

Signature Forgery Detection

AI in Forensic Science Project



Team Members 😊



Name	ID
George Hany Milad	21-00724
Bishoy Ezzat Hanna	21-00584

Supervised by: Dr. Yasmine Mahmoud

Assistant Supervisor: Eng. Bassma Bassem

01

Introduction

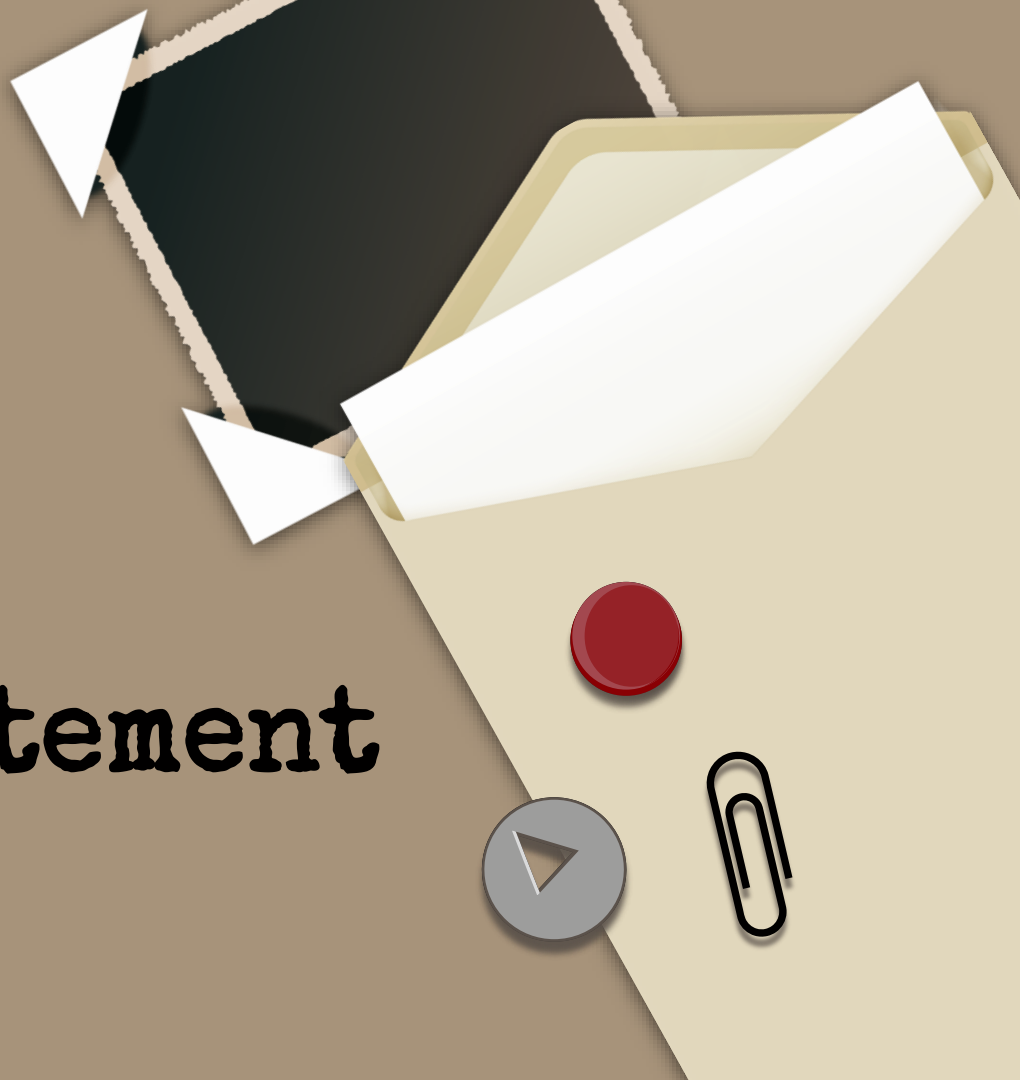


Introduction

- ❖ It is a great pleasure to present our project entitled "Signature Forgery Detection".
- ❖ In today's world, signature forgery represents a major challenge across many sectors, including banking, legal documentation, and identity verification.
- ❖ Our project focuses on developing an automated system that leverages Machine Learning and Image Processing techniques to distinguish between real and forged signatures accurately.
- ❖ Throughout this presentation, we will guide you through the problem definition, our methodology, the techniques we applied, and the results we achieved.
- ❖ Thank you for your attention, and let's begin.

02

Problem Statement



Problem Statement

- ❖ In today's digital world, signature forgery has become a significant threat, leading to financial fraud, identity theft, and legal complications.
- ❖ Importantly, our project is categorized under Forensic Document Examination Domain. It focuses on analyzing images of handwritten signatures, which supports modern forensic analysis efforts.
- ❖ Manual methods of detecting forged signatures are often time-consuming, subjective, and prone to human error.
- ❖ Therefore, our project addresses the strong need for an automated, accurate, and efficient Machine Learning solution capable of reliably detecting and differentiating between real and forged signatures.

03

Project Goal



Project Goal

- ❖ The main objective of this project is to develop a reliable and efficient system for signature forgery detection.
- ❖ Our goal is to build a Machine Learning model that can:
 - Accurately classify signatures as real or forged.
 - Reduce the reliance on manual verification processes.
 - Provide fast and consistent results to support real-world applications such as banking, legal, and security systems.



04

Methodology



Methodology

- ❖ To achieve our objective, we followed a structured approach consisting of the following key steps:

1. Data Collection:

We collected a dataset from [Kaggle](#) containing both real and forged signatures from multiple sources.

2. Preprocessing:

- Converted images to grayscale.
- Resized and applied Gaussian Blur to enhance feature extraction.

3. Feature Extraction:

We used the Histogram of Oriented Gradients (HOG) technique to extract meaningful features from the signature images.

Methodology (Cont.)

4. Dimensionality Reduction:

Applied Principal Component Analysis (PCA) to reduce feature space and improve model performance.

5. Model Building:

Trained a Support Vector Machine (SVM) classifier using the extracted and reduced features.

6. Evaluation:

Evaluated the model's performance using metrics like accuracy, precision, recall, and F1-score.

05

Dataset Description



Dataset Description

- ❖ The dataset used in our project is specifically designed for signature forgery detection tasks. It is organized as follows:

The training dataset contains four subfolders:

- dataset1
- dataset2
- dataset3
- dataset4

Each subfolder is further divided into two categories:

- real: Contains genuine signatures.
- forge: Contains forged (fake) signatures.

Dataset Description (Cont.)

The detailed description of subfolders are:

- dataset1: 60 real signatures, 60 forged signatures.
- dataset2: 60 real signatures, 60 forged signatures.
- dataset3: 150 real signatures, 150 forged signatures.
- dataset4: 90 real signatures, 90 forged signatures.

❖ All images are in PNG format.

❖ The dataset is well-balanced, ensuring that the model can learn equally from both real and forged examples, leading to more reliable and accurate predictions.

Dataset Description (Cont.)

The testing dataset contains two categories or subfolders:

- real: Contains genuine signatures.
- forge: Contains forged (fake) signatures.

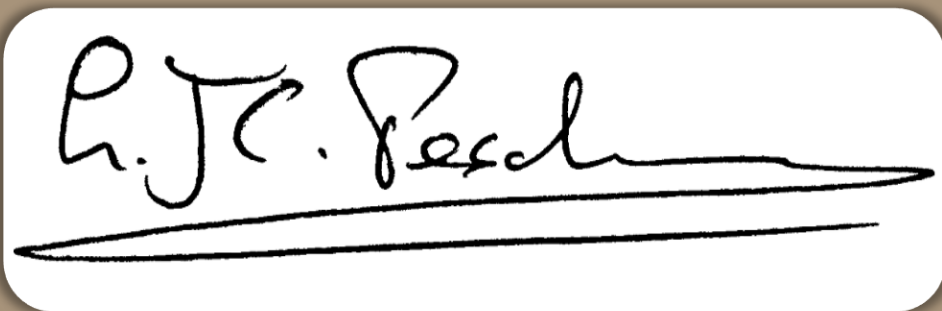
The detailed description of categories or subfolders are:

- real subfolder: **150** images.
- forge subfolder: **150** images.



Dataset Description (Cont.)

Real Signature :

A handwritten signature in black ink on a white background. The signature reads "R. J. C. Besch" in a cursive style. The letters are connected, and there is a long, horizontal, slightly wavy line extending from the end of the word "Besch".

Forge Signature :

A handwritten signature in black ink on a white background, appearing to be a forgery of the real signature. The signature reads "R J C Besch" in a cursive style. The letters are more separated than in the real signature, and the horizontal line at the end is straighter and less wavy.

Dataset Description (Cont.)

Real Signature :

A handwritten signature in black ink on a white background. The signature is cursive and reads "Mayuri". The letters are connected, and there is a long horizontal stroke at the bottom.

Forge Signature :

A handwritten signature in black ink on a white background. The signature is cursive and reads "Mayuri". The letters are connected, and there is a long horizontal stroke at the bottom. The signature appears to be a copy of the real signature.

06

Preprocessing



Preprocessing

❖ Preprocessing steps before extracting features from the image:

1. Convert to Grayscale:

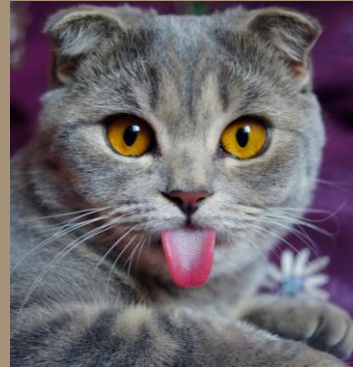
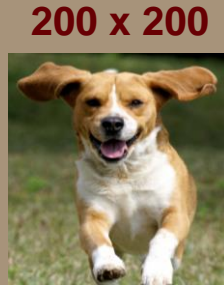
- Since color information is not useful for shape or edge detection, all images were converted to grayscale to reduce data complexity.



Preprocessing (Cont.)

2. Resize Images:

- All images were resized to 200x200 pixels to ensure a uniform input size for feature extraction.



Preprocessing (Cont.)

3. Apply Gaussian Blur:

- Gaussian blur with a 3x3 kernel was applied to reduce noise and smooth the image, helping HOG focus on meaningful edges rather than random small details.



Preprocessing (Cont.)

❖ **Purpose of Preprocessing:**

- These steps ensure that the images are simplified and standardized, allowing HOG to effectively capture geometric features (like edges, curves, and gradients) without interference from irrelevant details like color, scale, or noise.



07

Feature Extraction



Feature Extraction

Feature Extraction using HOG:

- After preprocessing, I extract meaningful features from the signature images using the Histogram of Oriented Gradients (HOG) method.

What is HOG?

- HOG is a feature descriptor that captures the shape and structure of objects by analyzing the distribution of intensity gradients or edge directions in an image.



Feature Extraction (Cont.)

Why HOG for Signature Forgery Detection?

- It focuses on the geometric details of the signature, such as curves, edges, and stroke directions.
- It ignores color and illumination, making it ideal for detecting subtle differences between real and forged signatures.
- It produces a compact and informative feature vector that can be fed into machine learning model.

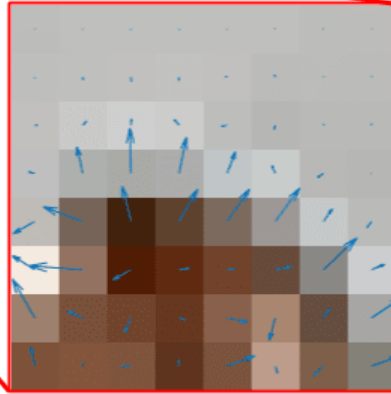


Feature Extraction (Cont.)

How it works in my project ?

1. **Pre-process the Image:** The image is first converted to grayscale and pre-processed (e.g., resized, blurred) to simplify the extraction of features.
2. **Divide the Image into Small Cells (8x8 pixels):** The image is divided into small cells, each of size 8x8 pixels. This allows us to focus on smaller regions for better feature extraction.
3. **Compute Gradient Direction and Magnitude:** For each cell, the gradient direction (the angle at which the intensity of pixel values changes the most) and gradient magnitude (the strength of that change) are computed. This helps in capturing edge and texture details in the image.

Feature Extraction (Cont.)



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

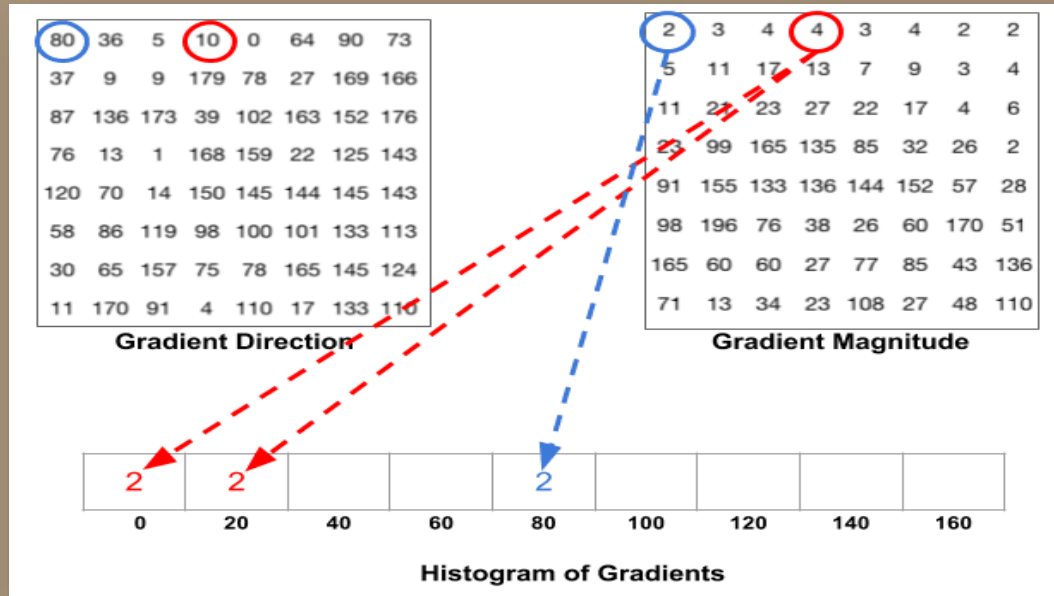
Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

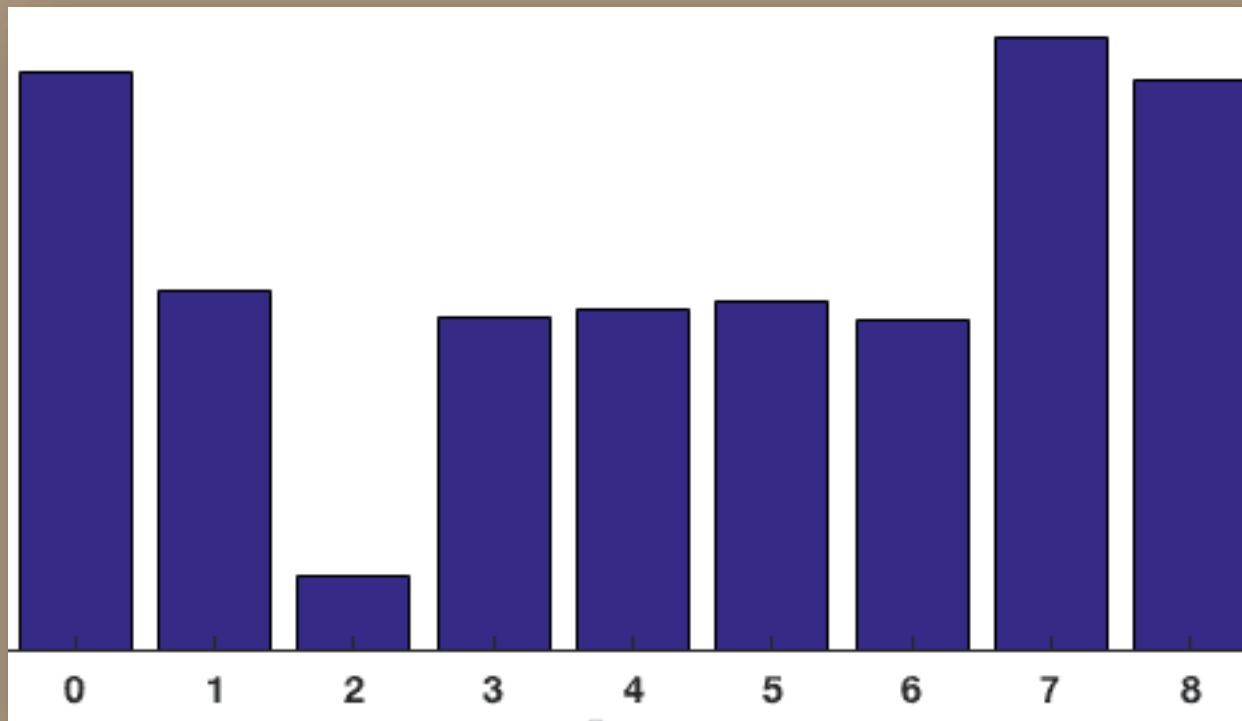
Gradient Direction

Feature Extraction (Cont.)

4. Create Histograms of Gradient Orientations: Within each cell, we create a histogram representing the distribution of gradient directions. Typically, 9 different gradient orientations are used (e.g., 0° , 20° , 40° , etc.), with each orientation representing how much of the gradient in that cell points in a particular direction.

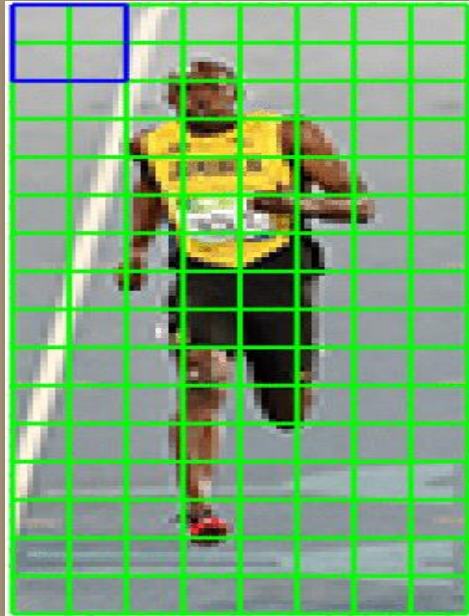


Feature Extraction (Cont.)



Feature Extraction (Cont.)

5. Group Cells into Blocks: Multiple cells are grouped together into blocks. In this case, we're using 2x2 blocks (4 cells per block). This grouping allows us to capture higher-level spatial information and makes the features more robust.



Feature Extraction (Cont.)

6. Normalize Each Block: To make the features invariant to changes in illumination and contrast, we normalize each block. This ensures that the feature representation is consistent regardless of the lighting conditions in the image

7. Build the Feature Vector: After block normalization, the features extracted from all cells and blocks can be combined into a single vector. This means that all the information extracted from the image (such as edges and directions) is combined into a long matrix, which serves as a digital representation of the image.

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_d \end{bmatrix}$$

Feature vector

08

Dimensionality Reduction



Dimensionality Reduction

- ❖ In our project, we applied Principal Component Analysis (PCA) after extracting HOG features to reduce the dimensionality of the data.
- ❖ HOG features generate a very large number of dimensions (features) for each image, many of which are redundant or less informative. To avoid overwhelming the machine learning model and to improve computational efficiency, we used PCA to compress the data while preserving its most important patterns.
- ❖ Specifically, we reduced the data to **200** principal components, which capture the majority of the variance (differences) in the dataset. This step helps the model focus on the essential information and reduces the risk of overfitting.

Dimensionality Reduction (Cont.)

PCA Steps ?

1. Data Preparation:

- Ensure the data is numerical and represented in matrix (rows & columns).

2. Calculate the Mean:

- Find the mean of each feature in the dataset to center the data around the origin (subtract the mean from each feature).

3. Normalize the Data:

- We normalize data in PCA to make sure all features (columns) are treated fairly.
- Normalization ensures that all features are on the same scale (mean = 0, standard deviation = 1).

Dimensionality Reduction (Cont.)

4. Compute the Covariance Matrix:

- Calculate the covariance matrix to understand how the features in the dataset vary with respect to each other.

5. Perform Eigenvalue Decomposition:

- Calculate the eigenvectors and eigenvalues of the covariance matrix.
- Eigenvectors represent the directions of the new axes (principal components), and eigenvalues represent the importance of each eigenvector.

Dimensionality Reduction (Cont.)

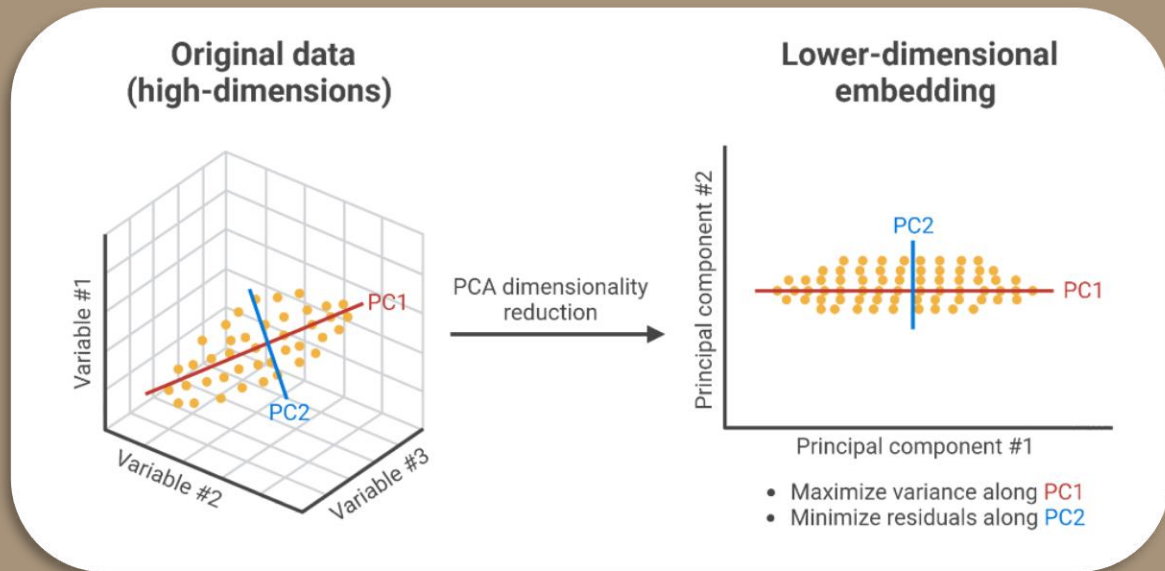
6. Select Principal Components:

- Sort the eigenvectors by their corresponding eigenvalues in descending order.
- Select the top N eigenvectors (the most significant ones) to define the new space.

7. Transform the Data (Projection):

- Data transformation means taking the original data and projecting it onto the new axes (principal components).
- Multiply the standardized data by the eigenvectors to get the new data in the principal component space.

Dimensionality Reduction (Cont.)



HOG Shape: (720, 20736)

After Applying PCA: (720, 200)

09

Model Building



Model Building

❖ After preparing the data (using HOG + PCA), we divided it into training and testing sets:

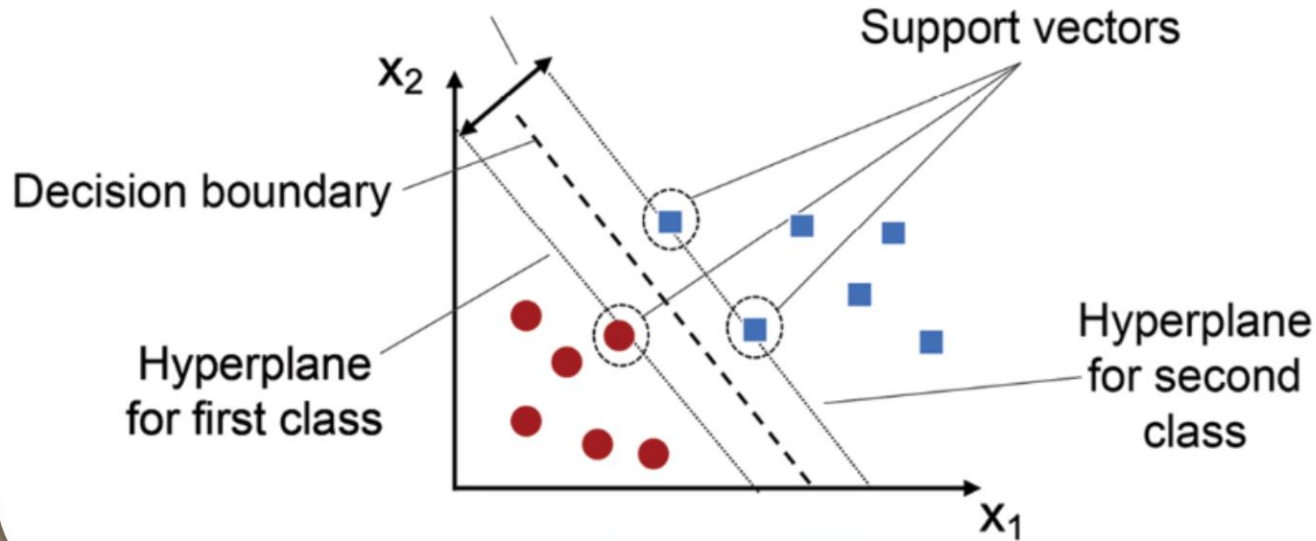
- It splits **80%** of the data for training and **20%** for testing, so we can later evaluate the model's performance on unseen data.

❖ Model Used: Support Vector Classifier (SVC)

- The Support Vector Classifier is a supervised machine learning algorithm used for classification tasks. It works by finding the optimal decision boundary that maximizes the margin between different classes. The closest data points to this boundary are called support vectors, and they play a critical role in defining the classifier. SVC can handle both linear and non-linear problems by using kernel functions to transform the data into higher-dimensional spaces, allowing for flexible and powerful separation of complex patterns.

Model Building (Cont.)

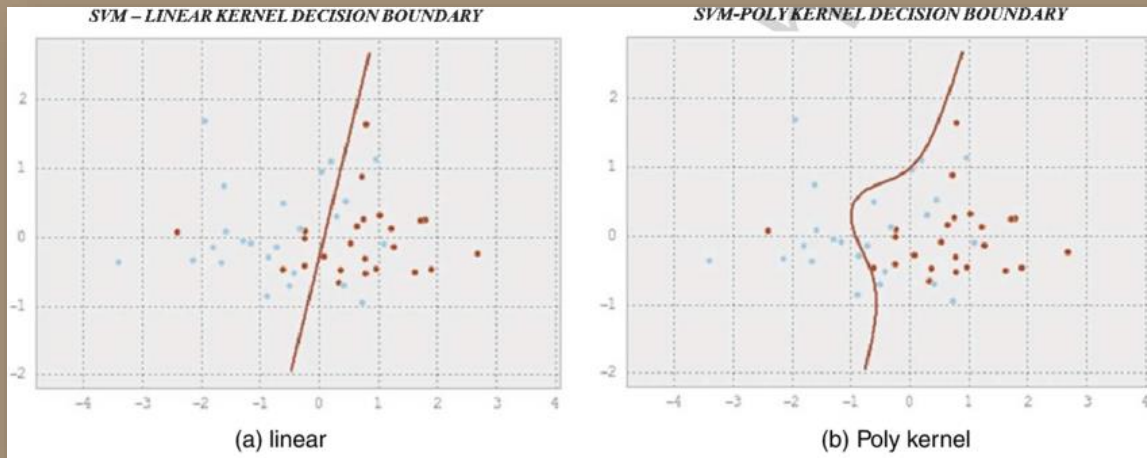
Margin (gap between decision boundary and hyperplanes)



Model Building (Cont.)

❖ We chose:

- ✓ **C=10** → fewer errors, smaller margin.
- ✓ **gamma='auto'** → controls influence of points; here it's set automatically.
- ✓ **kernel='poly'** → we use a polynomial kernel to allow learning non-linear patterns in the data.



10

Results and Evaluation



Results and Evaluation

- ❖ In this section, we present the results of our signature forgery detection system and evaluate its performance based on several key metrics:

Model Performance (Accuracy):

The model's performance was evaluated on the **20%** of the main dataset, and the following results were obtained:

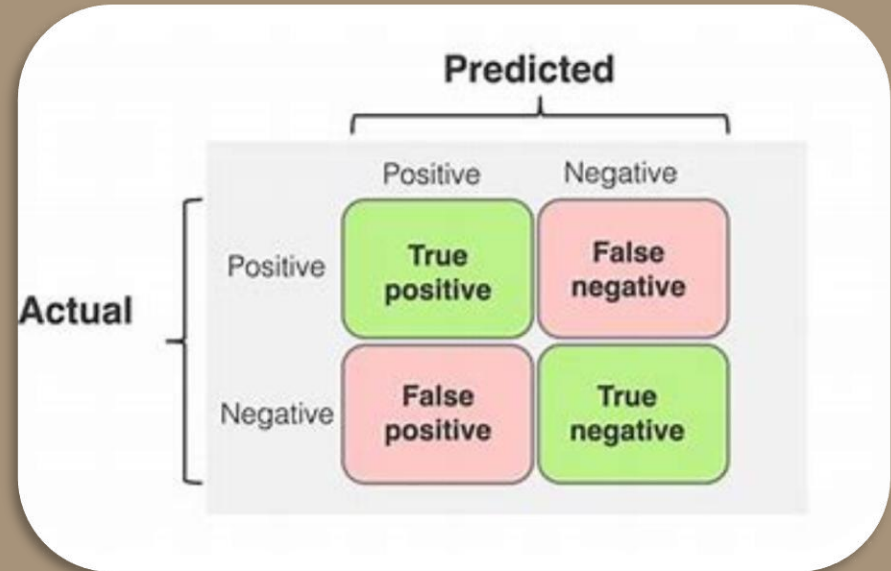
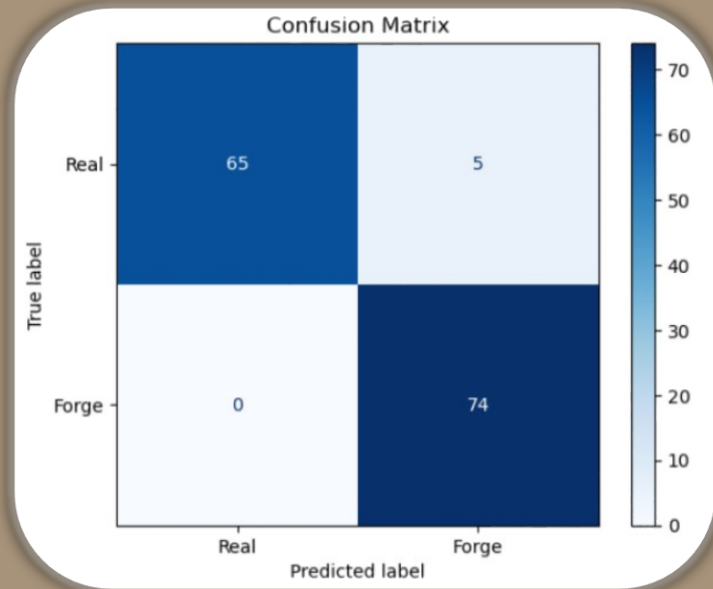
- Testing Accuracy: **96.53%**



Results and Evaluation (Cont.)

Confusion Matrix:

A confusion matrix was generated to evaluate the classification performance of the model. The matrix shows the number of correctly and incorrectly classified instances for both Real and Forged signatures.



Results and Evaluation (Cont.)

Precision:

- $\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$
- This measures how many of the signatures the model predicted as forged (or real) were correct.
- Result: **93.67%**

Recall:

- $\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$
- This shows how many of the actual forged (or real) signatures the model correctly identified.
- Result: **100.00%**

Results and Evaluation (Cont.)

F1 Score:

- **F1 Score = $2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$**
- **This is the balance between precision and recall, giving a single score that reflects both.**
- **Result: 96.73%**

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.93	0.96	70
1	0.94	1.00	0.97	74
accuracy			0.97	144
macro avg	0.97	0.96	0.97	144
weighted avg	0.97	0.97	0.97	144

11

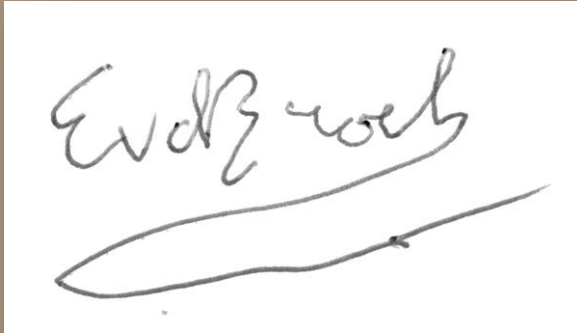
Testing



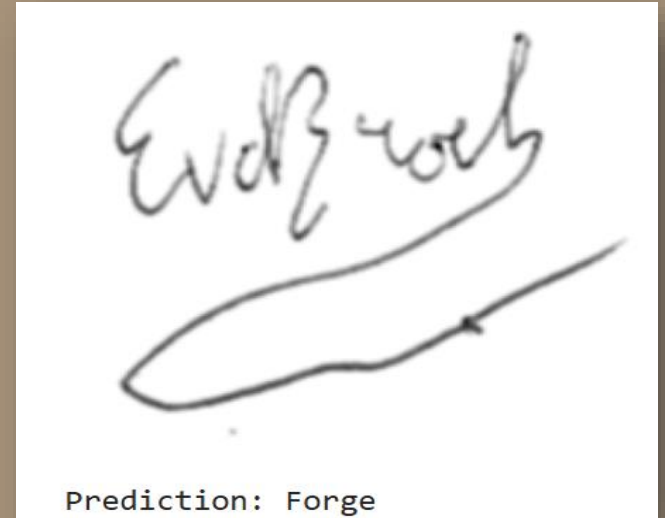
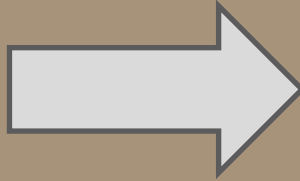
Testing

❖ How it works:

- We take a single new image, preprocess it, extract its HOG features.
- We transform these features using the same PCA fitted on the training dataset.
- We pass the result into the trained SVM model for prediction.

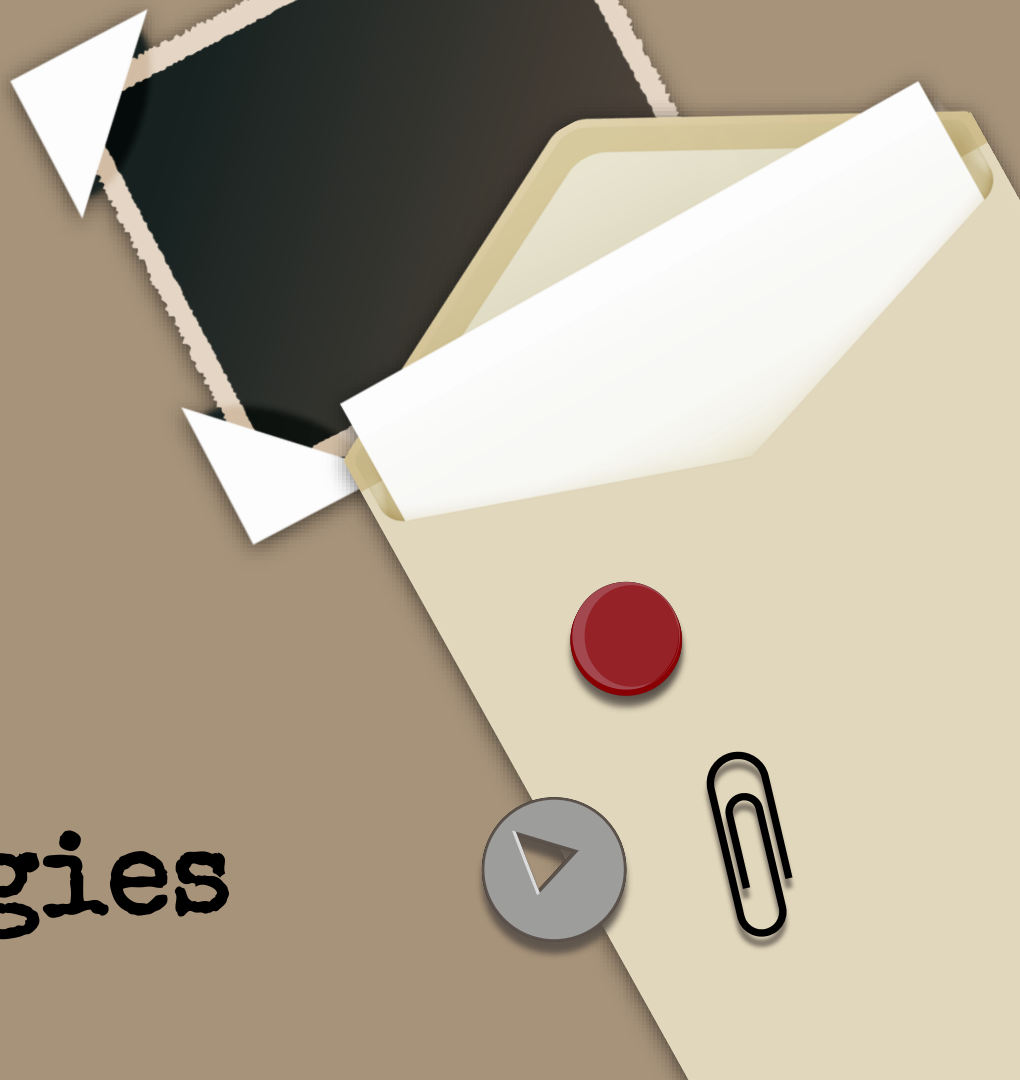


Input Image for Testing



12

Tools and Technologies



Tools and Technologies

- ❖ Our project utilized a variety of tools and technologies to ensure efficient and accurate signature forgery detection, The key technologies used are:

1. Programming Language:

Python: Chosen for its rich ecosystem of libraries for image processing, machine learning, and data analysis.

2. Development Environment:

Jupyter Notebook: Used for code development, testing, and presentation.



Tools and Technologies (Cont.)

3. Libraries and Frameworks:

- **os, cv2 (OpenCV):** Used for file handling and image processing, including reading images, grayscale conversion, resizing, and applying filters like Gaussian blur.
- **NumPy:** Used for numerical operations and efficient array handling.
- **scikit-image (hog):** Used for feature extraction through the Histogram of Oriented Gradients (HOG) technique.
- **Scikit-learn:** Utilized for building the machine learning model (SVM), dimensionality reduction (PCA), and evaluation metrics.
- **Matplotlib:** Used for visualizing results such as confusion matrices and performance metrics.

Tools and Technologies (Cont.)

4. Hardware:

The project was run on a standard laptop using its CPU, with moderate computational power, ensuring smooth execution for model training and evaluation.



13

Conclusion



Conclusion

- ❖ In conclusion, our Signature Forgery Detection system demonstrated impressive performance in distinguishing between real and forged signatures. The key outcomes of the project are as follows:

1. Successful Signature Forgery Detection:

The model effectively identified signature forgeries with high accuracy. The implementation of HOG (Histogram of Oriented Gradients) for feature extraction and PCA for dimensionality reduction significantly contributed to the system's performance.

2. Evaluation and Metrics:

Through various evaluation metrics such as accuracy, precision, recall, and F1-score, the system proved to be highly reliable in detecting forged signatures, making it a potential solution for security applications.

Conclusion (Cont.)

3. Potential for Real-World Application:

With further improvements, especially in terms of model scalability and the inclusion of deep learning techniques, the system can be deployed in real-world scenarios such as document verification in banking and legal sectors.

4. Final Thoughts:

This project laid the foundation for future research and development in the field of forensic signature analysis. By addressing the challenges faced during development and incorporating suggestions for future work, we are confident that this system can evolve into a robust and scalable solution for signature verification.

Thanks!

