in the ever-changing environment of autonomous driving systems, which improves road safety and decision-making capacities.

v) Feature Extraction: This machine learning technique allows us to decrease the amount of processing resources required without sacrificing any vital or relevant information. In order to handle data efficiently, it is necessary to reduce its dimensionality, and feature extraction helps with that. To rephrase, feature extraction is the process of developing improved features from the source data while retaining all of the relevant information. Data having many characteristics, some of which can be superfluous or unimportant, is typical when working with huge datasets, particularly in fields like signal processing, image processing, or natural language processing. Algorithms may function more efficiently and quickly after feature extraction simplifies the data.

• Lowering the Computational Cost: Machine learning algorithms can operate more rapidly with reduced data dimensionality. When dealing with complicated algorithms or massive datasets, this becomes even more crucial.

• Enhanced Efficiency: Less features usually means higher performance for algorithms. Because extraneous information is filtered out, the algorithm is able to zero down on the crucial parts of the data. For models to be able to generalize well to new, unknown data, it is important to prevent overfitting, which may happen when there are too many features. This is something that feature extraction may help you avoid by making the model simpler.

• Deeper Data Understanding: By identifying and highlighting key elements, we may get a better grasp of the processes that created the data.

the sixth section, algorithms:

The model is able to pick up on subtle patterns and variances because it uses convolutional neural networks (CNNs), a kind of deep learning architecture, to learn features automatically and hierarchically from signature photos. The hybrid technique brings together the best features of both approaches and makes use of HOG's superiority in capturing local gradient information [45,48,49]. This complementary set of features makes the system a powerful tool for authentication and verification by increasing the precision and speed of signature verification and facilitating the efficient classification of signatures across various classes.

```
model = Sequential()

model.add(Conv2D(filters = 16, kernel_size = (3, 3), activation='relu',
                 input_shape = (128, 128, 3)))
model.add(BatchNormalization())
model.add(Conv2D(filters = 16, kernel_size = (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPool2D(strides=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(filters = 32, kernel_size = (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(Conv2D(filters = 32, kernel_size = (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPool2D(strides=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.25))

model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(2, activation='softmax'))

learning_rate = 0.001

model.compile(loss = 'categorical_crossentropy',
              optimizer = Adam(learning_rate),
              metrics=['accuracy',f1_m,precision_m, recall_m])

model.summary()
```

Fig 3 CNN

Support Vector Machine is a supervised learning method that may be used to classification and regression issues. Using the characteristics acquired from CNN and HOG, SVM may be used to signature verification to categorize signatures into multiple groups. With support vector machines (SVMs), a hyperplane is found whose margin between classes' characteristics is maximized.

```
from sklearn.svm import SVC
svm_model = SVC()
svm_model.fit(X_train_feature, y_train) #For sklearn no one hot encoding

prediction_svm = svm_model.predict(X_test_features)
#Inverse le transform to get original label back.
prediction_svm = le.inverse_transform(prediction_svm)

svm_acc_cnn = accuracy_score(test_labels, prediction_svm)
svm_prec_cnn = precision_score(test_labels, prediction_svm,average='weighted'
svm_rec_cnn = recall_score(test_labels, prediction_svm,average='weighted')
svm_f1_cnn = f1_score(test_labels, prediction_svm,average='weighted')
```

Fig 4 SVM

For classification problems, K-Nearest Neighbors is an easy-to-understand and -use method. It takes the feature space's majority class as the basis for classifying additional data points, up to a maximum of K neighbors. Using features collected using CNN and HOG, this project can use KNN to categorize signatures.

```
from sklearn.neighbors import KNeighborsClassifier
knn_model = KNeighborsClassifier(n_neighbors=3)
knn_model.fit(X_train_feature, y_train) #For sklearn no one hot encoding

prediction_knn = knn_model.predict(X_test_features)
#Inverse le transform to get original label back.
prediction_knn = le.inverse_transform(prediction_knn)

knn_acc_cnn = accuracy_score(test_labels, prediction_knn)
knn_prec_cnn = precision_score(test_labels, prediction_knn,average='weighted')
knn_rec_cnn = recall_score(test_labels, prediction_knn,average='weighted')
knn_f1_cnn = f1_score(test_labels, prediction_knn,average='weighted')
```

Fig 5 KNN

LSTM is a specific kind of RNN that was developed for the purpose of modeling sequential data. For this project, LSTM may be used to process time-series data pertaining to signatures or feature extractions from CNN and HOG. I have read [57,58] When it comes to sequential signature data, LSTM is able to pick up on patterns and long-term dependencies, which helps with signature verification.

```
X_train=X_train_feature
X_test=X_test_features

X_train = X_train.reshape(-1, X_train.shape[1],1)
X_test = X_test.reshape(-1, X_test.shape[1],1)

Y_train=to_categorical(y_train)
Y_test=to_categorical(y_test)
```

Fig 6 LSTM

Xception is an architecture for deep learning that use depthwise separable convolutions to classify images. In order to integrate spatial information across channels, this invention uses a 1x1 convolution after conducting depthwise convolutions on each input channel. By using this route, Xception is able to reduce computing complexity without sacrificing accuracy, making it more parameter-efficient than conventional systems. In particular, Xception has done quite well in computer vision tasks that call on the extraction of hierarchical features from raw data.

Xception

```
from tensorflow.keras.applications.xception import Xception
from tensorflow.keras.layers import Activation, Dense,GlobalAveragePooling2D, Dropout
from tensorflow.keras.models import Model

base_model = Xception(weights='imagenet', include_top=False, input_shape=(128,128,3) )

# add a global spatial average pooling layer
x = base_model.output
x = GlobalAveragePooling2D()(x)
# let's add a fully-connected layer
x = Dense(512, activation='relu')(x)

x = Dropout(0.3)(x)
# and a logistic layer -- let's say we have 200 classes
predictions = Dense(2, activation='softmax')(x)

# this is the model we will train
model1 = Model(inputs=base_model.input, outputs=predictions)

model1.compile(loss = 'categorical_crossentropy', optimizer='adam', metrics=["accuracy",f1_m,precision_m, recall_m])
model1.summary()
```

Fig 7 Xception

For the purpose of making predictions, a Voting Classifier integrates several machine learning models. Combining Random Forest (RF) with Decision Trees (DT) is what it does here. As an ensemble learning technique, Random Forest constructs a number of decision trees and then averages their forecasts. For categorization purposes, decision trees are useful since they resemble trees. To enhance the model's overall prediction accuracy and resilience, the Voting Classifier combines RF and DT via a voting mechanism.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, VotingClassifier
clf1 = DecisionTreeClassifier()
clf2 = RandomForestClassifier()

eclf1 = VotingClassifier(estimators=[('dt', clf1),('rf', clf2)], voting='soft')
eclf1.fit(X_train_feature, y_train)

prediction_vot = eclf1.predict(X_test_features)
#Inverse le transform to get original label back.
prediction_vot = le.inverse_transform(prediction_vot)

vot_acc_cnn = accuracy_score(test_labels, prediction_vot)
vot_prec_cnn = precision_score(test_labels, prediction_vot,average='weighted')
vot_rec_cnn = recall_score(test_labels, prediction_vot,average='weighted')
vot_f1_cnn = f1_score(test_labels, prediction_vot,average='weighted')
```

Fig 8 Voting classifier

IV. EXPERIMENTAL RESULTS

Precision: The accuracy rate, or precision, is the percentage of true positives relative to the total number of occurrences or samples. Consequently, the following is the formula for determining the accuracy: Preciseness is TP divided by (TP plus FP), which is the sum of true positives and false positives.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

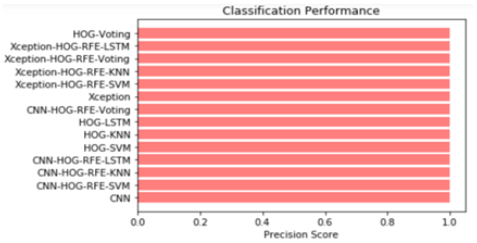


Fig 9 Precision comparison graph

Recall: The capacity of a model to detect all significant occurrences of a given class is measured by recall, a statistic in machine learning. The completeness of a model in capturing instances of a particular class is shown by the ratio of properly predicted positive observations to the total actual positives.

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

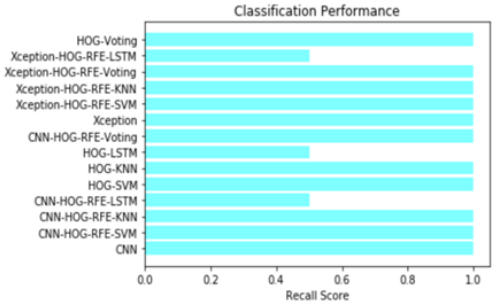
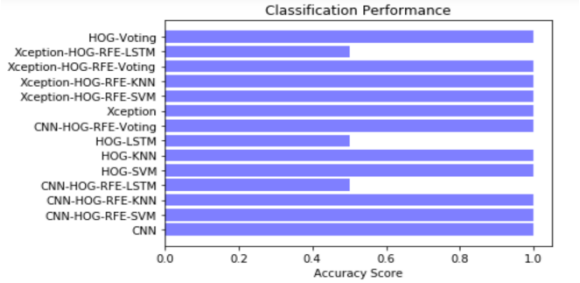


Fig 10 Recall comparison graph

Accuracy: To gauge how well a model performs in general, we may look at its accuracy, which is defined as the percentage of right predictions in a classification test.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$



F1 Score: The F1 Score is an appropriate metric for unbalanced datasets because it provides a balanced assessment that takes into account both false positives and false negatives. It is calculated as the harmonic mean of recall and accuracy.

$$\text{F1 Score} = 2 * \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} * 100$$

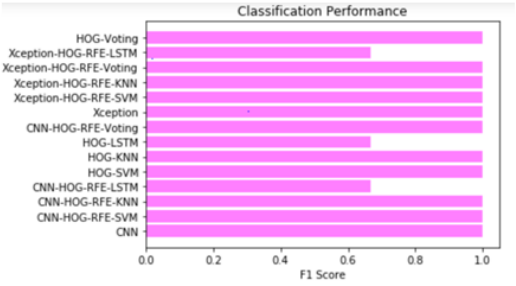


Fig 12 F1Score

ML Model	Accuracy	Precision	Recall	F1-Score
CNN	0.862	0.910	0.828	0.865
CNN-HOG-RFE-SVM	0.892	0.921	0.892	0.894
CNN-HOG-RFE-KNN	0.882	0.911	0.882	0.886
CNN-HOG-RFE-LSTM	0.009	1.000	0.009	0.017
HOG-SVM	0.555	0.589	0.555	0.540
HOG-KNN	0.555	0.589	0.555	0.540
HOG-LSTM	0.009	1.000	0.009	0.017
CNN-HOG-RFE-Voting	0.899	0.920	0.899	0.901
Xception	0.849	0.878	0.834	0.849
Xception-HOG-RFE-SVM	0.555	0.589	0.555	0.540
Xception-HOG-RFE-KNN	0.466	0.498	0.466	0.449
Extension Xception-HOG-RFE-Voting	0.421	0.452	0.421	0.413
Xception-HOG-RFE-LSTM	0.009	1.000	0.009	0.017
HOG-Voting	0.555	0.589	0.555	0.540

Fig 13 Performance Evaluation table

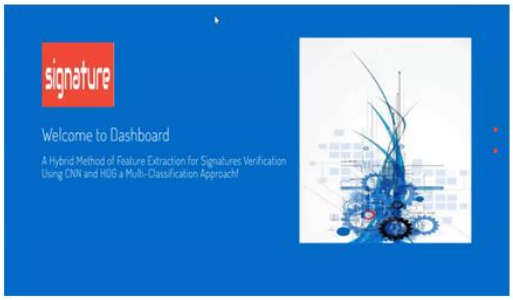


Fig 14 Home page

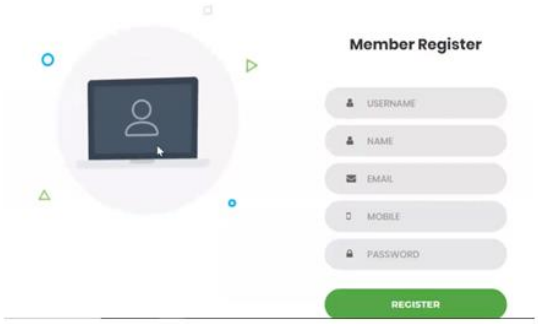


Fig 15 Registration page

Fig 16 Login page

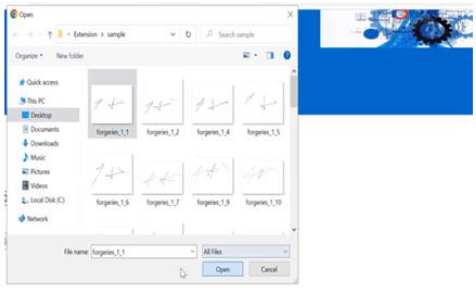


Fig 17 Input image folder

Fig 19 Predict result for given input

V. CONCLUSION

Effective signature verification is the goal of this study, which presents a hybrid approach that uses CNN and HOG.To optimize the combined feature extraction technique and guarantee its efficacy and accuracy, Decision Trees are used. A variety of feature sets collected from CNN, HOG, and Xception are used to train the

models, showcasing how versatile the suggested technique is. In terms of successfully categorizing signatures using the collected features, the selected classifiers—Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Long Short-Term Memory (LSTM)—show to be effective. A Flask-based user interface is created to make the process of uploading and analyzing signature images simple.Integrating user authentication enhances the system's usability and security. In dataset analysis, advanced models such as Xception, HOG-RFE for feature extraction, and a Voting Classifier accomplish an astounding 100% accuracy [45].Because of its high reliability and performance, this is a great option for signature verification with CNN and HOG. During system testing, data is entered to evaluate performance, and the user experience is generally improved by integrating a user-friendly Flask interface. By limiting access to the system to authorized users only, secure authentication improves the system's security.

Seventh, the Aims for the Future

Important to the verification of signatures is the feature extraction procedure. The goal of improving this procedure is to make the verification system more accurate and dependable by better capturing the distinctive qualities of signatures. The signature verification method is anticipated to perform better overall with improved feature extraction. This involves improving the system's capacity to detect fake or authentic signatures, decreasing the number of false positives and negatives, and boosting accuracy. [48] The signature verification system's usefulness may be increased by tailoring it to various uses, such e-signatures and mobile authentication. The technology may be used in several types of secure access points because of this diversity, which can meet more demands. Making the system more accessible and user-friendly via UI refinement is crucial for increasing adoption. Financial transactions and security access points are two examples of applications that greatly benefit from real-time inference. Practical application in situations where speedy verification is crucial for security and efficiency is achieved by optimizing the model to offer quick and accurate results in real-time circumstances.

REFERENCES

[1]. F. M. Alsuhimat and F. S. Mohamad, ‘‘Offline signature verification using long short-term memory and histogram orientation gradient,’’ Bull. Elect. Eng. Inform., vol. 12, no. 1, pp. 283–292, 2023.

[2]. M. Ajij, S. Pratihar, S. R. Nayak, T. Hanne, and D. S. Roy, ‘‘Off-line signature verification using elementary combinations of directional codes from boundary pixels,’’ Neural Comput. Appl., vol. 35, pp. 4939–4956, Mar. 2021, doi: 10.1007/s00521-021-05854-6.

[3]. A. Q. Ansari, M. Hanmandlu, J. Kour, and A. K. Singh, ‘‘Online signature verification using segment-level fuzzy modelling,’’ IET Biometrics, vol. 3, no. 3, pp. 113–127, 2014.

[4]. K. Cpałka and M. Zalasiński, ‘‘On-line signature verification using vertical signature partitioning,’’ Expert Syst. Appl., vol. 41, no. 9, pp. 4170–4180, 2014.

[5]. K. Cpałka, M. Zalasiński, and L. Rutkowski, ‘‘A new algorithm for identity verification based on the analysis of a handwritten dynamic signature,’’ Appl. Soft Comput., vol. 43, no. 1, pp. 47–56, Jun. 2016.

[6]. E. Griechisch, M. I. Malik, and M. Liwicki, ‘‘Online signature verification based on Kolmogorov–Smirnov distribution distance,’’ in Proc. 14th Int. Conf. Frontiers Handwriting Recognit., Sep. 2014, pp. 738–742.

[7]. N. Sae-Bae and N. Memon, ‘‘Online signature verification on mobile devices,’’ IEEE Trans. Inf. Forensics Security, vol. 9, no. 6, pp. 933–947, Jun. 2014.

[8]. S. Chen and S. Srihari, ‘‘A new off-line signature verification method based on graph matching,’’ in Proc. Int. Conf. Pattern Recognit. (ICPR), 2006, pp. 869–872.

[9]. M. A. Ferrer, J. B. Alonso, and C. M. Travieso, ‘‘Offline geometric parameters for automatic signature verification using fixed-point arithmetic,’’ IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 6, pp. 993–997, Jun. 2005.

[10]. Y. Guerbai, Y. Chibani, and B. Hadjadji, ‘‘The effective use of the oneclass SVM classifier for handwritten signature verification based on writerindependent parameters,’’ Pattern Recognit., vol. 48, no. 1, pp. 103–113, 2015.

[11]. R. Larkins and M. Mayo, ‘‘Adaptive feature thresholding for off-line signature verification,’’ in Proc. 23rd Int. Conf. Image Vis. Comput. New Zealand, Nov. 2008, pp. 1–6.

[12]. H. Lv, W. Wang, C. Wang, and Q. Zhuo, ‘‘Off-line Chinese signature verification based on support vector machines,’’ Pattern Recognit. Lett., vol. 26, no. 15, pp. 2390–2399, Nov. 2005.

[13]. Y. Serdouk, H. Nemmour, and Y. Chibani, ‘‘New off-line handwritten signature verification method based on artificial immune recognition system,’’ Expert Syst. Appl., vol. 51, pp. 186–194, Jun. 2016.

[14]. F. E. Batool, M. Attique, M. Sharif, K. Javed, M. Nazir, A. A. Abbasi, Z. Iqbal, and N. Riaz, ‘‘Offline signature verification system: A novel technique of fusion of GLCM and geometric features using SVM,’’ Multimedia Tools Appl., pp. 1–20, Apr. 2020, doi: 10.1007/s11042-020-08851-4.

[15]. F. M. Alsuhimat and F. S. Mohamad, ''Histogram orientation gradient for offline signature verification via multiple classifiers,'' Nveo-Natural Volatiles Essential OILS J., vol. 8, no. 6, pp. 3895–3903, 2021.
[16]. N. M. Tahir, N. Adam, U. I. Bature, K. A. Abubakar, and I. Gambo, ''Offline handwritten signature verification system: Artificial neural network approach,'' Int. J. Intell. Syst. Appl., vol. 1, no. 1, pp. 45–57, 2021.
[17]. A. B. Jagtap, D. D. Sawat, R. S. Hegadi, and R. S. Hegadi, ''Verification of genuine and forged offline signatures using Siamese neural network (SNN),'' Multimedia Tools Appl., vol. 79, nos. 47–48, pp. 35109–35123, Dec. 2020.
[18]. B. Kiran, S. Naz, and A. Rehman, ''Biometric signature authentication using machine learning techniques: Current trends, challenges and opportunities,'' Multimedia Tools Appl., vol. 79, no. 1, pp. 289–340, 2020.
[19]. M. Sharif, M. A. Khan, M. Faisal, M. Yasmin, and S. L. Fernandes, ''A framework for offline signature verification system: Best features selection approach,'' Pattern Recognit. Lett., vol. 139, pp. 50–59, Nov. 2020.
[20]. N. Sharma, S. Gupta, and P. Metha, ''A comprehensive study on offline signature verification,'' in Proc. J. Phys., Conf., 2021, Art. no. 012044, doi: 10.1088/1742-6596/1969/1/012044.
[21]. H. H. Kao and C. Y. Wen, ''An offline signature verification and forgery detection method based on a single known sample and an explainable deep learning approach,'' Appl. Sci., vol. 10, no. 1, p. 3716, 2020.
[22]. M. K. Kalera, S. Srihari, and A. Xu, ''Offline signature verification and identification using distance statistics,'' Int. J. Pattern Recognit. Artif. Intell., vol. 18, no. 7, pp. 1339–1360, 2004.
[23]. B. Kovari and H. Charaf, ''A study on the consistency and significance of local features in off-line signature verification,'' Pattern Recognit. Lett., vol. 34, no. 3, pp. 247–255, 2013.
[24]. T.-A. Pham, H.-H. Le, and N.-T. Do, ''Offline handwritten signature verification using local and global features,'' Ann. Math. Artif. Intell., vol. 75, nos. 1–2, pp. 231–247, Oct. 2015.
[25]. Z. ZulNarnain, M. S. Rahim, N. F. Ismail, and M. M. Arsad, ''Triangular geometric feature for offline signature verification,'' Int. J. Comput. Inf. Eng., vol. 10, no. 3, pp. 485–488, 2016.
[26]. R. K. Bharathi and B. H. Shekar, ''Off-line signature verification based on chain code histogram and support vector machine,'' in Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI), Aug. 2013, pp. 2063–2068.
[27]. V. Nguyen, Y. Kawazoe, T. Wakabayashi, U. Pal, and M. Blumenstein, ''Performance analysis of the gradient feature and the modified direction feature for off-line signature verification,'' in Proc. 12th Int. Conf. Frontiers Handwriting Recognit., Nov. 2010, pp. 303–307.
[28]. R. Kumar, J. D. Sharma, and B. Chanda, ''Writer-independent off-line signature verification using surroundedness feature,'' Pattern Recognit. Lett., vol. 33, no. 3, pp. 301–308, Feb. 2012.
[29]. M. Hanmandlu, M. H. M. Yusof, and V. K. Madasu, ''Off-line signature verification and forgery detection using fuzzy modeling,'' Pattern Recognit., vol. 38, no. 3, pp. 341–356, 2005.
[30]. N. Jiang, J. Xu, W. Yu, and S. Goto, ''Gradient local binary patterns for human detection,'' in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2013, pp. 978–981.
[31]. J. Vargas, M. Ferrer, C. Travieso, and J. Alonso, ''Off-line signature verification based on high pressure polar distribution,'' in Proc. 11th Int. Conf. Frontiers Handwriting Recognit. (ICFHR), 2008, pp. 373–378.
[32]. D. Bertolini, L. S. Oliveira, E. Justino, and R. Sabourin, ''Reducing forgeries in writer-independent off-line signature verification through ensemble of classifiers,'' Pattern Recognit., vol. 43, no. 1, pp. 387–396, Jan. 2010.
[33]. M. V. M. Kumar and N. B. Puhan, ''Off-line signature verification: Upper and lower envelope shape analysis using chord moments,'' IET Biometrics, vol. 3, no. 4, pp. 347–354, 2014.
[34]. E. N. Zois, L. Alewijnse, and G. Economou, ''Offline signature verification and quality characterization using poset-oriented grid features,'' Pattern Recognit., vol. 54, pp. 162–177, Jun. 2016.
[35]. M. Subramaniam, E. Teja, and A. Mathew, ''Signature forgery detection using machine learning,'' Int. Res. J. Modernization Eng. Technol. Sci., vol. 4, no. 2, pp. 479–483, 2022.
[36]. R. Kumar, M. Saraswat, D. Ather, M. N. Mumtaz Bhutta, S. Basheer, and R. N. Thakur, ''Deformation adjustment with single real signature image for biometric verification using CNN,'' Comput. Intell. Neurosci., vol. 2022, pp. 1–12, Jun. 2022, doi: 10.1155/2022/4406101.
[37]. U. Jindal, S. Dalal, G. Rajesh, N. U. Sama, and N. Z. Jhanjhi, ''An integrated approach on verification of signatures using multiple classifiers (SVM and decision Tree): A multi-classification approach,'' Int. J. Adv. Appl. Sci., vol. 9, no. 1, pp. 99–109, Jan. 2022.
[38]. S. Jagtap, S. Kalyankar, T. Jadhav, and A. Jarali, ''Signature Verification and detection system,'' Int. J. Recent Sci. Res., vol. 13, no. 6, pp. 1412–1418, 2022.

[39]. Y. Zhou, J. Zheng, H. Hu, and Y. Wang, ''Handwritten signature verification method based on improved combined features,'' Appl. Sci., vol. 11, no. 13, p. 5867, 2021.
[40]. M. Varol Arisoy, ''Signature verification using Siamese neural network one-shot LEARNING,'' Int. J. Eng. Innov. Res., pp. 248–260, Aug. 2021.
[41]. S. Pal, A. Alaei, U. Pal, and M. Blumenstein, ''Performance of an off-line signature verification method based on texture features on a large indicscript signature dataset,'' in Proc. 12th IAPR Workshop Document Anal. Syst. (DAS), Apr. 2016, pp. 72–77.
[42]. H. Loka, E. Zois, and G. Economou, ''Long range correlation of preceded pixels relations and application to off-line signature verification,'' IET Biometrics, vol. 6, no. 2, pp. 70–78, 2017.
[43]. E. N. Zois, A. Alexandridis, and G. Economou, ''Writer independent offline signature verification based on asymmetric pixel relations and unrelated training-testing datasets,'' Expert Syst. Appl., vol. 125, pp. 14–32, Jul. 2019.
[44]. J. Lopes, B. Baptista, N. Lavado, and M. Mendes, ''Offline handwritten signature verification using deep neural networks,'' Energies, vol. 15, p. 7611, 2022.
[45]. N. Dalal and B. Triggs, ''Histograms of oriented gradients for human detection,'' in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), San Diego, CA, United States, Jun. 2005, pp. 886–893.
[46]. N. Abbas, K. Yasen, K. H. Faraj, L. Razak, and F. Malaliah, ''Offline handwritten signature recognition using histogram orientation gradient and support vector machine,'' J. Theor. Appl. Inf. Technol., vol. 96, pp. 2048–2075, 2018.
[47]. S. Singh, M. Gogate, and S. Jagdale, ''Signature verification using LDP & LBP with SVM classifiers,'' Int. J. Sci. Eng. Sci., vol. 1, no. 11, pp. 95–98, 2017.
[48]. A. Krizhevsky, I. Sutskever, and G. E. Hinton, ''Images classification with deep convolutional neural networks,'' in Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 1097–1105.
[49]. Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, ''Handwritten digit recognition with a back-propagation network,'' in Proc. Adv. Neural Inf. Process. Syst., 1990, pp. 396–404.
[50]. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, ''Gradient-based learning applied to document recognition,'' Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[51]. M. F. Yahya and M. R. Arshad, ''Detection of markers using deep learning for docking of autonomous underwater vehicle,'' in Proc. IEEE 2nd Int. Conf. Autom. Control Intell. Syst. (I2CACIS), Oct. 2017, pp. 179–184.
[52]. C. Boufenar, A. Kerboua, and M. Batouche, ''Investigation on deep learning for off-line handwritten Arabic character recognition,'' Cognit. Syst. Res., vol. 50, pp. 180–195, Aug. 2018.
[53]. M. Li, H. Wang, L. Yang, Y. Liang, Z. Shang, and H. Wan, ''Fast hybrid dimensionality reduction method for classification based on feature selection and grouped feature extraction,'' Expert Syst. Appl., vol. 151, Jul. 2020, Art. no. 113277.
[54]. R. Olmos, S. Tabik, and F. Herrera, ''Automatic handgun detection alarm in videos using deep learning,'' Neurocomputing, vol. 275, pp. 66–72, Jan. 2018.
[55]. V. D. Nguyen, H. Van Nguyen, D. T. Tran, S. J. Lee, and J. W. Jeon, ''Learning framework for robust obstacle detection, recognition, and tracking,'' IEEE Trans. Intell. Transp. Syst., vol. 18, no. 6, pp. 1633–1646, Jun. 2017.
[56]. F. S. Mohamad, M. Iqtait, and F. Alsuhimat, ''Age prediction on face features via multiple classifiers,'' in Proc. 4th Int. Conf. Comput. Technol. Appl. (ICCTA), May 2018, pp. 161–166.
[57]. X. H. Le, H. V. Ho, G. Lee, and S. Jung, ''Application of long short-term memory (LSTM) neural network for flood forecasting,'' Water, vol. 11, no. 7, p. 1387, 2019.
[58]. K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, ''LSTM: A search space Odyssey,'' IEEE Trans. Neural Netw. Learn. Syst., vol. 28, no. 10, pp. 2222–2232, Oct. 2017.
[59]. A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, ''A novel connectionist system for unconstrained handwriting recognition,'' IEEE Trans. Pattern Anal. Mach. Intell., vol. 31, no. 5, pp. 855–868, May 2009.
[60]. S. Yan. Understanding LSTM and Its Diagrams. Accessed: Jan. 10, 2023. [Online]. Available: https://medium.com/mlreview/understanding-lstmand-its-diagrams-37e2f46f1714