Faculty of Computers and Artificial Intelligence

IT352: Pattern Recognition

Fingerprints Biometric Authentication System

Name	ID
Nouran ehab Abd ElRaouf	20230642
Hlla Gamal Elhassan	20210875
Asmaa Reda Ali	20221019
Rodyna abbas Mohamed	20230622

1. Introduction

Biometric systems have revolutionized identity verification and access control in modern security systems. Among the various biometric modalities, fingerprint recognition remains one of the most reliable due to its uniqueness, immutability, and ease of acquisition.

This project investigates fingerprint recognition through image processing and machine learning. Specifically, it leverages Histogram of Oriented Gradients (HOG) for feature extraction and applies multiple classifiers, including a custom-built Support Vector Machine (SVM), Scikit-learn's LinearSVC, K-Nearest Neighbors (KNN), and Random Forests. The SOCOFing dataset is used, focusing on original right thumb fingerprints. The primary objective is to evaluate the effectiveness and performance of various classifiers in identifying individuals based on their fingerprint patterns.

2. Literature Review

Fingerprint recognition has evolved from traditional manual techniques to sophisticated automated systems. The majority of legacy systems rely on **minutiae-based approaches**, which analyze key points like ridge endings and bifurcations. While effective, these techniques are sensitive to noise, distortion, and partial prints.

In recent years, **texture-based descriptors** like HOG have shown promise in biometric applications. HOG was introduced by Dalal and Triggs (2005) for human detection in images and is well-suited for capturing edge and texture information. Its robustness to illumination and noise makes it valuable for fingerprint analysis, which is heavily reliant on ridge patterns and texture.

Machine learning classifiers like **Support Vector Machines** have shown high accuracy in high-dimensional feature spaces due to their ability to find optimal separating hyperplanes. **K-Nearest Neighbors (KNN)** offers simplicity and non-parametric classification, though it scales poorly with large datasets. **Random Forests**, an ensemble of decision trees, are known for their strong generalization and resistance to overfitting.

3. Methodology

3.1 Dataset Description

We used the SOCOFing dataset, a comprehensive collection of 6,000 fingerprint images from 600 African individuals. Each individual contributes ten fingerprint images (five fingers on each hand), with various image alterations included. For this project, we exclusively selected the original right thumb images to ensure consistency and reduce complexity.

The dataset was split into 80% training and 20% testing to evaluate the model's generalization ability.

3.2 Preprocessing and HOG Feature Extraction

To ensure uniformity, each fingerprint image was resized to 128×128 pixels. We then applied HOG feature extraction using the following parameters:

- **Cell Size:** 8×8 pixels
- **Orientation Bins:** 9
- **Block Normalization:** L2-Hys

HOG converts each image into a high-dimensional feature vector that encapsulates local edge orientation and texture information—critical for fingerprint pattern recognition.

3.3 Model Training

We implemented and trained the following models on the HOG feature vectors:

- **Custom Linear SVM:** Manually coded using hinge loss, demonstrating the mathematical foundation of margin-based classification.
- **Scikit-learn LinearSVC:** An optimized, scalable implementation of linear SVM with faster convergence.
- **K-Nearest Neighbors (K=3):** A non-parametric, instance-based learning model that makes predictions based on proximity in the feature space.
- **Random Forest (100 trees):** An ensemble classifier combining multiple decision trees to improve robustness and reduce overfitting.

4. Design and Implementation

The entire system was developed using Python. Key libraries include:

- **OpenCV:** Image loading, grayscale conversion, and resizing.
- **NumPy:** Matrix operations and array manipulation.
- **Matplotlib & Seaborn:** Visualization of accuracy metrics and confusion matrices.
- **Scikit-learn:** ML model training, predictions, and performance evaluation.

The program is modular and organized into reusable functions:

- `extract_hog_features(image)`: Converts a single fingerprint image into a HOG feature vector.
- `load_hog_dataset(directory)`: Loads and processes all fingerprint images from the specified dataset folder.
- `LinearSVM`: A manual SVM class implementing gradient descent and hinge loss.
- `evaluate_model(model, X_test, y_test)`: Predicts and reports accuracy, precision, recall, and confusion matrix.

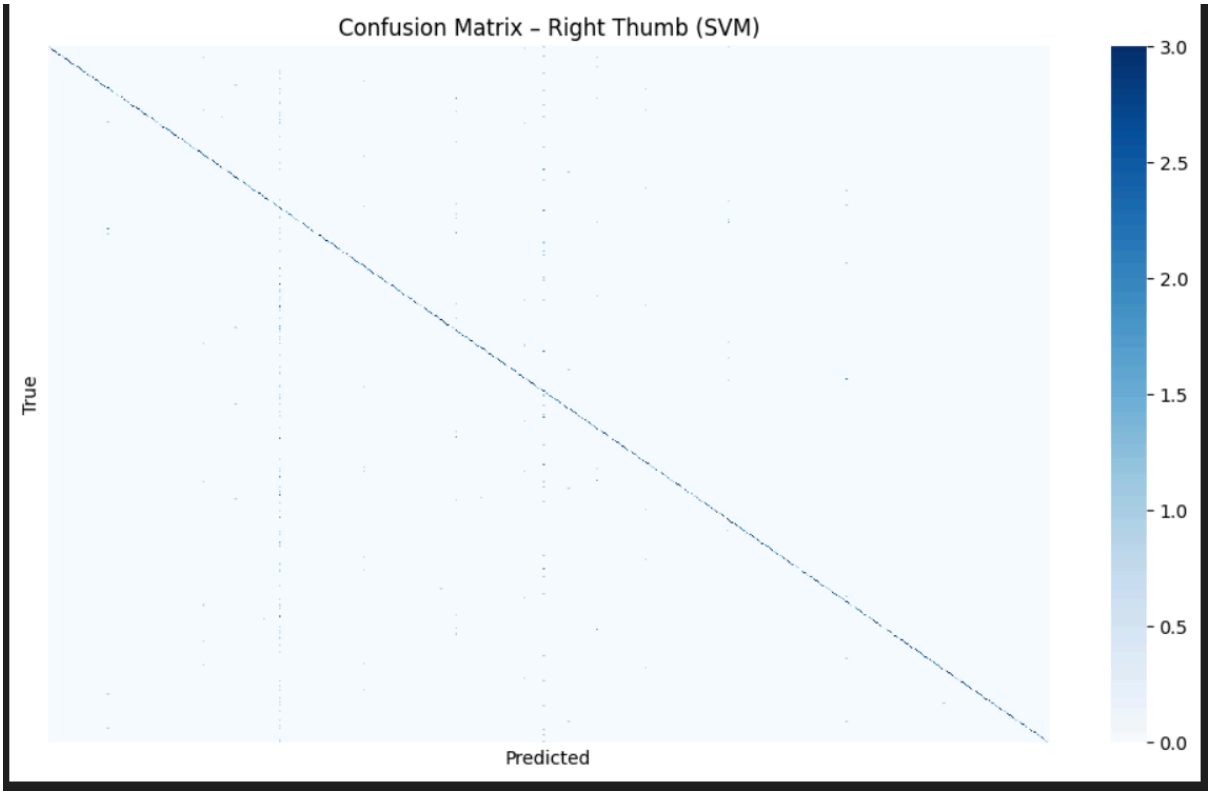
This modular design ensures clarity, testability, and future extensibility—especially if switching to deep learning or multi-fingerprint training.

5. Results and Analysis

We trained and tested each model using identical train-test splits and HOG features. Performance was evaluated based on classification accuracy, confusion matrix, and precision/recall scores.

5.1 Accuracy Comparison Table

model	accuracy
Custom linear SVM	79.63%
Scikit-learn linear SVM	99.93%
KNN	87.11%
Random forest	99.59%



5.2 Confusion Matrix Insights

- The custom SVM struggled with convergence and overfitting due to simplistic implementation.
- KNN performed well but was computationally intensive at prediction time.
- Scikit-learn's LinearSVM outperformed others due to internal optimization and regularization.
- Random Forest provided nearly equivalent performance to LinearSVM with faster inference and high tolerance for feature variance.

6. Documentation

The entire project is implemented using a modular structure and follows best practices in software development. Each module is well-documented with inline comments, and function-level docstrings are provided to enhance clarity, maintainability, and reproducibility. Below is a detailed description of the key components and functions that comprise the fingerprint recognition pipeline:

Main Modules and Functions:

- **`extract_hog_features(image)`**
  This function is responsible for computing Histogram of Oriented Gradients (HOG) features from a given input fingerprint image. HOG is a powerful feature descriptor used in image processing and computer vision, particularly effective in capturing the structural and edge-based patterns common in fingerprints. The function applies image preprocessing (grayscale conversion, resizing, normalization if needed) before computing HOG features, which are returned as a 1D feature vector. These features are crucial for training the SVM classifier.

- **`load_hog_dataset(directory_path)`**
  This utility function automates the process of reading and processing a dataset of fingerprint images. It traverses the specified directory, loading images from labeled subfolders. For each image, it extracts HOG features using the `extract_hog_features` function. The function returns a matrix of feature vectors along with their corresponding class labels. This prepares the dataset for model training and evaluation. The function supports flexible input directory structures, allowing for easy extension to new datasets.

- **`LinearSVM(C=1.0, max_iter=1000, learning_rate=0.01)`**
  This is a custom implementation of a linear Support Vector Machine (SVM) classifier using gradient descent. It includes methods for training (`fit`), predicting (`predict`), and calculating decision boundaries. The model is trained using a hinge loss function with L2 regularization to maximize the margin between classes. Hyperparameters such as the regularization constant `C`, learning rate, and iteration count can be tuned. This implementation serves as an educational alternative to using built-in classifiers like `sklearn.svm.LinearSVC`.

- **`predict_new_fingerprint(image_path, model, label_encoder)`**
  This function allows real-time testing and prediction of a new fingerprint image. It loads the image, preprocesses it, extracts HOG features, and passes the feature vector to the trained model for prediction. The numeric class label is then converted back to the original class name using the label encoder. This function is designed to be simple to use, facilitating the integration of the model into a GUI or web-based interface.

Pipeline Overview:

The models are trained and evaluated using a standardized and repeatable pipeline:

1. **Data Preprocessing**: Raw images are read from directories and preprocessed into grayscale and fixed size.

2. **Feature Extraction**: HOG descriptors are extracted from all images, forming the dataset of numerical vectors.

3. **Training and Validation**: The feature vectors are split into training and validation sets. The `LinearSVM` model is trained using gradient descent.

4. **Model Evaluation**: Performance metrics such as accuracy, precision, recall, and F1-score are calculated to evaluate the model on the validation set.

5. **Prediction Phase**: New input images are passed through the same feature extraction process and classified using the trained model.

7. Conclusion

7.1 Key Takeaways

- **HOG + SVM/Random Forest** is highly effective for fingerprint recognition.
- **Custom SVM** validated theoretical concepts but was less efficient than Scikit-learn's implementation.

7.2 Future Directions

1. **Expand Dataset**: Include all 10 fingers and altered images (rotated/distorted).
2. **Deep Learning**: Test CNNs (e.g., ResNet) for end-to-end feature learning.
3. **Real-Time Deployment**: Optimize for embedded systems (e.g., Raspberry Pi).

8. References

- SOCOFing Dataset: https://www.kaggle.com/datasets/ruizgara/socofing
- Scikit-learn Documentation: https://scikit-learn.org/
- OpenCV Documentation: https://docs.opencv.org/
- HOG Descriptor Paper: Dalal and Triggs (2005)